

SOCIEDADE EDUCACIONAL PINHALZINHO
HORUS FACULDADES

Cristian Feldkircher

**DESENVOLVIMENTO DE SISTEMA PARA FISCALIZAÇÕES NO COMBATE DO
AEDES AEGYPTI**

Pinhalzinho/SC

2024

CRISTIAN FELDKIRCHER

**DESENVOLVIMENTO DE SISTEMA PARA FISCALIZAÇÕES NO COMBATE DO
AEDES AEGYPTI**

Trabalho de Conclusão de Curso apresentado à
Horus Faculdades, como parte dos requisitos para
obtenção do grau de Bacharel em Sistemas de
Informação

Orientador(a): Prof. Esp. Ricardo Jeferson Hendges

Pinhalzinho/SC

2024

AGRADECIMENTOS

Primordialmente gostaria de agradecer a todos os meus familiares e amigos, por todo suporte e contribuição para conclusão desta pesquisa.

Ao meu orientador Ricardo Jeferson Hendges, por todo apoio e comprometimento, buscando me auxiliar da melhor forma.

Ao meu pai, Asabido Feldkircher, que sempre me auxiliou e sempre se fez presente em meu crescimento profissional.

A minha mãe, Inês Rosenbach Feldkircher, que sempre me auxiliou e sempre se fez presente em meu crescimento profissional.

A minha namorada Carolina Cecatto, por todo apoio e companheirismo, me acompanhou em toda minha trajetória profissional e acadêmica.

Aos colegas de curso, professores e a todos os colaboradores da Horus Faculdades, por todo conhecimento repassado, companheirismo e apoio, em toda trajetória na instituição buscando prestar o auxílio possível.

Ao vereador, Juliano de Almeida, por ser um dos “pilares” para que a pesquisa pudesse ser desenvolvida.

Ao secretário da saúde, Claudir Kollett, por toda base de conhecimento fornecida, comprometimento e auxílio.



CRISTIAN FELDKIRCHER

2024

“A tecnologia tornou possível a existência de grandes populações. Grandes populações agora tornam a tecnologia indispensável”.

Joseph Krutch.

FELDKIRCHER, Cristian. **DESENVOLVIMENTO DE SISTEMA PARA FISCALIZAÇÕES NO COMBATE DO Aedes Aegypti**, 2024. 43p. Trabalho de Conclusão de Curso Bacharelado em Sistemas de Informação. Horus Faculdades, Pinhalzinho/SC. 2024.

RESUMO

A endemia de dengue causada pelo mosquito *Aedes aegypti*, designou diversas mudanças na saúde pública, com isso levando os órgãos municipais a tomarem medidas para eliminar os focos do mosquito. Como um software de gestão pode otimizar o controle da proliferação do *Aedes aegypti*, impactando diretamente na prevenção da dengue e na saúde pública municipal? Com base na pesquisa efetuada, e sobre um estudo aprofundado da questão, o objetivo geral é realizar o desenvolvimento e documentação de um software de gestão para que o combate a proliferação de mosquitos *Aedes aegypti* seja mais eficaz e controlado, tendo como base o processo efetuado pelo órgão responsável pelo combate a dengue, e toda a comunidade do município. Foram coletados dados para que pudesse ter melhor entendimento de todo processo de fiscalização que é efetuado, com intuito de melhorar e tornar a fiscalização sobre os focos mais eficiente e com melhor precisão, com resultados confirmados de dengue no município, sendo 169 suspeitos notificados, 116 suspeitos descartados, 22 casos confirmados de dengue e 31 aguardando exames, dados estes do primeiro trimestre do ano de 2024. Em meio a este cenário foi elaborado um sistema que atenda o fluxo de trabalho dos agentes de fiscalização, da secretaria de endemia da dengue e também da população para que possa ter acompanhamento dos casos e feedback do trabalho efetuado pelo município, utilizando tecnologias como Vue, Node e Postgres como base para desenvolvimento do software. Espera-se como resultado final mais agilidade no processo de eliminação do foco da dengue no município de Pinhalzinho.

Palavras-chave: *Aedes aegypti*; Agentes; Focos; Dengue;

FELDKIRCHER, Cristian. **DEVELOPMENT OF A SYSTEM FOR SUPERVISION IN THE FIGHT OF Aedes Aegypti**, 2024. 43 pages. Bachelor's Thesis in Information Systems. Horus Colleges, Pinhalzinho/SC. 2024.

ABSTRACT

The dengue endemic caused by the *Aedes aegypti* mosquito has led to various changes in public health, prompting municipal authorities to take measures to eliminate mosquito breeding sites. How can a management software optimize the control of *Aedes aegypti* proliferation, directly impacting dengue prevention and municipal public health? Based on the conducted research and an in-depth study of the issue, the overall objective is to develop and document management software to make the combat against the proliferation of *Aedes aegypti* mosquitoes more effective and controlled. This software will be based on the process carried out by the authority responsible for dengue control and involve the entire community of the municipality. Data were collected to better understand the inspection process, aiming to improve and make the inspection of breeding sites more efficient and precise. Confirmed dengue results in the municipality include 169 suspected cases notified, 116 suspected cases discarded, 22 confirmed cases of dengue, and 31 cases awaiting examination, all from the first quarter of 2024. In this scenario, a system has been developed to meet the workflow of inspection agents, the dengue endemic department, and the population, allowing for case monitoring and feedback on the municipality's work. Technologies such as Vue, Node, and Postgres are used as the basis for software development. The expected final result is increased agility in eliminating dengue breeding sites in the municipality of Pinhalzinho.

Keywords: *Aedes aegypti*; Agents; Hotspots; Dengue.

ÍNDICE DE FIGURAS

| | |
|--|----|
| Figura 01: Distribuição do número de casos e da incidência de dengue por faixa etária, Pinhalzinho, Santa Catarina, 2015-2016..... | 21 |
| Figura 02: Distribuição do número de casos confirmados de dengue por semana epidemiológica da notificação, Pinhalzinho, Santa Catarina, 2016..... | 22 |
| Figura 03: Distribuição mensal do número de focos de <i>Aedes aegypti</i> em comparação com a temperatura média mensal e a umidade relativa do ar, Pinhalzinho, Santa Catarina, 2015-2018..... | 22 |
| Figura 04: Desenho esquemático da estrutura hierárquica da área de controle de vetores..... | 25 |
| Figura 05: Ciclo de vida do mosquito <i>Aedes aegypti</i> | 26 |
| Figura 06: Classificação dos índices de infestação por <i>Aedes aegypti</i> | 27 |
| Figura 07: Diagrama de entidade e relacionamento do software..... | 34 |
| Figura 08: Diagrama de Caso de Uso do software..... | 35 |
| Figura 09: Diferença node.js versus modelo tradicional..... | 41 |
| Figura 10: Exemplo de utilização do node.js..... | 42 |
| Figura 11: Exemplo utilizando express.js..... | 43 |
| Figura 12: Prototipação do projeto no Figma..... | 48 |
| Figura 13: Repositório do software no Github..... | 48 |
| Figura 14: Dockerfile presente no protótipo..... | 49 |
| Figura 15: Imagem Docker referente ao backend..... | 50 |
| Figura 16: Utilização do PostgreSQL (criação das tabelas)..... | 51 |
| Figura 17: Dependências utilizadas para backend..... | 52 |
| Figura 18: Estrutura das pastas presente no backend..... | 53 |
| Figura 19: Exemplo de método Post (inserção) na pasta services..... | 54 |
| Figura 20: Exemplo de método Post (inserção) na pasta routes..... | 54 |
| Figura 21: Exemplo de método Post (inserção) na pasta controllers..... | 54 |
| Figura 22: Exemplo de documentação utilizando Swagger.js..... | 55 |
| Figura 23: Instalação Vue..... | 56 |
| Figura 24: Exemplo do comando vue create..... | 57 |
| Figura 25: Estrutura das pastas do Front-end no sistema..... | 57 |
| Figura 26: Exemplo de uso do Vue Router..... | 58 |
| Figura 27: Exemplo de uso do Vue Router..... | 58 |
| Figura 28: Interface inicial do sistema (mobile)..... | 60 |
| Figura 29: Interface ao acessar “Saiba Mais” (mobile)..... | 60 |
| Figura 30: Interface ao acessar “Denuncie Focos de Mosquito” (mobile)..... | 61 |
| Figura 31: Interface de login do sistema..... | 62 |
| Figura 32: Interface do dashboard inicial..... | 62 |
| Figura 33: Interface de adicionar novo foco de dengue..... | 63 |
| Figura 34: Interface de controle de fiscalizações..... | 64 |

| | |
|---|----|
| Figura 35: Interface de monitoramento..... | 64 |
| Figura 36: Interface de usuários..... | 65 |
| Figura 37: Interface de cadastro de usuários..... | 66 |
| Figura 38: Interface de municípios..... | 66 |
| Figura 39: Interface do relatório de Fiscalizações..... | 67 |
| Figura 40: PDF do relatório de Fiscalizações..... | 68 |
| Figura 41: Interface do relatório de focos..... | 69 |
| Figura 42: Interface de controle de denúncias..... | 69 |
| Figura 43: Interface de controle de logs..... | 70 |
| Figura 44: Interfaces para Cadastrar Fiscalização..... | 71 |
| Figura 45: Interfaces para inclusão e remoção de focos..... | 72 |

LISTA DE ABREVIACÕES

DATASUS - *Departamento de Informática do Sistema Único de Saúde*
DENV - *Vírus da Dengue*
DIVE - *Diretoria de Vigilância Epidemiológica*
FIOCRUZ - *Fundação Oswaldo Cruz*
IBGE - *Instituto Brasileiro de Geografia e Estatística*
IDH - *Índice de Desenvolvimento Humano*
OPAS - *Organização Pan-Americana da Saúde*
PR - *Paraná*
RJ - *Rio de Janeiro*
SC - *Santa Catarina*
SINAN - *Sistema de Informação de Agravo de Notificação*
TABNET - *Ferramenta de consulta do Datasus*
DEN 1 - *Dengue vírus sorotipo 1*
DENV-2 - *Dengue vírus sorotipo 2*
DENV-3 - *Dengue vírus sorotipo 3*
LIRAA - *Levantamento Rápido de Índices de Infestação do Aedes aegypti*
UML - *Unified Modeling Language*
WEB - *World Wide Web*
SCR - *Source*
HTML - *HyperText Markup Language*
CSS - *Cascading Style Sheets*
YAML - *Yet Another Markup Language*
API - *Application Programming Interface*
JSON - *JavaScript Object Notation*
SQL - *Structured Query Language*
HTTP - *Hypertext Transfer Protocol*
IoT - *Internet of Things*
REST - *Representational State Transfer*
WSDL - *Web Services Description Language*
DB - *Database (usado em contexto de banco de dados)*
OSS - *Open Source Software (implícito no contexto de PostgreSQL e Docker)*

SOAP - Simple Object Access Protocol (Protocolo Simples de Acesso a Objetos)

RAML - RESTful API Modeling Language (Linguagem de Modelagem de API RESTful)

Git - Sistema de Controle de Versão

VS Code - Visual Studio Code

ACE - Agente de Controle de Endemias

ACS - Agente Comunitário de Saúde

C# - Linguagem de programação C Sharp

JSPDF - Biblioteca JavaScript para gerar PDFs

SUMÁRIO

| | |
|---|-----------|
| 1 INTRODUÇÃO..... | 14 |
| 1.1 TEMA..... | 16 |
| 1.2 DELIMITAÇÃO DO TEMA..... | 16 |
| 1.3 JUSTIFICATIVA..... | 17 |
| 1.4 PROBLEMA..... | 17 |
| 1.5 OBJETIVOS DA PESQUISA..... | 18 |
| 1.5.1 Objetivo geral..... | 18 |
| 1.5.2 Objetivos específicos..... | 18 |
| 2 REFERENCIAL TEÓRICO..... | 19 |
| 2.1 ORIGEM DA PALAVRA AEADES AEGYPTI..... | 19 |
| 2.2 SURGIMENTO DO AEADES AEGYPTI..... | 19 |
| 2.2.1 Primeiros casos de Dengue no Brasil..... | 19 |
| 2.2.2 Primeiros casos de Dengue em Pinhalzinho e no estado de Santa Catarina | 20 |
| 2.3 COMBATE CONTRA A PROLIFERAÇÃO DO MOSQUITO DA DENGUE..... | 23 |
| 2.3.1 Trabalho dos órgão públicos e agentes de combate a endemias..... | 24 |
| 2.3.2 O controle do criadouro..... | 25 |
| 2.3.3 Participação da sociedade no combate a Dengue..... | 27 |
| 3 PROCEDIMENTOS METODOLÓGICOS..... | 29 |
| 3.1 TIPO DE ESTUDO..... | 29 |
| 3.1.1 Quanto aos objetivos..... | 29 |
| 3.1.2 Quanto aos procedimentos..... | 29 |
| 3.2 UNIVERSO DA PESQUISA..... | 30 |
| 3.2 COLETA E TRATAMENTO DE DADOS..... | 30 |
| 4 ANÁLISE DOS REQUISITOS..... | 31 |
| 4.1 REQUISITOS FUNCIONAIS..... | 31 |
| 4.2 REQUISITOS NÃO FUNCIONAIS..... | 32 |
| 4.3 UML..... | 33 |
| 4.3.1 Diagrama de Entidade e Relacionamento..... | 33 |
| 4.3.3 Diagrama de Caso de Uso..... | 34 |
| 5 FERRAMENTAS, SISTEMAS, TECNOLOGIAS E METODOLOGIAS..... | 36 |
| 5.1 VISUAL STUDIO CODE..... | 36 |
| 5.2 POSTMAN..... | 37 |
| 5.3 GITHUB..... | 37 |
| 5.4 DOCKER..... | 38 |
| 5.5 JAVASCRIPT..... | 39 |
| 5.5.1 Vue.js..... | 39 |
| 5.5.2 Node.js..... | 40 |
| 5.5.2.1 Express.js..... | 42 |

| | |
|--|-----------|
| 5.5.2.2 Swagger.js..... | 43 |
| 5.5.2.3 PG..... | 44 |
| 5.5.2.3 JSPDF..... | 44 |
| 5.5.2.3 Leaflet.js..... | 45 |
| 5.6 BANCO DE DADOS..... | 45 |
| 5.6.1 PostgreSQL..... | 46 |
| 6 DESENVOLVIMENTO DO PROJETO..... | 47 |
| 6.1 PLANEJAMENTO..... | 47 |
| 6.1.1 Prototipação no Figma..... | 47 |
| 6.1.2 Versionamento de código..... | 48 |
| 6.2 BACKEND..... | 49 |
| 6.2.1 Configuração docker..... | 49 |
| 6.2.2 Estrutura do Banco de Dados..... | 50 |
| 6.2.3 Desenvolvimento API..... | 52 |
| 6.2.4 Documentação utilizando Swagger.js..... | 55 |
| 6.3 FRONTEND..... | 56 |
| 6.3.1 Instalação do Vue..... | 56 |
| 6.3.3 Estrutura de pastas..... | 57 |
| 6.3.4 Interfaces..... | 59 |
| 6.3.4.1 Interfaces para a comunidade (web e mobile)..... | 59 |
| 6.3.4.2 Interfaces de gestão do secretário (web e mobile)..... | 61 |
| 6.3.4.3 Interfaces para ACE's (mobile)..... | 70 |
| 7 CONSIDERAÇÕES FINAIS..... | 73 |
| 7.1 TRABALHOS FUTURO..... | 73 |
| 8 CRONOGRAMA..... | 74 |
| REFERÊNCIAS BIBLIOGRÁFICAS:..... | 75 |

1 INTRODUÇÃO

A endemia de dengue causada pelo mosquito *Aedes aegypti*, teve seus primeiros relatos no Brasil, nos anos de 1916 em São Paulo e 1923 em Niterói - RJ, entretanto a primeira epidemia de dengue surgiu no estado de Roraima, na cidade de Boa Vista, no ano de 1982. Após esse ocorrido a dengue foi reaparecer no ano de 1986 de forma epidêmica, ocorrendo em 3 estados brasileiros (Alagoas, Ceará e Rio de Janeiro) sendo a maior delas no estado do Rio de Janeiro, onde atingiu mais de um milhão de pessoas. (FUNASA et al, 2002)

A dengue é uma das doenças que vem aumentando todos os anos com mais frequência no Brasil, isso deve-se muito ao crescimento desordenado das cidades e as deficiências sobre coleta e destinação correta do lixo, juntamente com o abastecimento irregular de água. No Brasil em 1995 haviam 1.752 municípios com focos de mosquitos, já no ano de 2008 houve um grande aumento chegando a 4.006 municípios. (FUNASA et al, 2009)

Os casos de dengue vêm se repetindo todos os anos, tendo seus maiores aumentos no verão, por conta das chuvas e do calor. Aumentando assim, o número de locais com água parada e também os criadouros, e conseqüentemente os casos. O mosquito pode viver de 30 a 35 dias, sendo que a fêmea do *Aedes aegypti* põe ovos de 4 a 6 vezes durante a sua vida, podendo colocar mais de 100 ovos, com isso, aumentando cada vez mais a proliferação do mosquito. (FUNASA et al, 2009)

O órgão responsável pelo combate dos focos do mosquito é o Ministério da Saúde, o qual apresentou o “Programa Nacional de Controle da Dengue” (PNCD) no ano de 2002. O PNCD vem com intuito de incorporar as experiências nacionais e internacionais no controle da dengue, elaborando programas permanentes para a erradicação do mosquito *Aedes aegypti*. Contudo, os órgãos municipais também criaram iniciativas para o combate a dengue, criando assim a Secretária de Combate a endemia da Dengue, com uma função essencial para a comunidade, tendo o Agente de Vigilância Epidemiológica, o qual efetua visitas a comércios e residências. (FUNASA et al, 2002)

A comunidade necessita de auxílio no combate da propagação do *Aedes aegypti*, o agente de vigilância, efetua visitas com um cronograma pré definido pela secretaria do município, tendo como principal objetivo encontrar os focos, efetuar a devida eliminação do mesmo e orientar o morador(a) para que seja feita a eliminação do foco encontrado.

Atualmente, há uma enorme dificuldade para o agente nas anotações de suas visitas a um imóvel, é efetuado preenchimento de anotações manualmente, tendo retrabalho para a secretaria para obter um relatório sobre as visitas e repassar as mesmas para o digital.

A secretaria também possui dificuldades no processo para receber denúncias anônimas da população. Sendo assim, pensa-se em um método de denúncias sobre focos que não foram encontrados pelos agentes, melhoria no processo de feedback para a comunidade sobre as denúncias feitas e resolvidas pela secretaria de combate a dengue, e também relatórios para que possam ter melhores levantamentos dos resultados.

Em meio ao cenário atual, um software para gestão de denúncias e controle dos criadouros de mosquitos é de extrema importância, atualmente os números de casos de dengue vem crescendo rapidamente sem ter um controle eficaz para combate e para que se apresente resultados para a comunidade.

Visando efetuar a melhoria sobre todo processo de combate a proliferação do mosquito, o atual projeto busca como inovação a elaboração e desenvolvimento do software integrado com geolocalização, para que a comunidade possa acompanhar com exatidão todos os locais onde os focos de dengue estão confirmados no município.

Esta pesquisa acrescentará no âmbito científico e social no que diz respeito à forma em que a população irá efetuar denúncias sobre os casos de dengue. E agregará ao agente de vigilância, tornando-se ideal para utilização no trabalho do dia a dia, e para melhores resultados na saúde da população. Assim, possibilitando que o município, como principal órgão responsável pelo combate a doença, possa apresentar resultados positivos para a população, e também para os demais municípios da região.

Com base na pesquisa efetuada, e sobre um estudo aprofundado da questão, o objetivo geral é realizar o desenvolvimento e documentação de um software de gestão para que o combate a proliferação de mosquitos *Aedes aegypti* seja mais eficaz e controlado, tendo como base o processo efetuado pelo órgão responsável pelo combate a dengue, e toda a comunidade do município.

Logo, os objetivos específicos são: Estudar conceitos referentes ao sistema de combate a proliferação de mosquitos *Aedes aegypti*, e também sobre as necessidades da comunidade com o combate do mesmo; Realizar um levantamento dos principais aspectos relacionados aos procedimentos efetuados pelo agente de vigilância da dengue; Identificar modelos de gestão para controle mais adequado e com mais afinco, visando otimizar o trabalho do agente de fiscalização, e também melhorar a confiança da comunidade ao serviço público prestado pelo município.

O intuito da pesquisa é disponibilizar à população e ao município dados confiáveis para que possam ter conhecimento dos locais de foco de dengue, denúncias efetuadas pela população e resultados para acompanhar o andamento de resolução de criadouros de

mosquitos. E também, levantar dados para que a secretaria possa ter um controle melhor sobre as visitas que o agente de vigilância efetua no município.

O projeto aborda as considerações a respeito do tema desta pesquisa, apresenta toda proposta e delimita o tema, trazendo informações de como pode contribuir no âmbito acadêmico e social. Composto por diversas análises para que seja feito o levantamento do quanto avançado os órgãos de controle da dengue e a população estão sobre o assunto da doença transmitida pelo *Aedes aegypti*.

A partir do atual trabalho efetuado sobre os processos de combate contra a proliferação do mosquito, é possível contextualizar o desenvolvimento de um software a ser utilizado nas análises, tendo como objetivo apresentar a metodologia aplicada no estudo, a partir de pesquisas, parametrizar um software para utilização prol da saúde pública.

Assim, tem-se o seguinte problema de pesquisa: Como um software pode otimizar o controle da proliferação do *Aedes aegypti*, impactando diretamente na prevenção da dengue e na saúde pública municipal?

1.1 TEMA

A crescente proliferação do mosquito *Aedes aegypti* e os surtos de dengue no Brasil destacam a necessidade de métodos mais eficazes de controle. O desenvolvimento de um software de gestão com geolocalização surge como uma solução inovadora para enfrentar este desafio.

Esse software permitirá monitorar em tempo real os focos de dengue, facilitando o trabalho dos agentes de vigilância e otimizando o processo de resposta às denúncias da comunidade. Com dados precisos e atualizados, será possível melhorar a gestão das ações de combate ao mosquito, resultando em uma prevenção mais eficaz e um impacto positivo na saúde pública.

1.2 DELIMITAÇÃO DO TEMA

A proposta deste projeto é desenvolver um software de gestão integrada com geolocalização para otimizar o controle da proliferação do mosquito *Aedes aegypti*, transmissor da dengue, no município de Pinhalzinho/SC. O foco será aprimorar a eficácia das ações dos agentes de vigilância epidemiológica e a resposta às denúncias da comunidade sobre possíveis criadouros. Este software permitirá monitorar em tempo real os focos de

dengue, gerando dados precisos para análise e tomada de decisões, além de facilitar a comunicação entre os órgãos de saúde e a população. O estudo se concentrará na implementação tecnológica, nas necessidades dos agentes de vigilância, e na melhoria dos processos de coleta e gestão de informações.

1.3 JUSTIFICATIVA

A necessidade de um controle mais eficaz no combate contra a proliferação do mosquito *Aedes aegypti* é imediata, visto que ele é o principal vetor de doenças como a dengue, zika e chikungunya. O presente estudo busca identificar maneiras de aprimorar as estratégias de combate ao mosquito transmissor da dengue, com o objetivo de otimizar o tempo de fiscalização e alcançar resultados mais eficientes no controle dessa praga.

A motivação para a pesquisa surge do aumento significativo nos casos de dengue no município de Pinhalzinho, o que evidencia a urgência de adaptar as estratégias de combate às particularidades de cada município para aumentar a eficácia dos esforços de controle do mosquito *Aedes aegypti*.

A contribuição esperada para a comunidade inclui a inspiração para que outros municípios adotem estratégias semelhantes, o que poderá resultar na redução dos casos de dengue em toda a região, proporcionando maior tranquilidade à sociedade.

Além disso, há um grande potencial de replicação das estratégias desenvolvidas, uma vez que o objetivo é criar um modelo que possa ser implementado em outras localidades, contribuindo para a construção de comunidades mais saudáveis e seguras em todo o país.

1.4 PROBLEMA

A atual metodologia, refere-se ao combate da proliferação do mosquito *Aedes aegypti*. Que enfrenta diversas dificuldades, incluindo a ineficácia na gestão das visitas dos agentes de vigilância, a falta de um sistema eficiente para monitoramento dos focos de dengue e a comunicação ineficaz entre a comunidade e os órgãos de saúde. Essas lacunas resultam em respostas tardias sobre o controle dos criadouros, aumentando os riscos de surtos de dengue. Portanto, há uma necessidade urgente de desenvolver um sistema tecnológico que otimize o processo de controle e prevenção da dengue. Proporcionando maior eficiência e precisão nas ações de combate ao mosquito. Assim, tem-se o seguinte problema de pesquisa: Como um

software pode otimizar o controle da proliferação do *Aedes aegypti*, impactando diretamente na prevenção da dengue e na saúde pública municipal?

1.5 OBJETIVOS DA PESQUISA

Neste tópico, delinaremos os objetivos da pesquisa, os quais são fundamentais para orientar o desenvolvimento e a documentação do protótipo de sistema proposto. O escopo geral da pesquisa visa abordar as necessidades de gestão e controle de visitas e denúncias sobre pontos de criadouros do mosquito *Aedes aegypti*, considerando tanto aspectos históricos quanto às repercussões causadas pela endemia.

1.5.1 Objetivo geral

Realizar o desenvolvimento e documentação de um software de gestão para que o combate a proliferação de mosquitos *Aedes aegypti* seja mais eficaz e controlado, tendo como base o processo efetuado pelo órgão responsável pelo combate a dengue, e toda a comunidade do município de Pinhalzinho, Santa Catarina.

1.5.2 Objetivos específicos

Estudar conceitos referentes ao ciclo de proliferação de mosquitos *Aedes aegypti*, e também sobre as necessidades da comunidade com o combate do mesmo;

Identificar modelos de gestão e desenvolver sistema para controle de fiscalizações, denúncias e focos, diminuindo o trabalho manual da secretaria, otimizar o tempo de trabalho do agente e melhorar a confiança entre a comunidade e o serviço público;

Realizar um levantamento dos principais aspectos relacionados aos procedimentos efetuados pelo agente de vigilância da dengue.

2 REFERENCIAL TEÓRICO

Para a realização do referencial teórico foram utilizadas referências de autores que abordam sobre o mosquito da dengue (*Aedes aegypti*), o trabalho dos agentes de vigilância no combate à dengue, além de temas relacionados que possam auxiliar na elaboração desse projeto. Os principais tópicos levantados foram o combate contra a proliferação do mosquito, o trabalho dos órgãos públicos responsáveis pela vigilância epidemiológica e a participação da sociedade no combate do mosquito da dengue.

2.1 ORIGEM DA PALAVRA AEDES AEGYPTI

Segundo dicionário a origem da palavra “*Aedes*” que em latim significa “casa”, “prédio” e a palavra “*Aegypti*” que em latim significa “egito”, portanto *Aedes aegypti* significa “casa do Egito”, que na verdade em Latim a sua grafia correta é “*Ædes ægypti*”, esse nome se deve a abundância em que o inseto se encontra no país. (UFMA et. al, 2021).

2.2 SURGIMENTO DO AEDES AEGYPTI

O mosquito transmissor da dengue, originário do Egito na África, tem se disseminado pelas regiões tropicais e subtropicais do mundo desde o século 16, durante as Grandes Navegações. Acredita-se que o vetor tenha se deslocado para o Novo Mundo durante o período colonial através de navios que transportavam escravos. Foi descrito cientificamente pela primeira vez em 1762, sob o nome de *Culex aegypti*. O nome definitivo, *Aedes aegypti*, foi estabelecido em 1818 após a definição do gênero *Aedes*. (Instituto Oswaldo Cruz, 2011)

Relatórios da Organização Pan-Americana da Saúde (OPAS) indicam que a primeira epidemia de dengue nas Américas ocorreu no Peru, no início do século 19, com surtos também no Caribe, Estados Unidos, Colômbia e Venezuela. (Instituto Oswaldo Cruz, 2011).

2.2.1 Primeiros casos de Dengue no Brasil

Acredita-se que o mosquito *Aedes aegypti* tenha sido introduzido no Brasil através de transportes marítimos, ainda por volta do final do século XVIII, em embarcações que transportavam escravos para o continente americano (chamados de navios negreiros). (Instituto René Rachou Fiocruz Minas, 2024).

Os primeiros casos registrados de dengue ocorreram em Curitiba (PR) no final do século XIX e em Niterói (RJ) no início do século XX. Inicialmente, a preocupação com o mosquito *Aedes aegypti* estava relacionada à transmissão da febre amarela. Em 1955, o Brasil conseguiu erradicar o *Aedes aegypti* com medidas de controle voltadas para a febre amarela. No entanto, na década de 1960, essas medidas foram relaxadas, o que resultou na reintrodução do mosquito no país (Instituto Oswaldo Cruz, 2011).

Segundo o Ministério da Saúde (s.d), houve relatos de epidemias de dengue no início do século XX, com registros em São Paulo, SP e Niterói, RJ. A primeira epidemia documentada ocorreu em Boa Vista, Roraima, no ano de 1982, a dengue reapareceu de forma epidêmica em 3 estados (Rio de Janeiro, Ceará e Alagoas) em meados de 1986, sendo que a maior epidemia ocorreu no Rio de Janeiro, atingindo mais de um milhão de pessoas. Neste ano e nos anos seguintes (até 1989) o sorotipo Den 1 foi o responsável por epidemias e/ou surtos da doença nos estados do Rio de Janeiro, São Paulo, Minas Gerais, Ceará, Alagoas, Bahia e Pernambuco.

A disseminação dos vírus DENV-2 e DENV-3, que causam a dengue, em diversos países desde 1963. No Brasil, o mosquito *Aedes aegypti*, que transmite a dengue, voltou a se espalhar nas grandes cidades na década de 60. Em 1967, o pesquisador Leônidas Deane encontrou o *Aedes aegypti* em Belém, possivelmente trazido em pneus contrabandeados do Caribe. Até 1974, o mosquito já havia infestado Salvador e, no final da década de 70, alcançou novamente o Rio de Janeiro. (Instituto René Rachou Fiocruz Minas, 2024).

2.2.2 Primeiros casos de Dengue em Pinhalzinho e no estado de Santa Catarina

Segundo Ministério da Saúde (2017), o estado de Santa Catarina foi o último estado brasileiro a registrar casos de dengue, no ano de 2011. Logo após, em 2015, foi registrado a primeira epidemia da dengue no estado catarinense, mais especificamente na cidade de Itajaí.

A situação epidemiológica de Santa Catarina em relação à dengue tem se transformado anualmente. Os primeiros casos locais de dengue foram notificados em 2011 (Joinville e São João do Oeste), 2012 (Joinville), 2013 (Itapema e Chapecó) e 2014 (Itajaí). Em 2015, foi registrada a primeira epidemia de dengue no estado, no município de Itajaí, além de casos em Chapecó, Itapema, Joinville e São Miguel do Oeste. (DIVE, 2022).

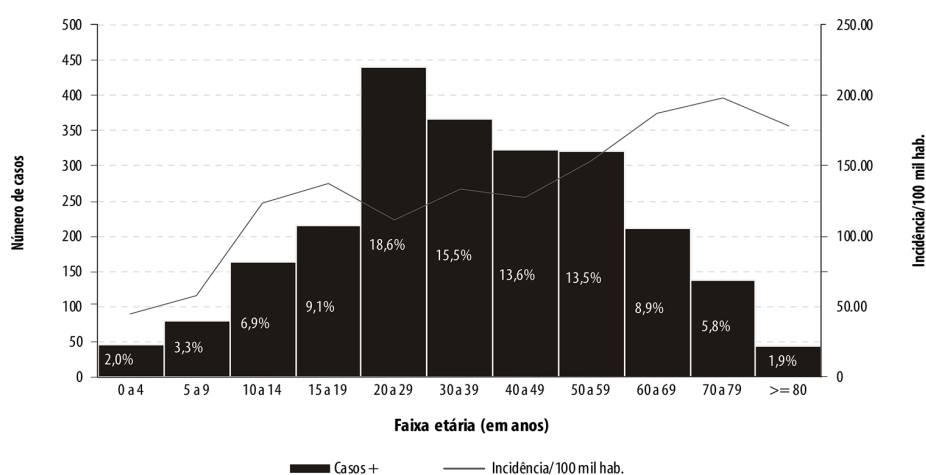
Um estudo ecológico foi conduzido em Pinhalzinho, no oeste de Santa Catarina, Brasil. Em 2015, a população era de 18.700 pessoas, com 84% vivendo na área urbana. O município tem clima subtropical, um Índice de Desenvolvimento Humano (IDH) de 0,783 e

94,9% dos lares possuem esgotamento sanitário adequado, conforme o Censo Demográfico de 2010. (IBGE, 2017).

Dados sobre notificações de casos de dengue e hospitalizações relacionadas entre 2015 e 2016 foram coletados junto à Secretaria de Saúde de Pinhalzinho, utilizando o Sistema de Informação de Agravos de Notificação (Sinan) e o sistema TABNET do Datasus. As taxas de incidência foram calculadas por faixas etárias e sexo, para cada 100 mil habitantes. Para verificar associações entre variáveis nominais, foi utilizado o teste qui-quadrado, com nível de significância de 5%. (Secretaria da Saúde de Pinhalzinho, 2015).

A maioria dos casos de dengue em Pinhalzinho, entre 2015 e 2016, ocorreu em mulheres, com alta incidência. A idade média dos casos foi de 38 anos, com mais ocorrências entre 20 e 29 anos e em faixas etárias mais avançadas (figura 1). (Secretaria da Saúde de Pinhalzinho, 2015).

Figura 01: Distribuição do número de casos e da incidência de dengue por faixa etária, Pinhalzinho, Santa Catarina, 2015-2016.

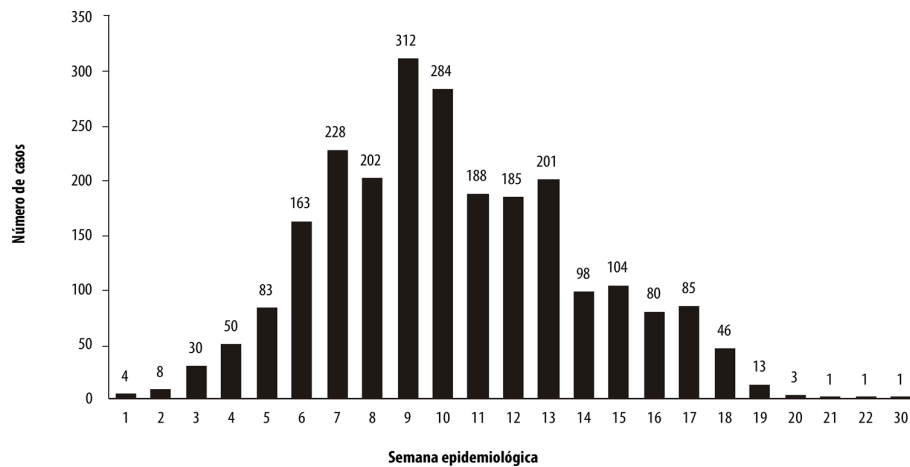


Fonte: (LUTINSKI JA, et. al, 2020).

Em 2015, foram confirmados quatro casos de dengue, um em novembro (importado) e três no mês de dezembro (autóctones).

Segundo Junir Antonio Lutinski (2020), no ano de 2016, houve um total de 2.370 casos de dengue confirmados localmente. No mesmo período, ocorreram 130 hospitalizações devido à doença, com pico em fevereiro, março e abril, respectivamente (figura 2).

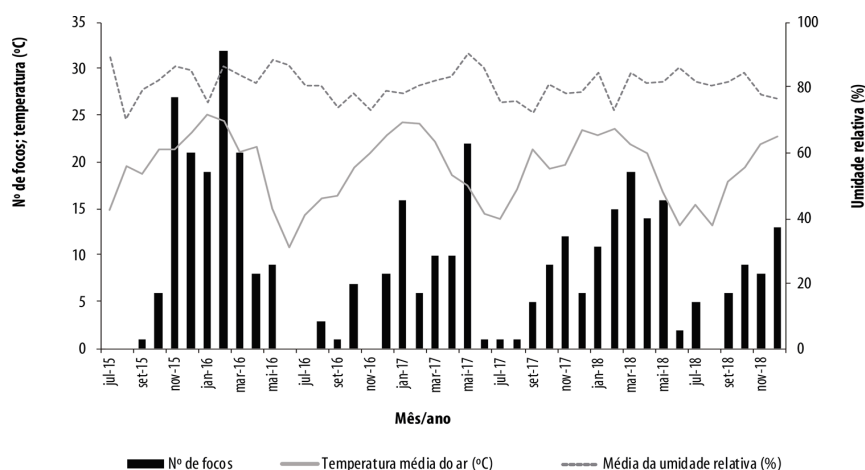
Figura 02: Distribuição do número de casos confirmados de dengue por semana epidemiológica da notificação, Pinhalzinho, Santa Catarina, 2016.



Fonte: (LUTINSKI JA, et. al, 2020).

Durante o período estudado, a temperatura mínima mensal variou de $-3,58$ a $16,04^{\circ}\text{C}$, a máxima de $26,65$ a $36,48^{\circ}\text{C}$, e a média de $10,88$ a $25,17^{\circ}\text{C}$. A precipitação mensal variou de $9,6$ a 428mm e a umidade relativa do ar de $70,22\%$ a $90,52\%$. A umidade relativa e a temperatura média do ar tiveram uma associação positiva e significativa com a infestação mensal pelo mosquito *Aedes aegypti* (figura 3). (LUTINSKI JA, et. al, 2020).

Figura 03: Distribuição mensal do número de focos de *Aedes aegypti* em comparação com a temperatura média mensal e a umidade relativa do ar, Pinhalzinho, Santa Catarina, 2015-2018.



Fonte: (LUTINSKI JA, et. al, 2020).

Diante disto é importante enfatizar que o aumento da incidência e disseminação da dengue é influenciado por múltiplos fatores sociais, ambientais e climáticos que, juntos,

podem gerar um cenário epidêmico.

2.3 COMBATE CONTRA A PROLIFERAÇÃO DO MOSQUITO DA DENGUE

Para evitar a dengue, é importante impedir que o mosquito *Aedes aegypti* se reproduza. Isso significa jogar fora qualquer água parada onde eles possam botar ovos, como em vasos de plantas, baldes, pneus velhos, ou até mesmo em tampinhas de garrafa. Manter as áreas limpas e sem acúmulo de água ajuda a diminuir o risco de ter mosquitos por perto. (Ministério da Saúde; 2014)

Desta forma pode ser evitada a proliferação do mosquito que transmite a dengue, podemos tomar algumas medidas fáceis, como trocar a água dos pratinhos dos vasos por areia, manter as caixas d'água sempre fechadas, cobrir piscinas e remover qualquer objeto que possa acumular água parada. Essas ações simples ajudam a evitar que os mosquitos se reproduzam e diminuam o risco de contrair a doença. (Ministério da Saúde; 2014)

Especialistas recomendam a limpeza regular das calhas para evitar obstruções que impeçam o fluxo de água, além de práticas como manter a lixeira fechada, colocar o lixo em sacos plásticos e remover entulhos do quintal. Também destacam a importância de trocar frequentemente a água dos recipientes dos animais de estimação. (GOV. MS; 2024)

Deve-se ressaltar a importância de medidas simples, porém eficazes, no combate à dengue. Ao eliminar objetos que possam acumular água, como copinhos plásticos e tampas de refrigerante, e cobrir piscinas sem uso, estamos reduzindo os locais propícios para a reprodução dos mosquitos transmissores da doença. Assim como a orientação de tampar os ralos é outra medida preventiva fundamental. São ações que, quando adotadas de forma consciente e consistente pela comunidade, contribuem significativamente para a redução dos casos de dengue e para a promoção da saúde pública. (GOV. MS; 2024).

Também é de grande importância manter a vigilância mesmo durante as obras em casa, para evitar que equipamentos como lonas, carrinhos de mão e betoneiras se tornem criadouros de mosquitos transmissores da dengue. Além disso, ressalta a necessidade de realizar a limpeza regular da bandeja externa da geladeira e da bandeja coletora de água do ar-condicionado, locais que podem acumular água e servir de ambiente propício para a proliferação dos mosquitos. São medidas simples, porém essenciais, para prevenir a disseminação da dengue e proteger a saúde da família. (GOV. MS; 2024).

O uso de roupas que cubram a pele durante o dia, quando os mosquitos estão mais ativos, pode oferecer proteção contra picadas, especialmente durante surtos. Também

repelentes e inseticidas, desde que usados conforme as instruções do rótulo, são recomendados. Mosquiteiros são eficazes para proteger pessoas que dormem durante o dia, como bebês, pessoas acamadas e trabalhadores noturnos. Essas medidas são sugeridas pelo Ministério da Saúde para prevenir picadas de mosquitos e possíveis doenças transmitidas por eles. (Ministério da Saúde; 2014).

2.3.1 Trabalho dos órgãos públicos e agentes de combate a endemias

Segundo a Secretaria de Vigilância em Saúde (2008), Santa Catarina realiza atividades de campo, como visitas a armadilhas, pontos estratégicos e domicílios, focando na vigilância e controle do vetor. Com a dispersão do vetor nos municípios, as ações se transformam em controle, exigindo uma reestruturação organizacional do programa em termos de pessoal, material e atividades, de acordo com a realidade local.

A prevenção da dengue é mais eficaz quando há uma vigilância sistemática dos vetores, permitindo a detecção precoce de sua presença no município e possibilitando ações imediatas para eliminá-los e controlá-los. (DIVE, 2022).

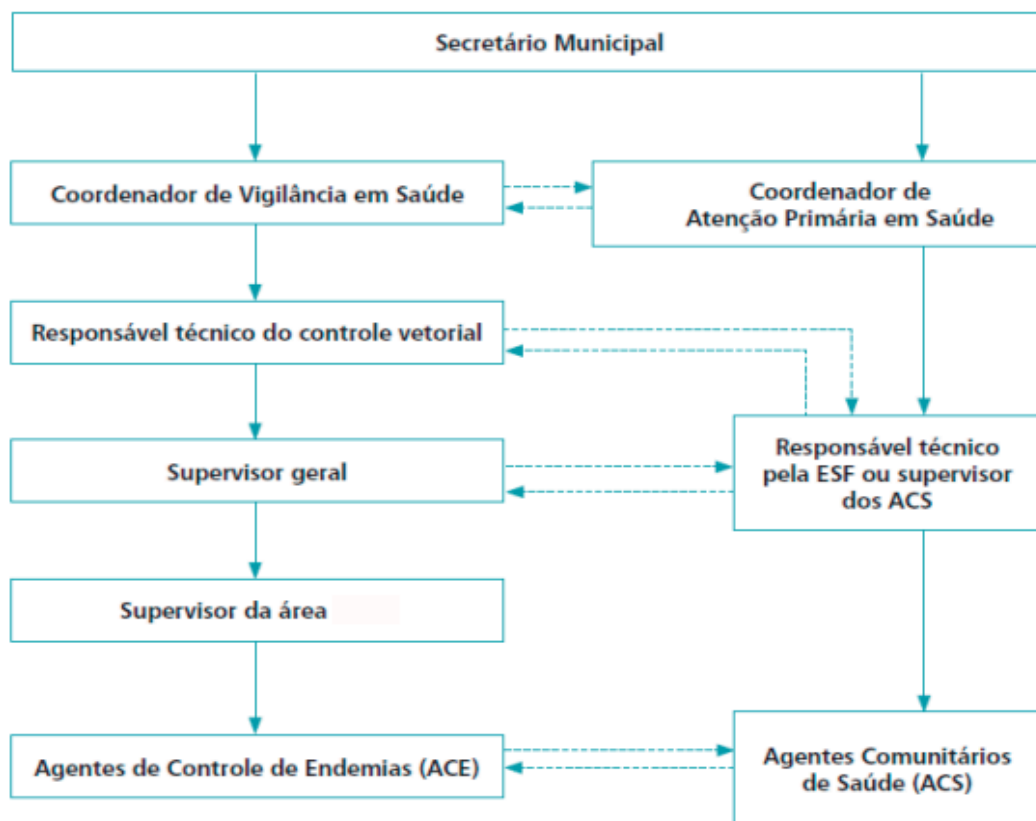
O governo contribui para a prevenção da dengue por meio de vigilância vetorial sistemática, colaboração intersetorial, fornecimento de água encanada em quantidade e qualidade adequadas, coleta regular e destinação apropriada do lixo, além de informar a população sobre a ocorrência da dengue e seus vetores. (DIVE, 2022).

O Agente de Controle de Endemias (ACE) é crucial no combate à dengue, atuando na eliminação de criadouros de difícil acesso, como caixas d'água, e utilizando larvicidas biológicos ou químicos. Ele também pode ser conhecido como Agente de Vigilância Ambiental ou de Zoonoses. (Secretaria de Vigilância em Saúde, 2008).

O ACS e o ACE compartilham a responsabilidade pelo controle da dengue, trabalhando de forma integrada em ações como educação em saúde, mobilização comunitária e identificação de criadouros. O ACS se concentra no acompanhamento das pessoas com dengue, enquanto o ACE é responsável pela eliminação de criadouros de difícil acesso e pelo uso de larvicidas. (Secretaria de Vigilância em Saúde, 2008).

Um esquema hierárquico do Programa de Controle da Dengue nos municípios, destacando a necessidade de comunicação entre os setores de Controle de Vetores, Vigilância Epidemiológica e Atenção Básica (figura 4).

Figura 04: Desenho esquemático da estrutura hierárquica da área de controle de vetores.



Fonte: (DIVE, 2022).

O agente de campo desempenha um papel crucial na vigilância e controle do *Aedes aegypti*. Sua responsabilidade principal é executar todas as atividades em campo, independentemente da situação do município. Isso inclui detectar precocemente focos do mosquito, eliminar criadouros potenciais e educar a comunidade por meio de ações educativas. (DIVE, 2022).

Os agentes colaboram com a comunidade para gerenciar o vetor do *Aedes aegypti*. Eles identificam, removem ou tratam de forma apropriada os locais onde o mosquito possa depositar seus ovos, incluindo reservatórios naturais ou artificiais de água. (FRANÇA, 2017).

2.3.2 O controle do criadouro

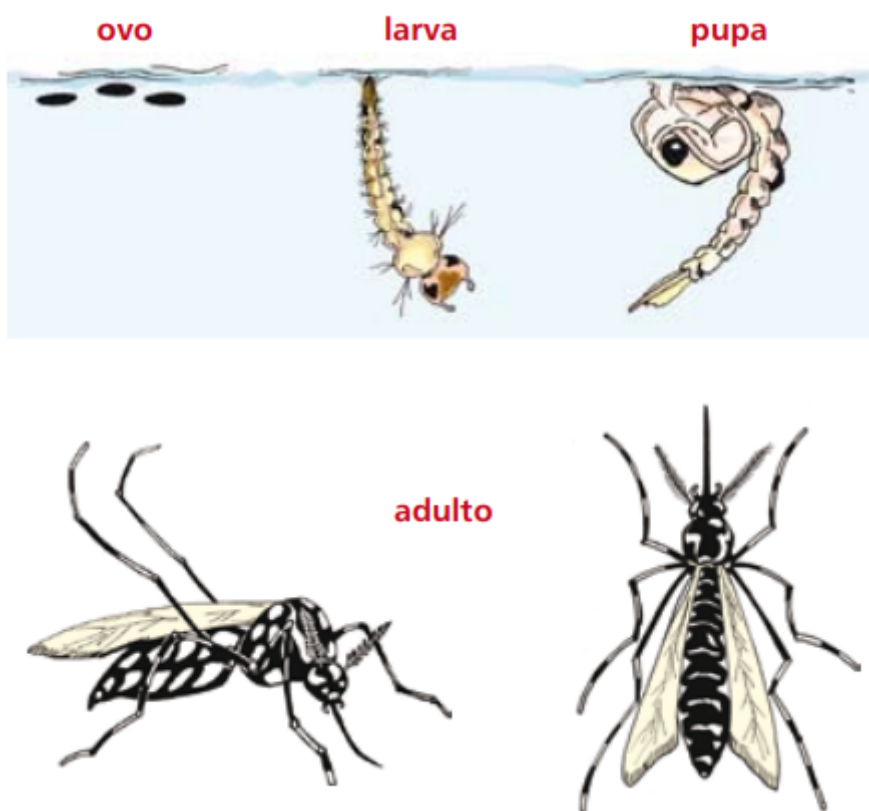
O controle eficaz da dengue requer o envolvimento de todos os profissionais de saúde, gestores e a comunidade em geral. Parcerias entre diferentes setores da administração municipal, como limpeza urbana, saneamento e meio ambiente, são essenciais para combater a proliferação do mosquito *Aedes aegypti*. (Secretaria de Vigilância em Saúde, 2008).

É fundamental conscientizar a população sobre a importância de eliminar recipientes que possam se tornar criadouros do mosquito, como garrafas, latas e pneus, encontrados comumente em áreas urbanas. Segundo a Secretaria de Vigilância (2008), as ações de controle devem incentivar a participação ativa de cada morador na identificação e eliminação de potenciais locais de reprodução do mosquito.

Os ovos do mosquito são de extrema resistência e podem ser suportados em um recipiente, sem ter entrado em contato com a água, entre 300 e 400 dias. Quando expostos à água novamente, podem se desenvolver rapidamente, passando pelas fases de pupa e larva até atingirem a fase adulta em apenas 2 a 3 dias. (GOV MS, 2024).

Seu ciclo apresenta quatro fases: ovo, larva, pupa e adulto, ilustradas abaixo, em tamanho ampliado (figura 5).

Figura 05: Ciclo de vida do mosquito *Aedes aegypti*.



Fonte: (Secretaria de Vigilância em Saúde, 2008)

Uma atividade crucial na prevenção da dengue é o Levantamento Rápido de Índices de Infestação do *Aedes aegypti* (LIRAA), que é realizado de forma amostral, não exigindo a visita a todas as casas. Os resultados fornecem índices de infestação predial, classificados em

condições satisfatórias (inferiores a 1%), alerta (de 1% a 3,9%) e risco de surto de dengue (superior a 4%). Esse levantamento permite identificar áreas prioritárias para intervenção, seja em locais de abastecimento de água, depósitos domiciliares, lixo, entre outros. (DIVE, 2022).

Os indicadores passíveis de serem utilizados para medir o grau de infestação com este levantamento são os descritos abaixo (figura 6).

Figura 06: Classificação dos índices de infestação por Aedes aegypti.

| IIP (%) | Classificação |
|---------|---------------|
| < 1 | Satisfatório |
| 1 - 3,9 | Alerta |
| > 3,9 | Risco |

Fonte: (DIVE, 2022).

Em municípios infestados, é importante estabelecer a Sala de Situação, onde os órgãos municipais discutem e planejam ações de combate ao *Aedes aegypti* com base em diagnósticos locais.

As atividades de controle vetorial são essenciais para prevenir a proliferação do mosquito *Aedes aegypti*, sendo ainda hoje a principal forma de evitar casos dessas enfermidades. Existem diversas técnicas disponíveis em saúde pública para o controle de vetores, que podem ser classificadas em controle mecânico/manejo ambiental, controle biológico, controle legal, controle químico e controle integrado de vetores. (DIVE, 2022).

Dada a relevância das ações coordenadas, esta pesquisa visa investigar a interação entre os Agentes Comunitários de Saúde (ACS) e os Agentes de Combate a Endemias (ACE) no dia-a-dia das atividades de prevenção e controle do *Aedes aegypti* na Atenção Primária à Saúde, além de analisar as táticas de resposta implementadas em todas as localidades do país.

2.3.3 Participação da sociedade no combate a Dengue

A participação ativa da comunidade é crucial na prevenção da dengue, visto que o engajamento social pode aumentar a compreensão sobre a doença e suas medidas preventivas, resultando na redução dos casos. A enfermagem desempenha um papel vital ao

educar a população sobre essas práticas, não só beneficiando a saúde pública, mas também reduzindo o impacto econômico associado. (FEITOSA, 2012).

Reconhecendo a importância de abordar essa questão, especialmente diante do elevado número de casos de dengue, é fundamental buscar parcerias com a comunidade local para desenvolver estratégias eficazes de combate à doença. (FEITOSA, 2012).

O envolvimento da comunidade na gestão ambiental e saneamento domiciliar é fundamental e inclui a eliminação adequada de recipientes inservíveis, como latas, materiais descartáveis, cascas de ovos e tampas de garrafas. Vale ressaltar que também é importante manter vedados caixas d'água, poços, cisternas, tanques e outros reservatórios de água, para evitar a reprodução do mosquito transmissor da doença. (DIVE, 2022).

Medidas para controlar a disseminação do *Aedes aegypti*, que devem ser seguidas pela população em residências e estabelecimentos comerciais, junto com as penalidades possíveis por não obedecer às regras, estão definidas na Lei 18.024, datada de 26 de outubro de 2020. Todas as leis anteriores relacionadas a esse assunto foram revogadas. (DIVE, 2022).

A eficácia na prevenção da transmissão da doença cresce significativamente quando são realizadas medidas de controle larvário, como a eliminação e tratamento de focos, além da intensificação das visitas domiciliares e mutirões de limpeza, com a participação ativa da comunidade (DIVE, 2022).

O protagonismo da comunidade na eliminação do mosquito é essencial, pois estudos apontam que cerca de 90% dos criadouros estão no interior dos domicílios. As instituições governamentais têm buscado organizar ações integradas de saúde, educação, comunicação e mobilização social, na Atenção Primária à Saúde, em especial a Estratégia Saúde da Família (ESF). (GOMES et al., 2015)

Há fatores que intervêm na aderência da população às práticas preventivas que têm relação direta com a falta de comunicação entre o serviço de saúde e a comunidade. Se faz necessário prover ambientes que estimulem a discussão nas comunidades sobre as questões sanitárias relevantes e de forma permanente (SILVA et al., 2011).

Para combater o *Aedes* é imperativo repensar a configuração de intervenções no meio ambiente, pois sem a implantação de sistemas públicos de abastecimento de água para consumo doméstico de forma regular, esgotamento sanitário, o correto destino dos resíduos sólidos urbanos e sem redução da violência em muitas comunidades urbanas não há como reduzir os índices de infestação dos mosquitos (PERES, 2016).

3 PROCEDIMENTOS METODOLÓGICOS

Procedimentos metodológicos podem ser entendidos como um precursor para alcançar um fim ou pelo qual se atinge um objetivo, ou seja, é o caminho realizado quando focado na produção de conhecimentos (PEREIRA, 2016).

Na abordagem metodológica, foram conduzidas várias pesquisas em documentos, artigos e portais de informações relacionados à saúde e também ao combate contra o mosquito *Aedes aegypti*, visando compreender a interação dos órgãos responsáveis pela fiscalização dos focos do mosquito e também sobre a interação da comunidade a favor do combate a dengue, especialmente em períodos de surto epidêmico. O objetivo era analisar como isso impactou a gestão das secretarias de controle da dengue no município de Pinhalzinho.

3.1 TIPO DE ESTUDO

A abordagem desta pesquisa é considerada mista, pois incorpora tanto métodos qualitativos quanto quantitativos. Essa abordagem permite a utilização de procedimentos de ambas as naturezas, sendo conhecida como pesquisa de métodos mistos, assim, terá questionário e ainda a realização de entrevista (GIL, 2019).

3.1.1 Quanto aos objetivos

A Pesquisa pode ser caracterizada como descritiva, pois busca referenciar partes importantes do estudo. A pesquisa, caracterizada como descritiva, busca proporcionar diferentes visões sobre algo já conhecido (LOZADA; NUNES, 2019).

3.1.2 Quanto aos procedimentos

Este trabalho apresentará um estudo de caso, um estudo de caso envolve a realização de uma pesquisa investigativa focada em um evento ou situação específica, com resultados que podem ser aplicados a outras situações semelhantes. Este método é projetado para oferecer uma compreensão detalhada e aprofundada das características de um determinado assunto ou problema (NASCIMENTO, 2016).

3.2 UNIVERSO DA PESQUISA

Os levantamentos de elementos para realização de uma pesquisa abrangem um universo tão grande, que se torna impossível considerá-los em sua totalidade. Para isso se faz necessário trabalhar com uma população, que é o órgão público responsável pelo combate do mosquito *Aedes aegypti*, para fins de estudo e uma amostra, que seriam os agentes de fiscalização, ou seja, com uma pequena parte dos elementos que compõem a pesquisa (GIL, 2019).

Esta pesquisa decorre da necessidade de compreender por que e como os surtos de dengue representam desafios significativos e promovem mudanças na área da saúde, tanto globalmente quanto no Brasil.

3.2 COLETA E TRATAMENTO DE DADOS

A pesquisa documental exerce um papel de grande importância nos processos observacionais, na condição da descoberta, e faz com que o pesquisador tenha um contato direto com a realidade (MARCONIS;LAKATOS, 2017).

Já a utilização do questionário, é constituído por uma série de perguntas, que devem ser respondidas por escrito, além da entrevista semiestruturada com pessoas selecionadas, dentre as principais vantagens estão a economia de tempo, a eficiência na coleta de um grande número de dados, a possibilidade de atingir um número maior de pessoas em uma área geográfica mais ampla (TOMÉ et al., 2019).

Ao compreender esses conceitos e sua relação com a situação atual e os objetivos mencionados na pesquisa, podemos identificar oportunidades para aprimorar a prestação de serviços aos pacientes e melhorar a eficiência do controle interno. Uma proposta que surgiu é o desenvolvimento de um protótipo de software, para gerenciar rotinas de fiscalização ao combate do *Aedes aegypti*.

4 ANÁLISE DOS REQUISITOS

Nesta etapa abordaremos os passos necessários para o desenvolvimento bem sucedido do protótipo, estabelecendo uma base consistente de requisitos e compreendendo as necessidades elencadas.

4.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais são especificações das funcionalidades e capacidades que um sistema, software ou produto deve oferecer para atender às necessidades e expectativas dos usuários, clientes ou stakeholders, visando a satisfação de maneira ideal. Com este embasamento, os requisitos funcionais identificados do protótipo são:

- Cadastros de usuários:
 - Os novos usuários só podem ser cadastrados através de um usuário que possua a permissão de efetuar novos cadastros.
 - Validação dos dados do usuário para que ao efetuar o cadastro, a integridade do sistema seja mantida.
- Controle de fiscalizações:
 - Usuários podem efetuar novos cadastros de fiscalizações de rotina, sobre o controle dos focos de dengue.
 - Consultas de fiscalizações.
 - Relatório de fiscalizações.
 - Possibilidade de edição de fiscalização.
- Controle de denúncias:
 - Registro de denúncia sobre um possível criadouro do mosquito *Aedes aegypti*.
 - Indicativo através de localização apresentado no mapa.
 - Possibilidade de consultar denúncia.
 - Possibilidade de inclusão de imagens ou vídeos sobre a denúncia efetuada.
- Controle de estatísticas:
 - Possibilidade de controlar números sobre os casos de dengue, casos ativos, mortes confirmadas, casos confirmados e casos monitorados.

- Controle de focos:
 - Possibilidade de apontar novos focos em um mapa, para que seja demonstrado à população.
 - Possibilidade de excluir focos

4.2 REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais são critérios que descrevem como um sistema deve operar, ao invés de especificar comportamentos ou funções específicas. Eles são essenciais para garantir a qualidade do sistema e sua capacidade de atender às necessidades dos usuários e do negócio. Os requisitos não funcionais do software são:

- **Disponibilidade:** Como um requisito não funcional, refere-se à capacidade de um sistema estar operacional e acessível quando necessário. Este é um aspecto crítico, especialmente para sistemas web, onde a interrupção do serviço pode resultar em perda de receita, insatisfação do usuário e danos à reputação.
- **Desempenho:** Refere-se à velocidade e eficiência com que o sistema responde a solicitações. Isso pode incluir tempos de resposta, taxa de processamento e capacidade de lidar com um número específico de usuários simultâneos.
- **Segurança:** Envolve a proteção dos dados e a integridade do sistema contra acessos não autorizados e ataques cibernéticos. Requisitos de segurança incluem autenticação, autorização, criptografia e auditoria.
- **Manutenibilidade:** Refere-se à facilidade com que o sistema pode ser atualizado ou modificado. Um sistema manutenível permite a correção de bugs, a implementação de novas funcionalidades e a adaptação a mudanças nas necessidades do negócio de forma eficiente.
- **Usabilidade:** Trata-se da facilidade com que os usuários podem interagir com o sistema. Um sistema com alta usabilidade proporciona uma experiência positiva, facilitando a adoção e o uso.

4.3 UML

A UML (Unified Modeling Language) é uma linguagem padronizada que define uma série de artefatos para auxiliar na modelagem e documentação de sistemas orientados a objetos. Com ela, podemos representar graficamente os componentes, interações e estrutura de um sistema, facilitando tanto o desenvolvimento quanto a comunicação entre os membros da equipe (DEV MEDIA, 2018).

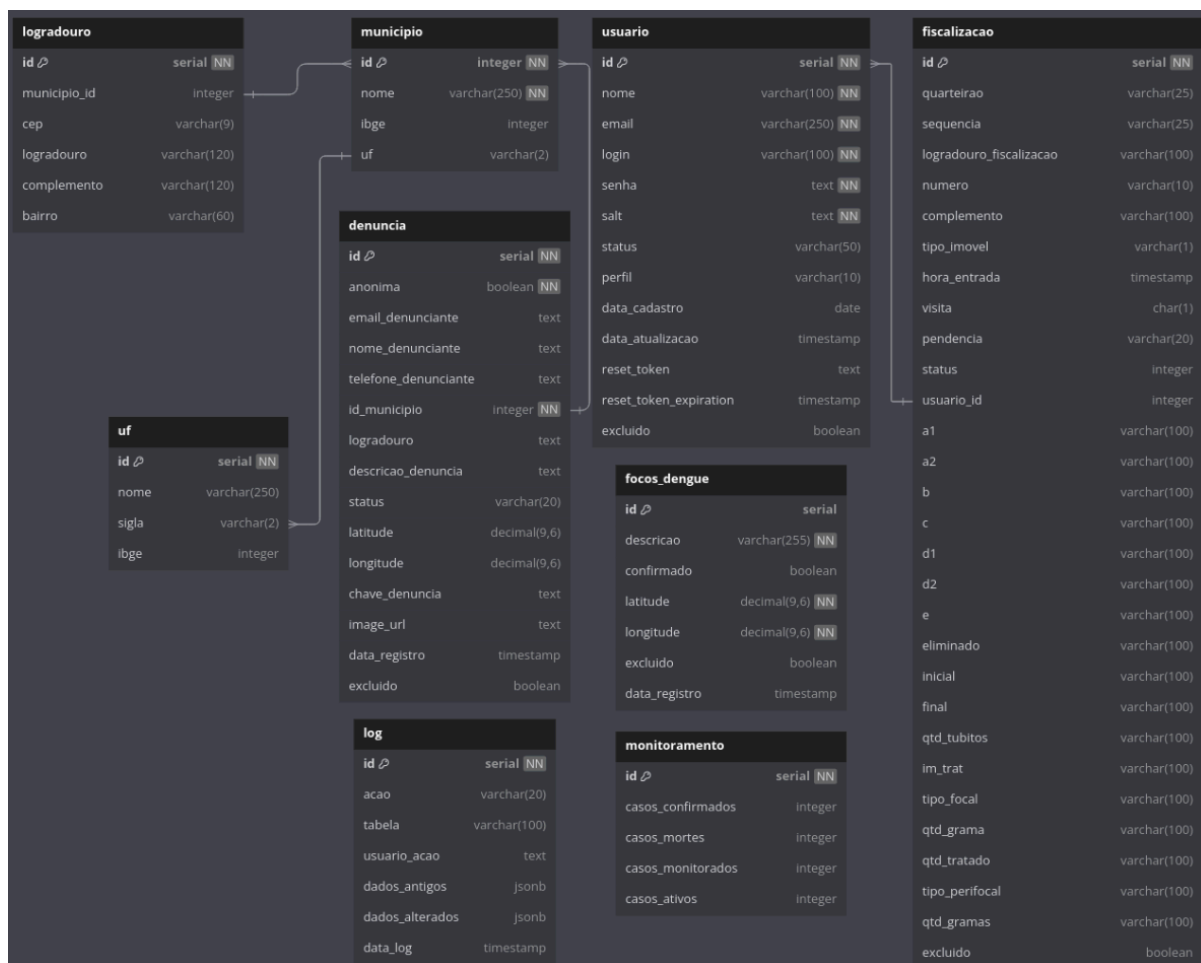
O desenvolvimento de software envolve várias etapas que buscam garantir a qualidade e reduzir a ocorrência de erros. Uma das partes fundamentais desse processo é a documentação, que organiza e detalha a estrutura do projeto. A UML (Unified Modeling Language) é uma linguagem de notação amplamente utilizada para modelar e documentar sistemas orientados a objetos, permitindo que as equipes de desenvolvimento representem graficamente componentes, interações e transformações ao longo do projeto (PEDRO, 2023).

Utilizando elementos visuais como retângulos, setas e linhas, a UML facilita a criação de diagramas que fornecem uma visão clara e padronizada do software em desenvolvimento. Isso ajuda a equipe a visualizar as áreas do sistema e suas interações, facilitando a comunicação e reduzindo falhas nas fases de implementação. Como uma linguagem de notação universal, a UML é objetiva e compreensível, tornando-a uma ferramenta valiosa para documentar e planejar as fases do projeto (PEDRO, 2023).

4.3.1 Diagrama de Entidade e Relacionamento

Diagrama de entidade relacionamento é um fluxograma que ilustra como entidades, pessoas, conceitos ou objetos se relacionam em relação a um sistema, tendo seu uso voltado a depuração de banco de dados relacionais, definindo assim a estrutura de um banco de dados (FRANK et al., 2021).

Figura 07: Diagrama de entidade e relacionamento do software.



Fonte: Elaborado pelo autor (2024).

Na figura 07, está representado o Diagrama de Entidade e Relacionamento do software, que demonstra toda a estrutura do banco de dados e também suas ligações entre as classes, proporcionando uma melhor visão de toda estrutura.

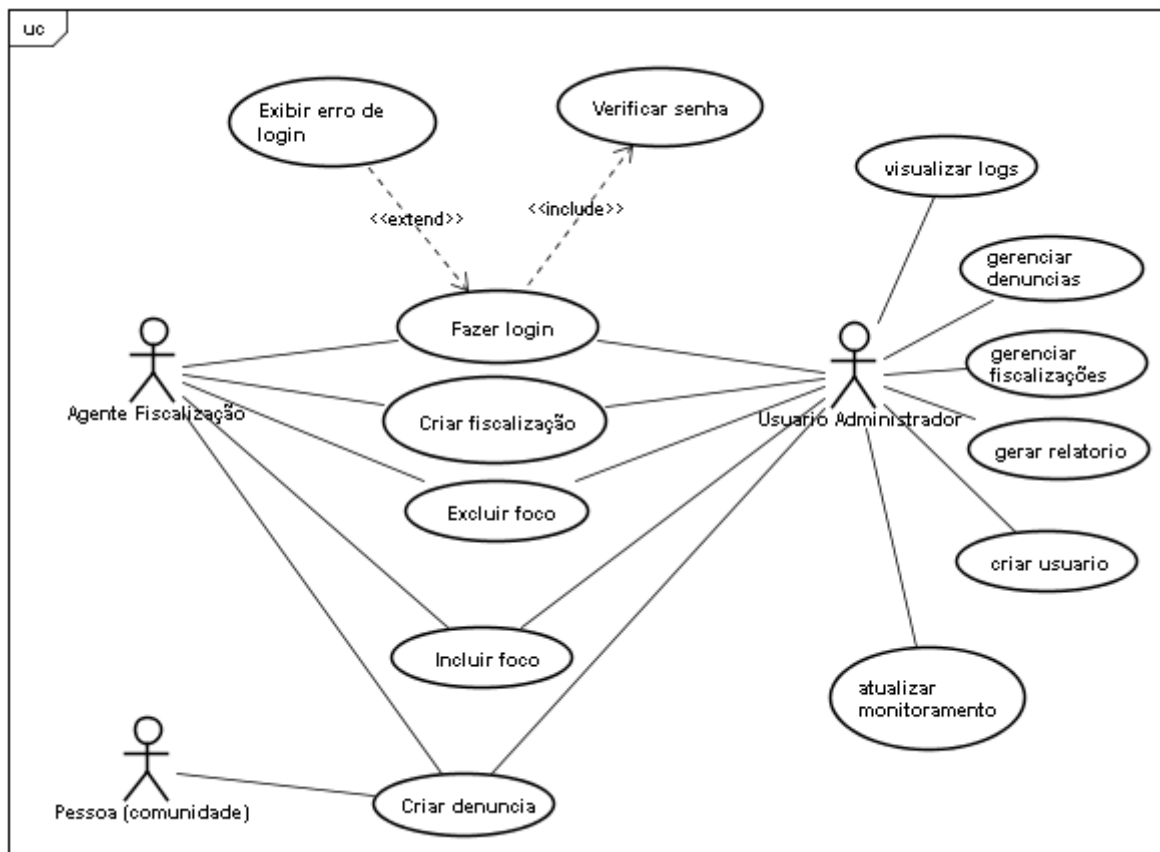
4.3.3 Diagrama de Caso de Uso

Um diagrama de caso de uso em UML mostra os usuários (atores) de um sistema e como eles interagem com ele. Utiliza símbolos e conectores específicos para ilustrar essas interações. Esse diagrama é útil para a equipe visualizar e discutir o funcionamento do sistema e as funções que ele oferece aos usuários (LUCIDCHART, 2023).

O diagrama de caso de uso fornece uma visão geral das relações entre atores, casos de uso e o sistema, sem detalhar a sequência de passos. Ele complementa uma descrição textual

de casos de uso, oferecendo uma perspectiva visual útil, mas sem aprofundar nos detalhes de execução (LUCIDCHART, 2023).

Figura 08: Diagrama de Caso de Uso do software.



Fonte: Elaborado pelo autor (2024).

Na figura 08, está representado o diagrama de caso de uso do software, onde existem os atores, Agente Fiscalização sendo os ACE'S, Usuário Administrador representando o usuário com controle geral do software, geralmente sendo o secretário responsável pelo combate a dengue e o ator Pessoa (comunidade), representando a população, que por sua vez pode efetuar as denúncias através do site.

5 FERRAMENTAS, SISTEMAS, TECNOLOGIAS E METODOLOGIAS

Neste segmento, iremos explorar as ferramentas, tecnologias, metodologias e também os sistemas empregados na elaboração do protótipo. Cada um desempenha um papel fundamental na concepção e implementação eficaz do projeto, colaborando para a funcionalidade, desempenho e êxito geral do produto em fase de desenvolvimento.

5.1 VISUAL STUDIO CODE

O Visual Studio Code (VS Code) é um editor de código-fonte desenvolvido pela Microsoft, que se destaca por sua leveza, flexibilidade e extensibilidade. Lançado em 2015, o VS Code rapidamente ganhou popularidade entre os desenvolvedores devido à sua interface intuitiva e suas poderosas funcionalidades de edição de texto e depuração de código (Microsoft; 2015).

Uma das principais características do VS Code é sua vasta coleção de extensões, que permitem aos usuários personalizar o ambiente de desenvolvimento de acordo com suas necessidades e preferências. Essas extensões abrangem uma ampla gama de linguagens de programação, frameworks e ferramentas, tornando o VS Code uma escolha versátil para desenvolvedores de diferentes áreas (Microsoft; 2015).

Além disso, o VS Code oferece integração nativa com controle de versão, facilitando o trabalho colaborativo em projetos de software. Os desenvolvedores podem se conectar a repositórios Git, SVN e outros sistemas de controle de versão diretamente do editor, simplificando o processo de gerenciamento de código-fonte (Microsoft; 2015).

Outro aspecto notável do Visual Studio Code é sua comunidade ativa e engajada. Os usuários podem contribuir com extensões, temas e sugestões de melhorias diretamente no GitHub, onde o código-fonte do VS Code é hospedado como um projeto de código aberto. Essa colaboração aberta tem impulsionado constantes atualizações e aprimoramentos no editor ao longo dos anos (Microsoft; 2015).

Em resumo, o Visual Studio Code é uma ferramenta poderosa e flexível para desenvolvimento de software, oferecendo uma experiência de edição de código rápida, eficiente e altamente personalizável. Sua ampla adoção e constante evolução o consolidam como uma escolha popular entre desenvolvedores de todo o mundo (Microsoft; 2015).

5.2 POSTMAN

O Postman é uma ferramenta que facilita a documentação, teste e execução de requisições em APIs. Com ele, é possível trabalhar de forma eficiente, construindo e armazenando solicitações para uso futuro, além de analisar as respostas recebidas das APIs. (VERSIANI, 2024).

Além de ser um aplicativo gratuito e fácil de aprender, com pouco tempo você já estará enviando seus primeiros requests (solicitações/requisições). No mais, trata-se de uma ferramenta com um amplo suporte para todas as APIs e Schemas. (VERSIANI, 2024).

O Postman oferece diversas vantagens, tais como suporte a diferentes tipos de chamadas de API (REST, SOAP, HTTP), compatibilidade com formatos de dados populares (OpenAPI, GraphQL, RAML), fácil acessibilidade através de login na conta, uso de coleções para organizar chamadas de API, capacidade de colaboração e compartilhamento de arquivos, criação de ambientes para evitar repetições de testes, elaboração de testes com pontos de verificação, automação de testes com Collection Runner ou Newman, depuração facilitada com o console do Postman e suporte à integração contínua, mantendo práticas de desenvolvimento. (VERSIANI, 2024).

5.3 GITHUB

O GitHub é uma plataforma de desenvolvimento colaborativo de software baseada na web que utiliza o sistema de controle de versão Git. Criado em 2008 por Chris Wanstrath, PJ Hyett e Tom Preston-Werner, o GitHub revolucionou a forma como os desenvolvedores trabalham em projetos de código aberto e privados, oferecendo ferramentas poderosas para gerenciar o ciclo de vida do desenvolvimento de software (GitHub, 2008).

Desde o seu lançamento, o GitHub cresceu exponencialmente, tornando-se o lar de milhões de repositórios de código-fonte e atraindo uma vasta comunidade de desenvolvedores de todo o mundo. A plataforma permite que os desenvolvedores colaborem em projetos, contribuindo com código, reportando problemas, propondo alterações e revisando o trabalho de seus colegas (GitHub, 2008).

Uma das características mais poderosas do GitHub é sua capacidade de facilitar o trabalho em equipe distribuída. Desenvolvedores de diferentes partes do mundo podem colaborar em projetos compartilhados, trabalhando em conjunto em tempo real ou de forma

assíncrona. Isso promove a inovação e permite que projetos de código aberto cresçam e evoluam de maneira rápida e eficiente (GitHub, 2008).

Em suma, o GitHub é um ecossistema fundamental para o desenvolvimento colaborativo de software, promovendo a inovação e a transparência na comunidade de desenvolvimento (GitHub, 2008).

5.4 DOCKER

Docker é uma plataforma de código aberto que utiliza a tecnologia de containers para simplificar o processo de desenvolvimento, implantação e execução de aplicativos. Com o Docker, os desenvolvedores podem empacotar seus aplicativos junto com suas dependências em um contêiner virtualizado, garantindo que o aplicativo seja executado de maneira consistente em qualquer ambiente (Merkel; 2014).

Esses contêineres são leves, portáteis e autossuficientes, o que significa que podem ser facilmente movidos entre diferentes ambientes, como máquinas locais, servidores de nuvem e data centers.. Além disso, os contêineres Docker são altamente escaláveis, permitindo que os aplicativos sejam dimensionados horizontalmente com facilidade, conforme necessário (Merkel; 2014).

Uma das principais vantagens do Docker é a sua eficiência no uso de recursos, pois os contêineres compartilham o kernel do sistema operacional host e apenas executam os processos necessários para o aplicativo, sem a sobrecarga de uma máquina virtual completa. Isso resulta em tempos de inicialização mais rápidos e uma pegada de recursos significativamente menor em comparação com outras tecnologias de virtualização. (Merkel; 2014).

Além disso, o Docker oferece um ecossistema robusto de ferramentas e serviços que facilitam a construção, o gerenciamento e a implantação de aplicativos em contêineres. Isso inclui o Docker Compose para definir e gerenciar aplicativos com vários contêineres, o Docker Swarm para orquestração de contêineres em escala e o Docker Hub para compartilhar e descobrir imagens de contêineres pré-construídas (Merkel; 2014).

Em resumo, o Docker é uma tecnologia revolucionária que simplifica o desenvolvimento e implantação de aplicativos por meio do uso de containers, oferecendo portabilidade, escalabilidade e eficiência de recursos. Com sua crescente adoção na indústria de software, o Docker se tornou uma ferramenta essencial para equipes de desenvolvimento que buscam acelerar o ciclo de vida de seus aplicativos (Merkel; 2014).

5.5 JAVASCRIPT

JavaScript é uma linguagem de programação de alto nível, interpretada e multi-paradigma, amplamente utilizada para desenvolvimento web. Originalmente desenvolvida pela Netscape, JavaScript se tornou uma das linguagens de programação mais populares e influentes na web, permitindo interatividade dinâmica e manipulação do conteúdo em páginas da web (Flanagan; 2011).

Uma das principais características do JavaScript é sua capacidade de ser executado diretamente no navegador do usuário, sem a necessidade de compilação prévia. Isso possibilita a criação de aplicações web interativas e responsivas, que respondem às ações do usuário em tempo real, sem a necessidade de recarregar a página (Flanagan; 2011).

JavaScript é uma linguagem orientada a objetos, embora também suporta outros paradigmas de programação, como programação funcional e programação imperativa. Isso permite que os desenvolvedores usem diferentes estilos de codificação para atender às necessidades específicas de seus projetos (Flanagan; 2011).

Além de ser amplamente utilizado no desenvolvimento front-end de páginas da web, JavaScript também é cada vez mais adotado no desenvolvimento de aplicativos back-end, graças a plataformas como Node.js. Com Node.js, os desenvolvedores podem usar JavaScript em ambientes de servidor, permitindo a criação de aplicativos web completos, desde o front-end até o back-end, com uma linguagem unificada (Flanagan; 2011).

Em resumo, JavaScript é uma linguagem de programação poderosa e versátil, essencial para o desenvolvimento web moderno, tanto no front-end quanto no back-end. Sua capacidade de executar no navegador do usuário e no servidor, juntamente com seu suporte a vários paradigmas de programação, o torna uma ferramenta indispensável para criar experiências web interativas e dinâmicas (Flanagan; 2011).

5.5.1 Vue.js

Vue.js é um framework progressivo de JavaScript usado para construir interfaces de usuário interativas e ricas para aplicativos da web. Desenvolvido por Evan You, Vue.js tem ganhado popularidade rapidamente devido à sua simplicidade, flexibilidade e excelente documentação (Vue.js; 2014).

Uma das principais vantagens do Vue.js é sua abordagem gradual e incremental para o desenvolvimento de aplicativos web. Os desenvolvedores podem começar a usar Vue.js em

partes específicas de seus projetos, sem a necessidade de reescrever todo o código existente, tornando-o fácil de integrar em projetos existentes (Vue.js; 2014).

Vue.js adota uma arquitetura de componentes, o que significa que as interfaces de usuário são construídas a partir de pequenos componentes reutilizáveis, cada um responsável por uma parte específica da interface. Isso promove a modularidade, facilita a manutenção do código e permite que os desenvolvedores criem interfaces complexas dividindo-as em componentes menores e mais gerenciáveis (Vue.js; 2014).

Além disso, Vue.js oferece uma reatividade eficiente e uma API simples e intuitiva para manipulação do DOM (Documento Object Model). Isso permite que os desenvolvedores criem interfaces de usuário responsivas que atualizam automaticamente conforme os dados mudam, sem a necessidade de escrever código manualmente para manipular o DOM (Vue.js; 2014).

Vue.js também é altamente extensível, permitindo que os desenvolvedores adicionem facilmente funcionalidades adicionais por meio de plugins e mixins. Isso significa que Vue.js pode ser adaptado para atender às necessidades específicas de diferentes projetos e equipes de desenvolvimento (Vue.js; 2014).

Em resumo, Vue.js é um framework JavaScript poderoso e flexível, ideal para o desenvolvimento de interfaces de usuário interativas e reativas para aplicativos da web. Sua abordagem gradual, arquitetura de componentes e reatividade eficiente o tornam uma escolha popular entre os desenvolvedores que procuram criar experiências web modernas e responsivas (Vue.js; 2014).

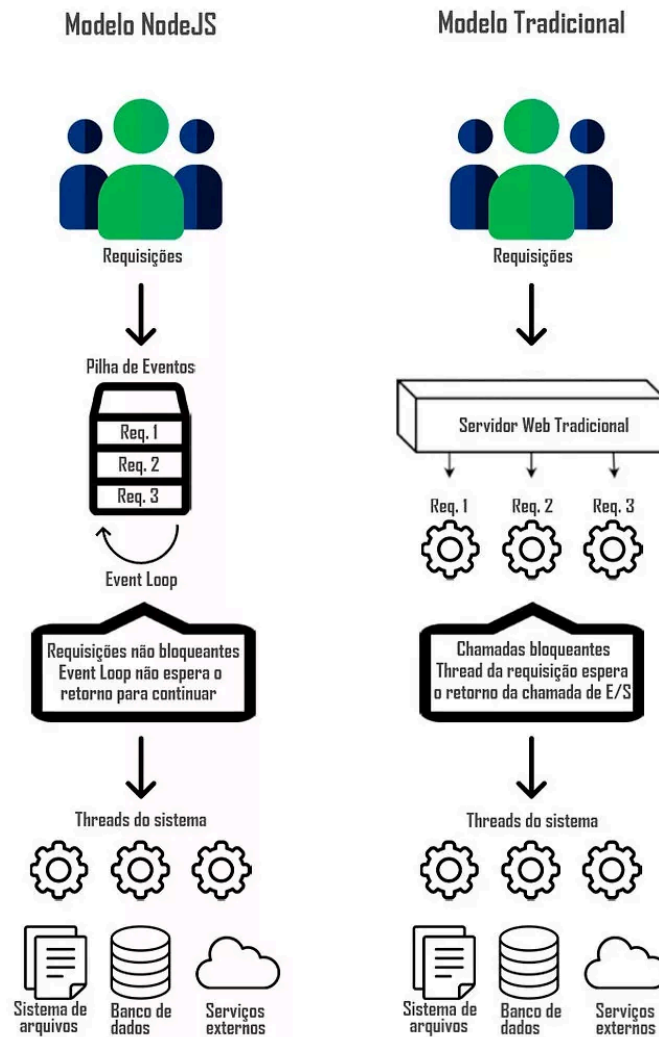
5.5.2 Node.js

Node.js é uma plataforma construída sobre o motor JavaScript do Google Chrome para facilmente construir aplicações rápidas e escaláveis. Essa tecnologia vem crescendo cada vez mais no mercado de desenvolvimento e, como sabemos, é amplamente usada por várias empresas do mundo. (FOX IOT, 2020).

A principal característica que diferencia o Node.js de outras tecnologias, como PHP e Java, é a sua execução single-thread. Apenas uma thread, chamada Event Loop, é responsável por executar o código JavaScript da aplicação. O Event Loop trata cada requisição como um evento e fica em execução contínua, aguardando novos eventos para processar. Em contraste, outras linguagens utilizam execução multithread, onde uma nova thread é criada para cada requisição. Quando o limite de threads é alcançado, novas requisições precisam esperar para

serem tratadas. (FOX IOT, 2020).

Figura 09: Diferença node.js versus modelo tradicional.



Fonte: (FOX IOT, 2020).

Apesar de ser single-threaded, a arquitetura do Node.js permite tratar um número maior de requisições concorrentes em comparação com o modelo tradicional. Isso ocorre porque ele evita o alto consumo computacional associado à criação e manutenção de threads para cada requisição. (OPUS, 2018).

Figura 10: Exemplo de utilização do node.js.

```
const http = require('node:http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Fonte: Elaborado pelo autor (2024).

A figura 10, representa um exemplo disponibilizado na documentação do Node.js, onde trata-se de permitir várias conexões simultâneas e onde cada chamada de conexão é ativada na função de retorno e em caso que não há chamadas, entrará em modo de espera.

5.5.2.1 Express.js

O Express.js, também conhecido como Express, é um framework backend ágil e semelhante ao Sinatra Node.js, fornecendo funcionalidades robustas e uma variedade de recursos para o desenvolvimento de aplicativos altamente escaláveis. Com um sistema de roteamento e ferramentas simplificadas, o Express permite a construção de componentes e estruturas robustas, oferecendo um conjunto completo de recursos para lidar com solicitações e respostas HTTP, roteamento e middleware, tornando-o ideal para o desenvolvimento de aplicativos web empresariais. (COUTINHO, 2023).

Opera como um framework cliente-servidor, aceitando solicitações dos usuários e enviando respostas, seguindo um comportamento semelhante a outros frameworks como o Laravel. Quando um usuário envia uma solicitação HTTP, o servidor a recebe através de rotas definidas e a processa com um controlador correspondente. Em seguida, o servidor envia uma resposta ao cliente usando HTTP, estabelecendo uma comunicação bidirecional. Essa resposta pode ser em formato de texto ou HTML dinâmico, facilmente processado pelo navegador, ou em JSON, permitindo que os desenvolvedores frontend exibam as informações na página da web. (COUTINHO, 2023).

Figura 11: Exemplo utilizando express.js.

```
import express from 'express';  
  
const app = express();  
  
app.listen(3000, () =>  
  console.log('Servidor iniciado na porta 3000')  
);
```

Fonte: Elaborada pelo autor.

Neste caso acima (figura 11), passamos como parâmetro a porta (3000) e uma arrow function para exibir um log com a mensagem de que tudo está

5.5.2.2 Swagger.js

Swagger é uma especificação aberta que define APIs REST, fornecendo um documento que descreve os recursos disponíveis, as operações que podem ser executadas nesses recursos e os parâmetros associados a essas operações. Similar ao WSDL para serviços web SOAP, o documento Swagger delinea detalhes cruciais da API, como os tipos e a obrigatoriedade dos parâmetros, além de informações sobre os valores aceitáveis. Adicionalmente, o documento pode incluir JSON Schema para descrever a estrutura das solicitações e respostas da API REST. (IBM, 2022).

Uma maneira simplificada de criar documentação no Swagger é através do Swagger Editor, utilizando arquivos JSON ou YAML. Essa abordagem elimina a dependência de uma linguagem técnica específica, já que a documentação é criada seguindo as regras da OpenAPI e é facilmente editável no formato YAML. (MARTINS, 2022).

Ao detalhar os recursos, operações e parâmetros disponíveis em uma API REST, o documento Swagger oferece uma estrutura clara para desenvolvedores entenderem e interagirem com a API de forma eficaz. Essa especificação é essencial para garantir a consistência e a compreensão entre os desenvolvedores que colaboram em projetos que utilizam APIs REST, fornecendo uma base sólida para o desenvolvimento e integração de sistemas baseados em serviços web. (IBM, 2022).

5.5.2.3 PG

PG, também conhecido como `node-postgres`, é um módulo do Node.js que permite a comunicação com bancos de dados PostgreSQL. Desenvolvido para ser um cliente PostgreSQL puro em JavaScript, o PG facilita a execução de consultas SQL, a manipulação de transações e a interação com bancos de dados PostgreSQL diretamente do ambiente Node.js (Schmidt, 2016).

Uma das características distintivas do PG é sua capacidade de lidar com operações de banco de dados de forma assíncrona, alinhando-se à natureza assíncrona do Node.js. Isso permite que os desenvolvedores executem consultas SQL de forma eficiente e sem bloqueio, garantindo a escalabilidade e o desempenho de aplicativos que utilizam o PostgreSQL como banco de dados (Schmidt, 2016).

Além disso, o PG oferece uma API simples e intuitiva que permite aos desenvolvedores estabelecer conexões com bancos de dados PostgreSQL, executar consultas SQL parametrizadas, lidar com erros de forma robusta e processar resultados de consultas de forma eficiente (Schmidt, 2016).

Em resumo, o PG é uma ferramenta essencial para desenvolvedores que desejam integrar aplicativos Node.js com bancos de dados PostgreSQL. Sua capacidade de interagir de forma eficiente e assíncrona com o PostgreSQL, juntamente com sua API simples e robusta, torna-o uma escolha popular entre os desenvolvedores que buscam criar aplicativos web escaláveis e de alto desempenho com o Node.js e o PostgreSQL (Schmidt, 2016).

5.5.2.3 JSPDF

Não é surpresa que o JavaScript se tornou uma solução universal para uma variedade de desafios tecnológicos. Desde aplicações web a Internet das Coisas (IoT), a linguagem permite realizar praticamente qualquer tarefa. O mesmo se aplica à geração de arquivos em PDF: o JavaScript também é uma solução eficaz (PINHO, 2019).

Ao buscar na internet maneiras de criar PDFs, você encontrará uma ampla gama de opções, que variam entre boas, medianas e ruins. A biblioteca que mais tem destaque é o jsPDF que se apresenta como a melhor alternativa (PINHO, 2019).

5.5.2.3 Leaflet.js

Quando se fala em mapas na internet, o Google Maps é frequentemente a primeira opção que vem à mente. No entanto, existem várias alternativas disponíveis. Ao desenvolver uma aplicação web que utiliza essa solução, corre-se o risco de vendor lock-in, que é a dependência de um único fornecedor de mapas. A biblioteca Leaflet.js se destaca como uma solução *vendor agnostic*, permitindo que você troque de fornecedor com facilidade, utilizando a mesma base de código (PEREIRA, 2016).

Um dos principais pontos positivos é a flexibilidade. Soluções baseadas no OpenStreetMap, uma base cartográfica colaborativa de licença livre, oferecem uma ampla gama de funcionalidades que não são viáveis com o Google Maps, como acesso offline, mapas vetoriais e personalização da aparência do mapa. Além disso, o custo é um fator relevante; muitas empresas estão adotando provedores de mapas alternativos. Por exemplo, o Foursquare e o Strava passaram a utilizar mapas renderizados pela Mapbox, que se baseiam em dados do OpenStreetMap (PEREIRA, 2016).

5.6 BANCO DE DADOS

Originando do termo inglês Databanks e posteriormente trocado para Database, um banco de dados é considerado um conjunto de dados persistentes e armazenados com um determinado objetivo e que atende a diversos usuários ou de uma organização. As vantagens da utilização de um banco de dados para armazenamento dos dados se dá em relação ao controle centralizado, independência e a garantia de integridade dos dados em questão (COSTA, 2011).

Ainda quando falamos de banco de dados, temos dois tipos:

- Banco de dados relacional: Um banco de dados relacional é constituído por tabelas, criando relacionamento entre as próprias tabelas, por meio de chaves denominadas 56 primárias e estrangeiras, permitindo o uso dos dados de forma associada além de garantir o uso destes dados de forma eficiente e com integridade (CALANCA, 2016);

- Banco de dados não relacional: Um banco de dados não relacional não é constituído por tabelas e existem diversos modelos de uso de um banco não relacional, como: grafos e chave-valor, o que oferece uma flexibilidade e escalabilidade muito vantajosa quando se tem um alto número de dados (CALANCA, 2016).

Para o desenvolvimento deste projeto, atendendo aos requisitos especificados e identificado que há a necessidade da relação dos dados, foi optado pelo uso do modelo relacional dos bancos de dados, em específico o uso do PostgreSQL.

5.6.1 PostgreSQL

O PostgreSQL é um sistema de gerenciamento de banco de dados relacional de código aberto amplamente aclamado por sua robustez, desempenho e capacidade de extensibilidade. Destaca-se por sua aderência aos padrões SQL e pelo suporte a recursos avançados de banco de dados, como arrays, JSON e JSONB (POSTGRESQL, 2023).

Conhecido por sua reputação, desde a confiabilidade e flexibilidade até mesmo por aderir ao código aberto, o PostgreSQL, que surgiu em 1986 como "POSTGRES" a partir da ideia de Michael Stonebraker, vem crescendo ao longo do tempo e tem como características principais o desempenho, escalabilidade, suporte de simultaneidade e código aberto (IBM, 2023).

Neste contexto, destacamos três fundamentos que fundamentam a escolha do PostgreSQL. Primeiramente, a natureza de software livre do PostgreSQL não apenas promove acessibilidade, mas também propicia inovação e flexibilidade, eliminando custos associados a licenciamento. Em segundo lugar, a notável capacidade de escalabilidade do sistema facilita a administração eficiente de grandes volumes de dados.

Por fim, o ecossistema robusto do PostgreSQL é marcado por uma comunidade engajada de colaboradores, constantemente aprimorando o sistema e oferecendo suporte, resolução de problemas e compartilhamento de conhecimentos. Esses aspectos combinados fundamentam a escolha do PostgreSQL como uma solução confiável e eficiente para aqueles que almejam excelência em gerenciamento de banco de dados (EDUCAÇÃO, 2022).

6 DESENVOLVIMENTO DO PROJETO

Nesta seção do projeto, exploraremos detalhadamente a criação do software, com foco nas tecnologias previamente mencionadas. Após uma análise minuciosa dessas ferramentas, destacamos como elas desempenharam um papel fundamental na concepção, arquitetura e implementação do sistema, transformando a teoria em prática.

A seção está estruturada em três partes: Planejamento, Backend e Frontend. Cada uma dessas áreas aborda suas respectivas responsabilidades e contribuições no processo de desenvolvimento, evidenciando seu impacto no sucesso do projeto.

6.1 PLANEJAMENTO

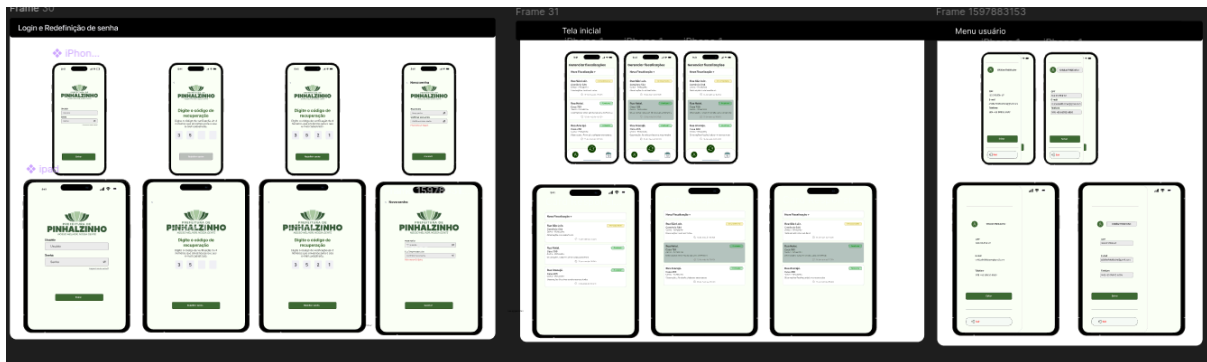
Nesta etapa, discutiremos as atividades realizadas antes do início do desenvolvimento da aplicação, incluindo o planejamento e a organização do projeto. Serão abordados tópicos como a prototipação e o versionamento de código, essenciais para garantir uma estrutura sólida e o controle eficiente do progresso durante todo o processo de desenvolvimento.

6.1.1 Prototipação no Figma

Nesta etapa, exploraremos a aplicação prática do Figma no processo de prototipação. A utilização do Figma não apenas facilita a visualização das funcionalidades, fluxos e interações previstas, mas também serve como uma ferramenta valiosa para o refinamento contínuo e validação do design antes da implementação. Esse processo dinâmico no Figma desempenha um papel crucial no desenvolvimento, promovendo uma comunicação eficaz entre a equipe e permitindo a convergência para soluções que sejam visualmente impactantes e funcionalmente sólidas.

Prototipar é o processo de criar um modelo preliminar de uma ideia para testá-la, validar sua viabilidade e obter feedback antes da implementação final. A prototipação pode ser física ou digital, variando de esboços simples a modelos complexos, e é utilizada em diversas áreas, como design, engenharia e desenvolvimento de software. Ela permite testar, identificar problemas e refinar a solução antes de seu lançamento final (MOSHE, 2023).

Figura 12: Prototipação do projeto no Figma



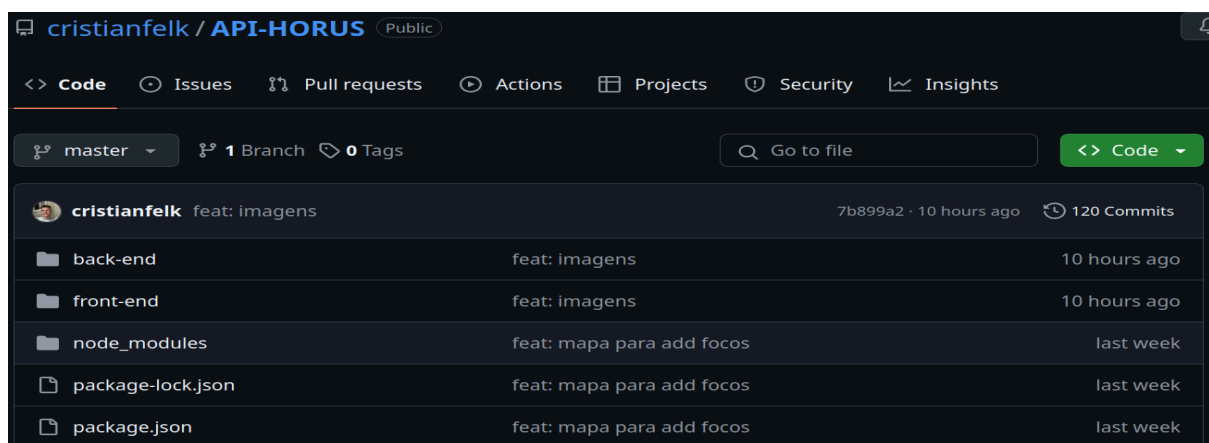
Fonte: Elaborado pelo autor (2024).

Representado na Figura 12, temos a prototipação realizada na plataforma Figma referente às telas iniciais de login, controle de denúncias e também de informações do usuário. Este passo foi essencial para entender e estudar ainda mais como seria a experiência do usuário ao utilizar o sistema.

6.1.2 Versionamento de código

Antes de iniciar o desenvolvimento do projeto, foi organizado cuidadosamente o ambiente de trabalho. Para isso, é adotado a prática de armazenar todo o progresso na nuvem, garantindo que, em caso de imprevistos, nenhum dado seja perdido. Para assegurar essa confiabilidade e controle, é utilizado o Git como sistema de versionamento e o GitHub como plataforma de hospedagem, permitindo o acompanhamento e a recuperação de todas as versões do projeto de forma eficiente.

Figura 13: Repositório do software no Github



Fonte: Elaborado pelo autor (2024).

Dessa forma, conforme representado através da figura 13, já configuramos os repositórios no GitHub, o que possibilita o desenvolvimento em qualquer máquina, mantendo sempre a versão mais atualizada. Além disso, essa prática nos permite preservar um histórico detalhado de todas as alterações realizadas ao longo do projeto.

6.2 BACKEND

O backend pode ser visto como a "ponte" entre os dados manipulados pelo navegador e o banco de dados, representando tudo que ocorre nos bastidores de uma aplicação. Ele engloba desde a lógica de negócios até as validações e interações com o banco de dados.

O desenvolvimento backend foca na construção da parte "invisível" da aplicação, responsável pela execução das regras de negócios, processamento de dados, comunicação com o banco de dados e gerenciamento das requisições. Essas funcionalidades, embora fundamentais, não são diretamente percebidas pelos usuários finais.

Neste tópico, discutiremos as configurações e estruturas essenciais para o desenvolvimento do backend, incluindo as definições do Docker, a arquitetura do banco de dados, da API e a organização das pastas no projeto.

6.2.1 Configuração docker

Para uso do Docker no protótipo, a exemplo da imagem 15, foi necessário a criação de um arquivo denominado **Dockerfile**, utilizado tanto no banco de dados para o Postgres quanto para a api, no Node. Este arquivo é composto inicialmente utilizando o *from* que indica a imagem a ser utilizada no Docker, que neste caso foi o postgres:15.2 e o node:18-alpine além de demais informações essenciais como as credenciais do banco de dados, por exemplo (figura 14).

Figura 14: Dockerfile presente no protótipo

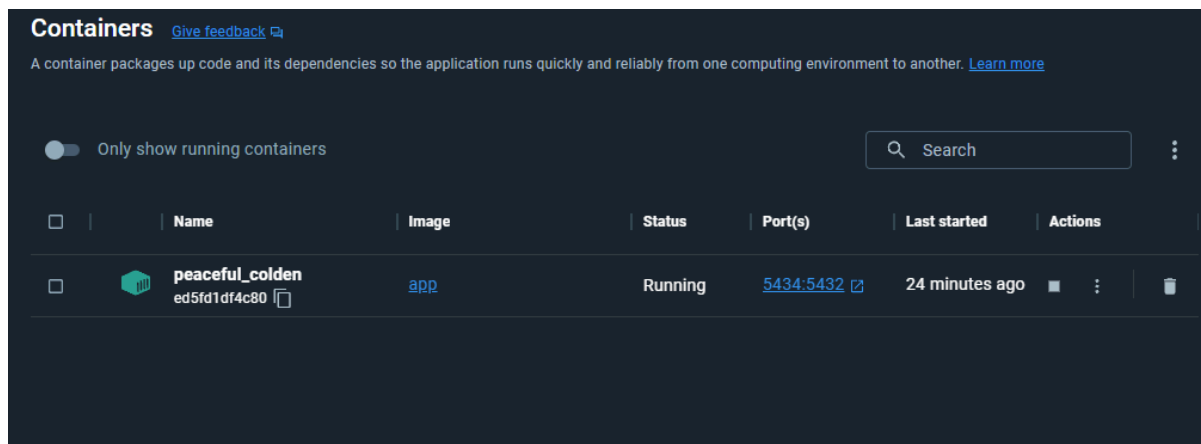
```
back-end > src > db > Dockerfile > ...
You, 3 days ago | 1 author (You)
1 FROM postgres:15.2-alpine
2
3 COPY init.sql /docker-entrypoint-initdb.d/
4
5 ENV POSTGRES_USER admin
6 ENV POSTGRES_PASSWORD admin
7 ENV POSTGRES_DB APP_SECRETARIA You, 3 d
```

Fonte: Elaborado pelo autor (2024).

Após ter configurado o ambiente do Docker, é necessário criar as imagens, dentro da pasta do backend, que podem ser utilizadas pelos seguintes comandos no terminal:

- `docker build -t app .`
- `docker run -p 5434:5432 -d app`

Figura 15: Imagem Docker referente ao backend



Fonte: Elaborado pelo autor (2024).

Após executar os devidos comandos, pode ser identificado através da figura 15, o container em execução, referente ao banco de dados e API

6.2.2 Estrutura do Banco de Dados

Com base na estrutura dos diagramas, foram criadas as tabelas no banco de dados. Essas tabelas, definidas no PostgreSQL, são integradas à API, permitindo a manipulação dos dados conforme necessário, seja para consulta, inserção, edição ou exclusão.

Para facilitar a criação das tabelas, foi desenvolvido um arquivo chamado **init.sql**, que contém a definição de todas as tabelas, além das chaves primárias e estrangeiras. Esse arquivo garante que, ao iniciar o Docker, as tabelas sejam automaticamente criadas no PostgreSQL, assegurando a estrutura do banco de dados desde o início.

Figura 16: Utilização do PostgreSQL (criação das tabelas).

```

init.sql x
back-end > src > db > init.sql
You, 2 minutes ago | 2 authors (You and one other)
1 create table uf (
2     id serial primary key not null,
3     nome varchar(250),
4     sigla varchar(2) unique,
5     ibge integer
6 );
7
8 create table monitoramento (
9     id serial primary key not null,
10    casos_confirmados integer,
11    casos_mortes integer,
12    casos_monitorados integer,
13    casos_ativos integer
14 );
15
16 create table focos_dengue (
17     id serial primary key,
18     descricao varchar(255) not null,
19     confirmado boolean,
20     latitude decimal(9, 6) not null,
21     longitude decimal(9, 6) not null,
22     data_registro timestamp default current_timestamp
23 );
24
25 create table municipio (
26     id integer primary key not null,
27     nome varchar(250) not null,
28     ibge integer,
29     uf varchar(2),
30     foreign key (uf) references uf (sigla)
31 );
32
33 create table logradouro (
34     id serial primary key not null,
35     municipio_id integer,
36     cep varchar(9),
37     logradouro varchar(120),
38     complemento varchar(120),
39     bairro varchar(60),
40     foreign key (municipio_id) references municipio (id)
41 );
42

```

Fonte: Elaborado pelo autor (2024)

Na representação da figura 16 temos alguns exemplos do uso do PostgreSQL, com script de criação de algumas tabelas do banco de dados, tabelas de uf, monitoramento, focos_dengue, município e logradouro.

6.2.3 Desenvolvimento API

No backend, foi desenvolvida a API e estabelecida a conexão com o banco de dados PostgreSQL, permitindo a manipulação dos dados por meio de requisições. Nesse processo, foram aplicadas todas as tecnologias abordadas até o momento, além de boas práticas de programação, com foco na organização e padronização do código.

Durante o desenvolvimento da API, diversos pacotes foram instalados para atender a diferentes propósitos, conforme detalhado nos tópicos a seguir, utilizando os comandos listados (figura 17).

- **npm i express**: Instala o pacote Express, utilizado para o desenvolvimento da API;
- **npm i swagger-ui-express**: instala o pacote Swagger, utilizado para a documentação dos endpoints da API;
- **npm i swagger-autogen**: instala o pacote que gera automaticamente a documentação a partir dos endpoints criados.
- **npm i nodemailer**: instala o pacote Nodemailer, utilizado para o envio de e-mails através da API;
- **npm i multer**: instala o pacote Multer, utilizado para o upload e manipulação de arquivos enviados via requisições HTTP.

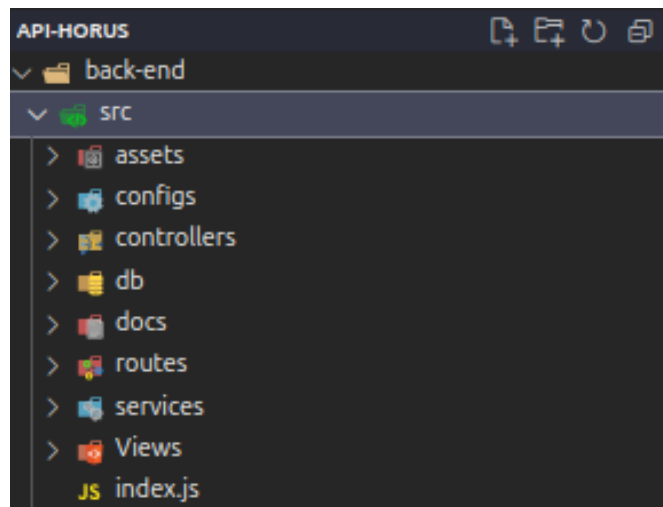
Figura 17: Dependências utilizadas para backend.

```
"dependencies": {  
  "cors": "^2.8.5",  
  "express": "^4.18.2",  
  "multer": "^1.4.5-lts.1",  
  "nodemailer": "^6.9.14",  
  "pg": "^8.11.3",  
  "swagger-autogen": "^2.23.1",  
  "swagger-ui-express": "^4.6.3"  
},
```

Fonte: Elaborado pelo autor (2024)

A figura 18, demonstra toda a estrutura de pastas criadas no backend:

Figura 18: Estrutura das pastas presente no backend



Fonte: Elaborado pelo autor (2024)

Na pasta denominada SRC temos diversos arquivos, dentre os principais temos:

- Assets: Pasta responsável por armazenar uploads de usuários.
- Configs: Pasta responsável por arquivos que contém a conexão com o banco de dados.
- Controllers: Pasta responsável pelos arquivos que gerenciam os retornos das chamadas da API e o processamento dos parâmetros de entrada.
- Db: Pasta onde armazena o arquivos Dockerfile, que se refere a criação do container da imagem do banco Postgresql,
- Docs e Views: Pastas responsáveis pela documentação da API com swagger.js.
- Routes: Pasta responsável por armazenar os arquivos que gerenciam o redirecionamento das requisições feitas à API.
- Services: Pasta responsável pelos arquivos que contêm as regras de negócio e realizam a manipulação dos dados.

Figura 19: Exemplo de método Post (inserção) na pasta services

```
const postUsuario = async (params) => {
  try {
    const salt = crypto.randomBytes(16).toString('hex');
    const senhaCriptografada = crypto.createHash('sha256').update(params.senha + salt).digest('hex');

    const sql_post = `insert into usuario (nome, login, senha, salt, email, status, data_cadastro)
      values ($1, $2, $3, $4, $5, $6, current_date)`;
    const values = [params.nome, params.login, senhaCriptografada, salt, params.email, params.status];
    await db.query(sql_post, values);
  } catch (error) {
    console.error('Erro ao cadastrar usuário:', error);
  }
}
```

Fonte: Elaborado pelo autor (2024)

Figura 20: Exemplo de método Post (inserção) na pasta routes

```
back-end > src > routes > JS usuarioRouter.js > ...
...
1  const usuarioController = require('../controllers/usuarioController');
2
3  module.exports = (app) => {
4    app.post('/usuario', usuarioController.postUsuario)
5    app.get('/usuario', usuarioController.getUsuario)
6    app.get('/usuario/:id', usuarioController.getUsuarioById)
7    app.delete('/usuario/:id', usuarioController.deleteUsuario)
8    app.put('/usuario/:id', usuarioController.putUsuario)
9    app.patch('/usuario/:id', usuarioController.patchUsuario)
10 }
```

Fonte: Elaborado pelo autor (2024)

Figura 21: Exemplo de método Post (inserção) na pasta controllers

```
3  const postUsuario = async (req, res) => {
4    try {
5      const emailInvalido = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
6
7      if (!emailInvalido.test(req.body.email)) {
8        return res.status(400).json({ error: "Formato de email inválido" });
9      }
10
11      const usuarioPorEmail = await usuarioService.getUsuarioByEmail(req.body.email);
12      if (usuarioPorEmail) {
13        return res.status(400).json({ error: "E-mail já cadastrado" });
14      }
15
16      const usuarioPorLogin = await usuarioService.getUsuarioByLogin(req.body.login);
17      if (usuarioPorLogin) {
18        return res.status(400).json({ error: "Login já cadastrado" });
19      }
20
21      await usuarioService.postUsuario(req.body);
22      res.status(201).json({ message: "Usuário cadastrado com sucesso" });
23
24    } catch (err) {
25      res.status(500).json({ error: "Erro ao cadastrar usuário", message: err.message });
26    }
27  };
```

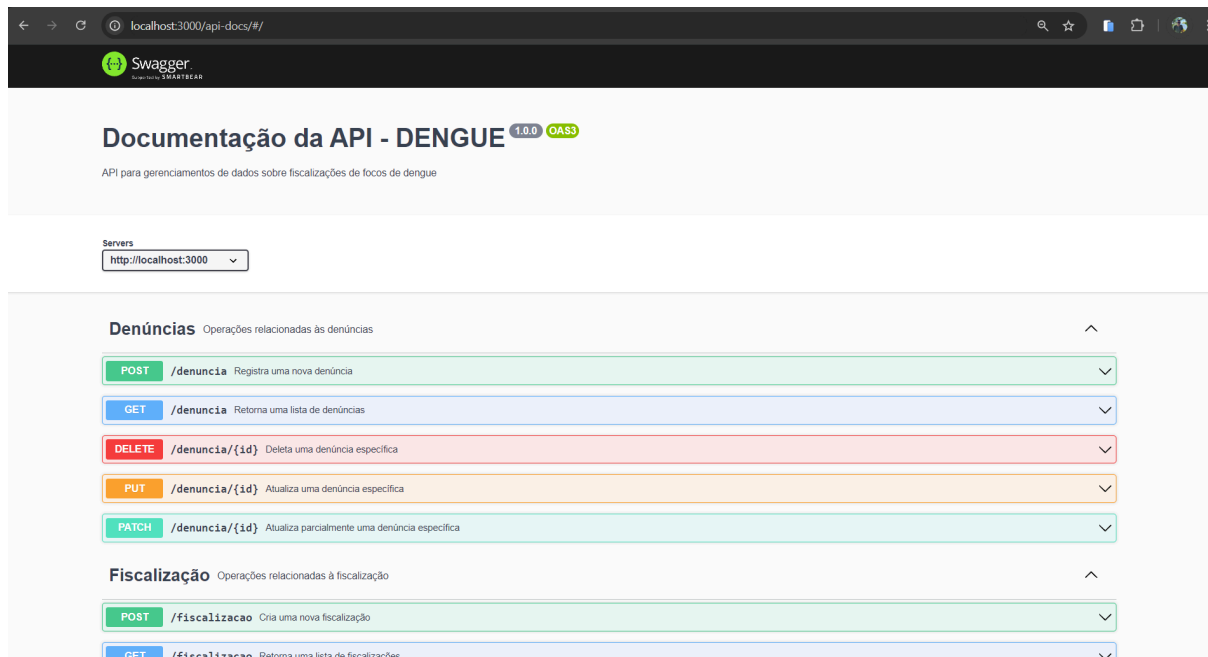
Fonte: Elaborado pelo autor (2024)

Nas figuras 19, 20 e 21, é apresentado o desenvolvimento da API responsável pela inserção de dados de um usuário na tabela "usuario" do banco de dados PostgreSQL. A figura 19 ilustra o service, responsável pela manipulação dos dados, pela conexão com o banco e pela aplicação das regras de negócio. Na figura 20, é mostrado o arquivo de routes, que redireciona as requisições para os métodos adequados, encaminhando-os posteriormente ao controller. Já a figura 21 destaca o controller, que trata os parâmetros de entrada e define os retornos das chamadas realizadas.

6.2.4 Documentação utilizando Swagger.js

O uso do Swagger segue no conceito de manter as boas práticas de programação, neste caso de organização, agora documentando a API, visto que a API em questão pode ser utilizada por outras integrações e a partir disso, é necessário uma descrição do que é capaz, de qual forma deve ser utilizada e o que faz cada requisição disponibilizada.

Figura 22: Exemplo de documentação utilizando Swagger.js



Fonte: Elaborado pelo autor (2024)

Assim, como ilustrado na figura 22, o Swagger.js facilita a documentação da API, reunindo todos os endpoints disponíveis em um único local. Ele cria uma interface abrangente que apresenta todos os métodos e parâmetros de forma clara e acessível.

6.3 FRONTEND

O front-end é responsável pela interface gráfica de um projeto, onde o usuário interage diretamente com sites, aplicativos ou sistemas. O desenvolvedor front-end não desenha a interface, mas escreve o código que define as ferramentas e funcionalidades com as quais o usuário irá interagir. A usabilidade e experiência do usuário são prioridades no desenvolvimento front-end (TOTVS, 2021).

Sua importância está em garantir que a aplicação seja fácil de usar e funcione corretamente. Sites com erros ou lentidão perdem usuários e confiança. Além disso, a responsividade para dispositivos móveis é crucial devido ao crescente tráfego vindo desses dispositivos (TOTVS, 2021).

6.3.1 Instalação do Vue

O Vue é utilizado no projeto como um facilitador, mantendo os conceitos familiares de HTML, CSS e JavaScript padrão, mas introduzindo uma organização diferenciada, separada em seções, cada uma com um papel específico:

- **Template:** Define a estrutura e a apresentação visual do componente, abrigando o HTML que será exibido ao usuário.
- **Script:** Seção responsável pela lógica do componente, onde se define a manipulação de dados, métodos e outros comportamentos utilizando JavaScript.
- **Style:** Lida com a estilização do componente, contendo o CSS que permite ajustar a aparência conforme necessário.
-

Para começar, é preciso instalar o Vue com o seguinte comando: *npm install vue* (figura 23).

Figura 23: Instalação Vue

```
cris@ACER-CRISTIAN MINGW64 ~/OneDrive/Área de Trabalho/TCC-DENGUE
● $ npm install vue
Debugger listening on ws://127.0.0.1:60877/b77a8df0-7fe4-43d6-8966-e20dc1d2610d
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.

added 24 packages in 7s


3 packages are looking for funding
  run `npm fund` for details
Waiting for the debugger to disconnect...
```

Fonte: Elaborado pelo autor (2024)

Após a instalação do Vue, é necessário criar um projeto com o seguinte comando:

- `vue create <nome>`

Figura 24: Exemplo do comando vue create



```
PROBLEMS OUTPUT TERMINAL PORTS GITLENS DEBUG CONSOLE

Vue CLI v5.0.8
? Please pick a preset: (Use arrow keys)
> Default ([Vue 3] babel, eslint)
  Default ([Vue 2] babel, eslint)
  Manually select features
```

Fonte: Elaborado pelo autor (2024)

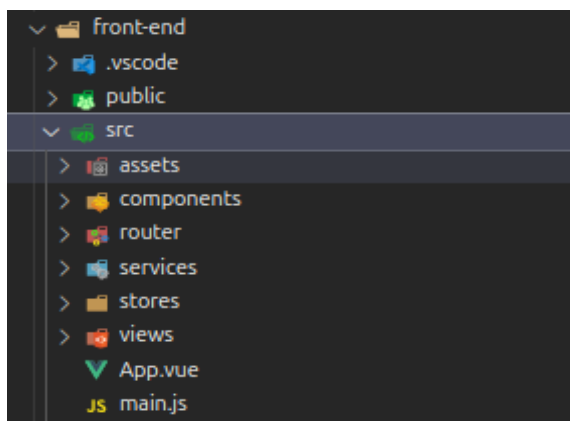
Após utilizar o comando da criação, devem ser seguidas as opções conforme necessidade do projeto, na escolha do Vue 2 ou Vue 3 e outras dependências (figura 24).

6.3.3 Estrutura de pastas

Nesta etapa, será descrita a organização da estrutura de pastas, um aspecto essencial no processo de desenvolvimento. A estrutura de pastas desempenha um papel fundamental na manutenção e organização do projeto, facilitando a localização e o gerenciamento eficiente dos arquivos.

A figura 25, apresentada a seguir, ilustra a estrutura de pastas criada no frontend após a instalação do Vue. Cada pasta possui uma função específica, contribuindo para uma arquitetura mais organizada e modular.

Figura 25: Estrutura das pastas do Front-end no sistema



Fonte: Elaborado pelo autor (2024)

Na pasta denominada SRC temos diversos arquivos, dentre os principais temos:

- Assets: Responsável por armazenar imagens fixas, que são utilizadas no front-end do software.
- Components: Responsável por ter todos os componentes que são utilizados no software.
- Router: Responsável pelo redirecionamento entre todas as interfaces, a qual é definido as rotas para um componente.
- Services: Responsável por comportar o arquivo apiService.js, o qual possui a configuração do Axios, que faz comunicação com a API.
- Stores: Responsável por possuir configuração do Pinia, essa estrutura permite gerenciar e acessar um contador simples de forma reativa em uma aplicação Vue.js.
- Views: Representa as principais interfaces do sistema, na sua maioria sendo composta por outro arquivo da pasta components.
- App.vue: Serve como o ponto de entrada da aplicação, fornecendo uma estrutura básica e estilizada para a renderização das rotas.
- Main.js: Responsável pela criação da aplicação, funcionalidades extras, e também instanciar e criar a URL base para utilização do Axios.

Figura 26: Exemplo de uso do Vue Router

```
{
  path: '/AdminFocoDengue',
  name: 'AdminFocoDengue',
  component: () => import('@/views/AdminFocoDengue.vue'),
},
```

Fonte: Elaborado pelo autor (2024)

Na imagem acima, na figura 26, está ilustrado o sistema de rotas do protótipo, destacando uma rota específica e seu redirecionamento ao ser acionada pela propriedade "to" ou pelo uso do método `router.push` no código.

Figura 27: Exemplo de uso do Vue Router

```
<li class="sidebar-item">
  <router-link to="/AdminFocoDengue">
    
    <span v-if="!isSidebarCollapsed">Focos</span>
  </router-link>
</li>
```

Fonte: Elaborado pelo autor (2024)

Na representação acima, na figura 27, mostramos a utilização do Vue Router ao clicar em um botão. Nesse contexto, a propriedade "to" indica o caminho de destino da ação de clique, que, neste caso, redireciona para a URL de registro.

6.3.4 Interfaces

Nesta etapa, serão exploradas as interfaces do sistema, que desempenham um papel essencial na interação entre o usuário e o software.

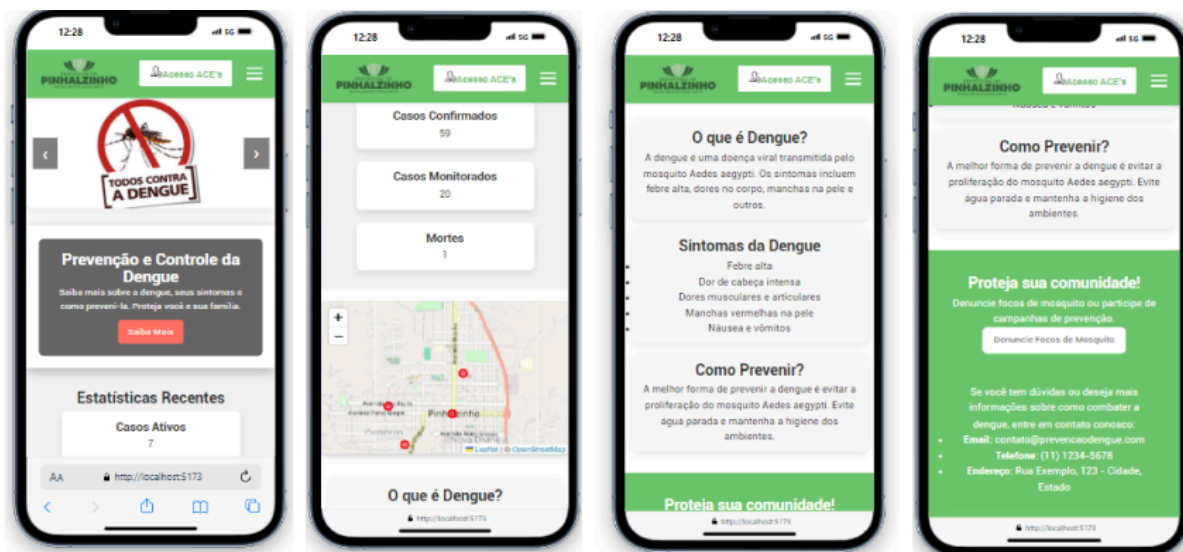
Formalmente, uma interface inclui elementos explícitos no código, como assinaturas de métodos. Informalmente, envolve regras de uso e comportamentos que não são expressos diretamente na linguagem, mas precisam ser entendidos pelo desenvolvedor (GUIMARÃES, 2023).

6.3.4.1 Interfaces para a comunidade (web e mobile)

Nesta etapa será descrito e representado em imagens todas as interfaces sobre o uso para a comunidade. O uso para população foi elaborado para que pudessem efetuar um acompanhamento sobre os focos e estatísticas, assim como ter acesso a um formulário para que possam efetuar uma denúncia de um novo foco. As interfaces podem ser acessadas tanto por mobile quanto web.

Visando uma interface simples e intuitiva para a população, com dados importantes para acompanhamento, foi elaborada uma página para que acompanhem os resultados e fiquem por dentro do que está ocorrendo atualmente (figura 28).

Figura 28: Interface inicial do sistema (mobile)



Fonte: Elaborado pelo autor (2024)

Na figura 28, temos a representação da página inicial do sistema para mobile, a qual possui carrossel de imagens, informativos sobre prevenção contra a dengue, estatísticas sobre o município de Pinhalzinho, mapa com locais de focos ativos, botão para efetuar denúncia de foco e contato da secretária de combate à dengue.

Ainda seguindo detalhes informativos da página inicial do software, ao acessar o botão de “Saiba Mais”, será direcionado para página de prevenção (figura 29).

Figura 29: Interface ao acessar “Saiba Mais” (mobile)

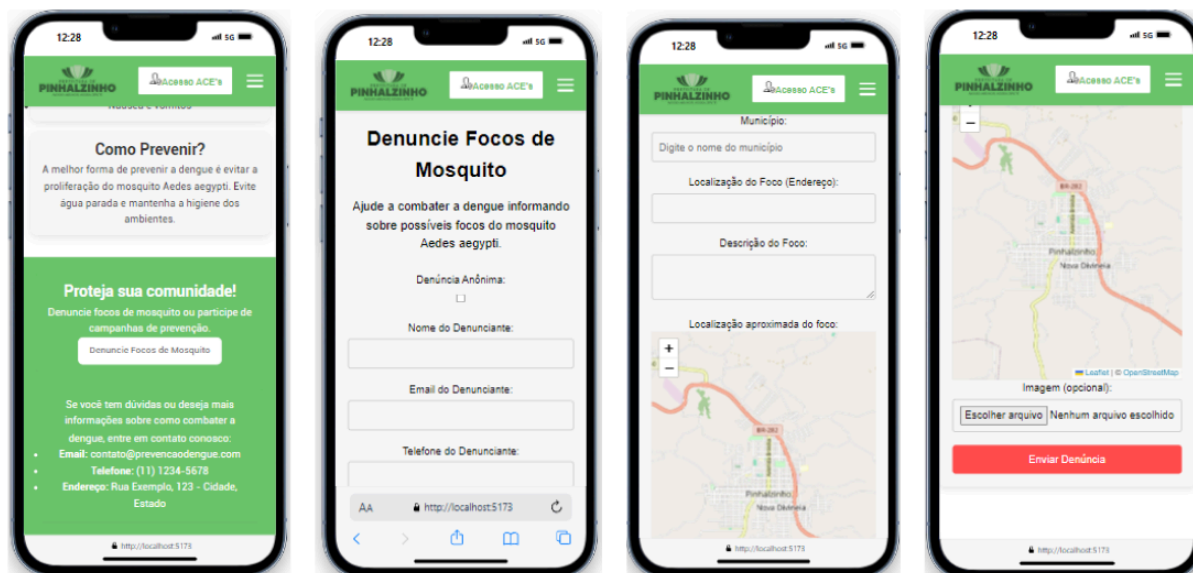


Fonte: Elaborado pelo autor (2024)

Na figura 29, temos a representação sobre a interface de informações para prevenção da dengue, sendo possível acessá-la pelo botão “Saiba Mais” destacado na cor vermelha.

Para que pudesse ser atendida a necessidade da comunidade, foi elaborado uma interface onde é possível efetuar denúncias de focos de dengue, conforme figura 30.

Figura 30: Interface ao acessar “Denuncie Focos de Mosquito” (mobile)



Fonte: Elaborado pelo autor (2024)

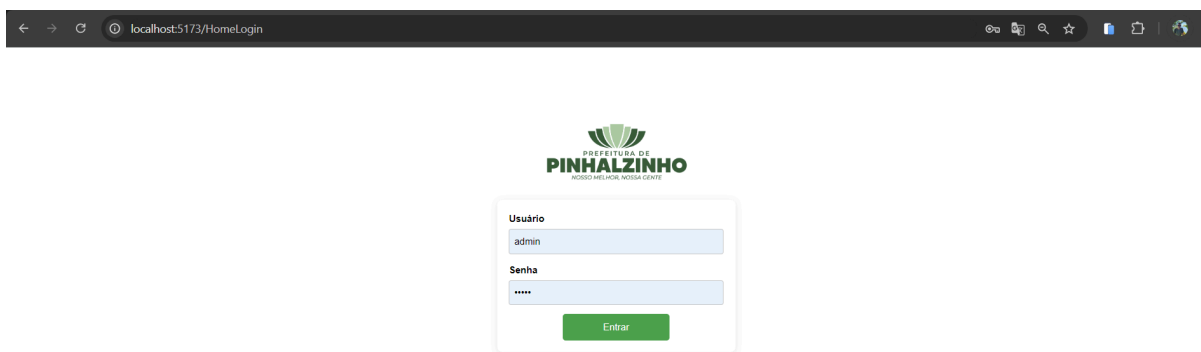
Na figura 30, temos a representação sobre a interface de denúncias de focos de dengue, onde é acessível pelo botão “Denuncie Focos de Mosquito”, ao usuário que irá efetuar a denúncia é apresentado um formulário para que informe alguns dados, nome, e-mail, telefone, município, localização do foco, breve descrição da denúncia, um mapa onde pode ser apontado a localização aproximada do foco, enviando para a denúncia informações de latitude e longitude e por último campo de anexo, o qual pode ser adicionado uma imagem para a denúncia. No formulário possui um checkbox denominado “Denúncia Anônima”, caso marcado, não será necessário preencher os campos nome, e-mail e telefone.

6.3.4.2 Interfaces de gestão do secretário (web e mobile)

Nesta etapa será descrito e representado em imagens todas as interfaces sobre o uso da secretaria de combate a dengue (gestão). As interfaces podem ser acessadas tanto por mobile quanto web.

Necessário efetuar login para que seja acessado com perfil de administrador, onde possui acesso total no software (figura 31).

Figura 31: Interface de login do sistema

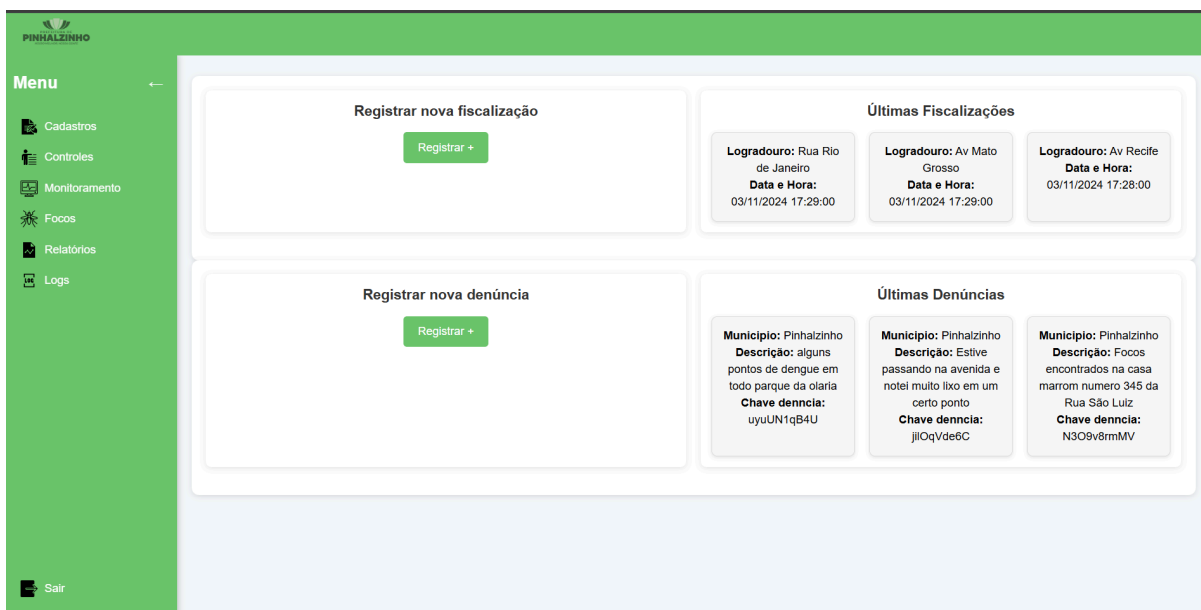


Fonte: Elaborado pelo autor (2024)

Na figura 31, está representada a tela de login do software, sendo necessário usuário e senha para acessar.

Após efetuar o login, o usuário é direcionado ao dashboard, onde possui visão sobre diversas funcionalidades, representado na figura 32.

Figura 32: Interface do dashboard inicial

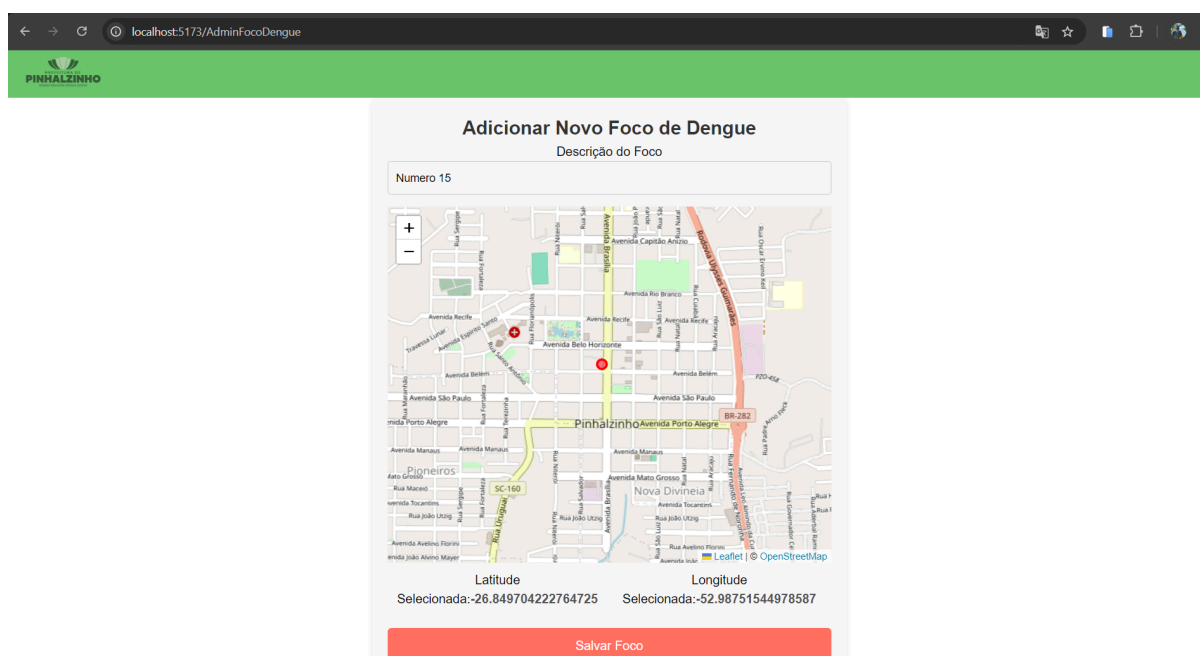


Fonte: Elaborado pelo autor (2024)

Na figura 32, está representada a interface inicial do software, após efetuar o login, será direcionado para o dashboard acima. A interface inicial, é composta por menu superior (navbar) com logo do município de Pinhalzinho SC, menu lateral (sidebar), que possui as principais funcionalidades, cadastros, controles, monitoramento, focos, relatórios, logs e opção de logout. A interface inicial também é composta por duas sessões de cards, fiscalização, que contém cards com as últimas fiscalizações emitidas e a opção de criar uma nova fiscalização, e também o card de denúncias com as últimas denúncias recebidas e opção para registrar nova denúncia.

Na rotina do usuário, o mesmo poderá adicionar novos focos de dengue (ativos), esta opção está disponível no menu lateral (sidebar) e direciona para a tela de inclusão (figura 33).

Figura 33: Interface de adicionar novo foco de dengue

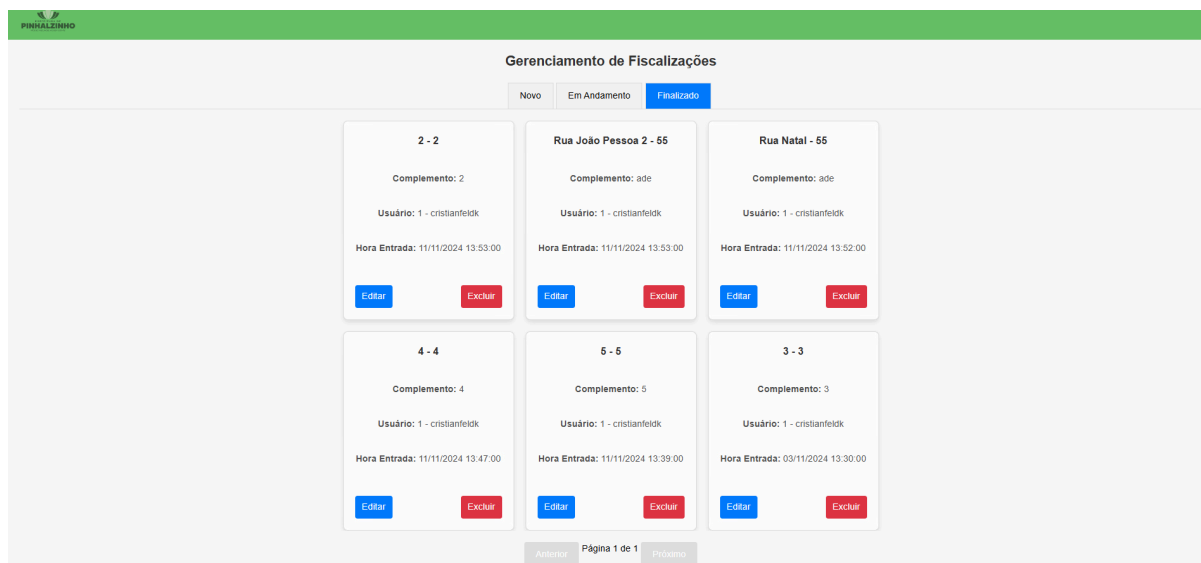


Fonte: Elaborado pelo autor (2024)

Na figura 33, está a representação da interface de “Adicionar Novo Foco de Dengue”, que possui descrição do foco, mapa utilizando a biblioteca do leaflet, que ao marcar o local do foco, é demonstrado latitude e longitude, ao salvar o foco, é demonstrado também na página inicial do software, no mapa abaixo das Estatísticas recentes.

Avaliando as funcionalidades do software, se tornou indispensável por parte da secretária, possuir controle sobre as fiscalizações efetuadas pelos usuários (ACE's), com isso tendo melhor controle sobre a gestão, e também onde são os maiores pontos de fiscalização no município (figura 34).

Figura 34: Interface de controle de fiscalizações

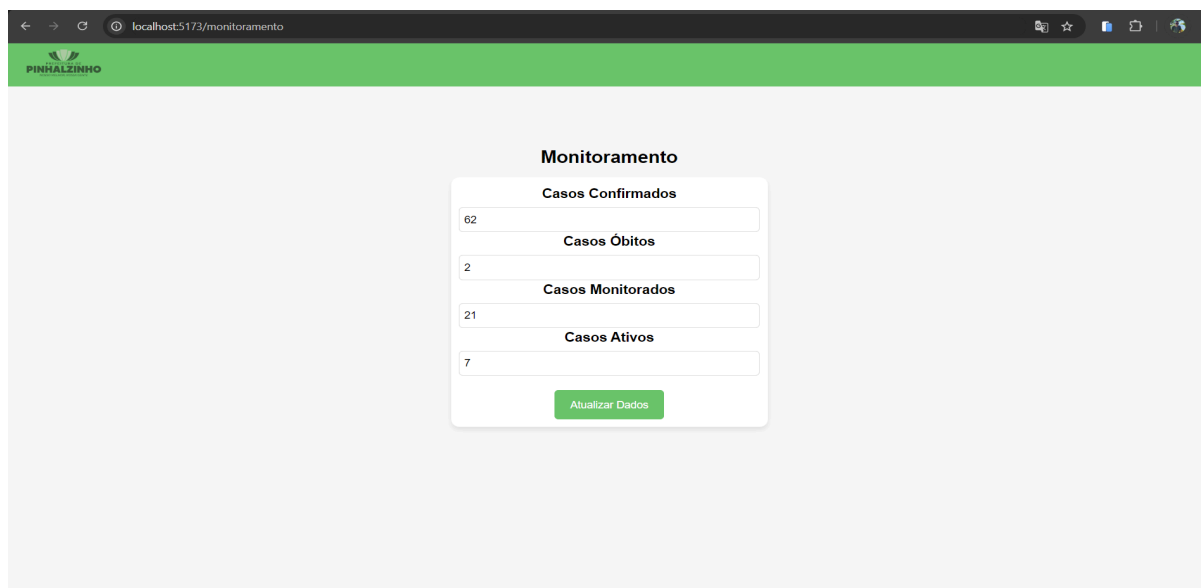


Fonte: Elaborado pelo autor (2024)

Na figura 34, está a representação da interface de controle de fiscalizações, divididas em 3 tabs, Novo, Em Andamento e Finalizado, onde serão apresentados as fiscalizações conforme o campos Data Entrada, contendo também, paginação sobre os registros.

Para que os administradores possam atualizar de forma rápida as estatísticas atuais do município, foi elaborado interface para que possam ser inseridos os dados e atualizado de forma direta na página inicial acessada pela população (figura 35).

Figura 35: Interface de monitoramento

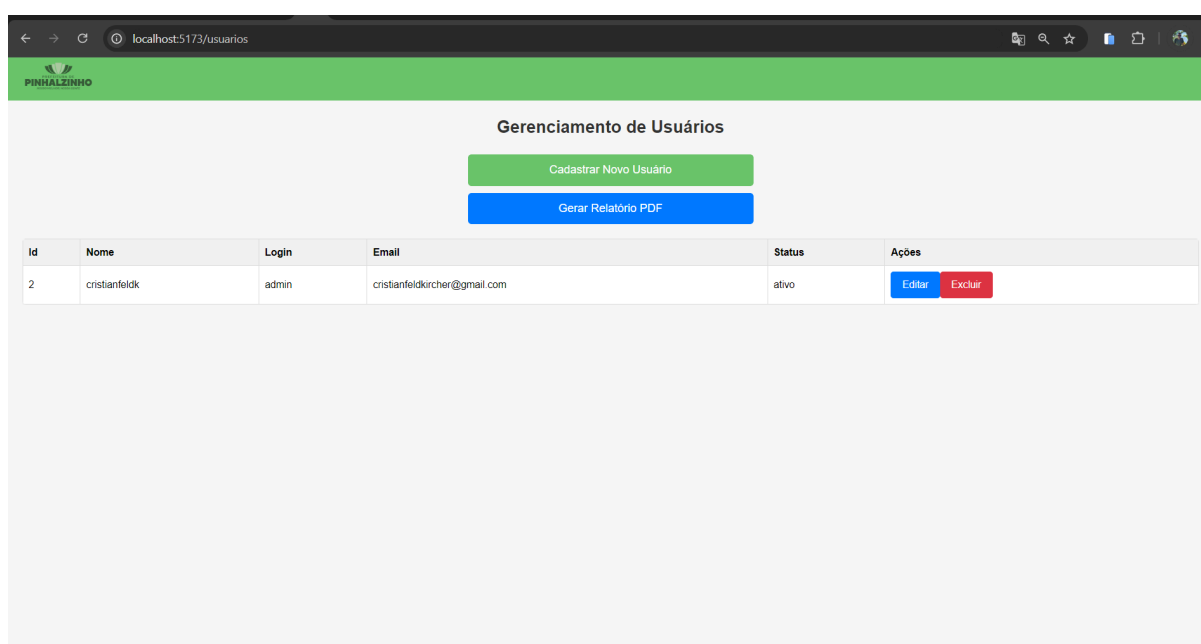


Fonte: Elaborado pelo autor (2024)

Na figura 35, está representada a interface de monitoramento, contendo campos de casos confirmados, óbitos, monitorados e ativos, os números atualizados nessa interface serão apresentados na seção “Estatísticas recentes” na página do software, que está disponível para a comunidade.

Para que os administradores possam ter mais controle sobre os acessos ao software, foi elaborada a tela de controle de usuários, que pode ser acessada somente por usuário com perfil admin, desta forma, qualquer alteração, inclusão ou exclusão deve ser efetuada através de um usuário administrador (figura 36).

Figura 36: Interface de usuários



Fonte: Elaborado pelo autor (2024)

Na figura 36, está representada a interface de usuários, a qual permite editar, cadastrar, excluir e também gerar relatório dos usuários existentes, nessa interface somente usuários com perfil igual **admin** podem acessar.

Através da interface de usuário, é possível acessar a interface de cadastrar novo usuário (figura 37), podendo ser incluso por um outro usuário com perfil **admin**.

Figura 37: Interface de cadastro de usuários

Cadastrar Novo Usuário

Nome

Login

Senha

Email

Status

Perfil

Fonte: Elaborado pelo autor (2024)

Na figura 37, temos a representação da interface para cadastrar novo usuário, sendo inserido campos como nome, login (deve ser único), senha, e-mail, status (ativo ou inativo) e perfil (fiscal ou admin).

Tendo em vista a possível expansão do software para outros municípios, foi efetuado desenvolvido possibilidade de inclusão, exclusão, edição e consulta de municípios (figura 38).

Figura 38: Interface de municípios

Filtrar por UF: Filtrar por Nome:

| Id | Nome | IBGE | UF | Ações |
|------|-----------------|---------|----|--|
| 4312 | Abdon Batista | 4200051 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4313 | Abelardo Luz | 4200101 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4314 | Agrolândia | 4200200 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4315 | Agronômica | 4200309 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4320 | Alfredo Wagner | 4200705 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4321 | Alto Bela Vista | 4200754 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4322 | Anchieta | 4200804 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |
| 4323 | Angelina | 4200903 | SC | <input type="button" value="Editar"/> <input type="button" value="Excluir"/> |

Fonte: Elaborado pelo autor (2024)

Na figura 38, está representada a interface de municípios, contendo paginação de 10 em 10 registros, filtros por UF ou por nome do município, nos registros possuem dois botões, editar e excluir municípios.

Visando poder visualizar informações de fiscalizações de forma mais rápida e também a possibilidade de exportar as informações, foi efetuado a elaboração do Relatório de Fiscalizações (figura 39).

Figura 39: Interface do relatório de Fiscalizações


The image shows a web interface for generating a report of fiscalizations. At the top, there is a green header with the logo of Pinhalzinho. Below the header, the title "Relatório de Fiscalizações" is displayed. The interface includes a "Filtros Gerais" button and a "Listar" button. There are two date pickers: "Data Inicial: 01/11/2024" and "Data Final: 30/11/2024". Below these are four filter input fields: "Quarteirão: Filtrar por Quarteirão", "Sequência: Filtrar por Sequência", "Logradouro: Filtrar por Logradouro", and "Tipo de Imóvel: Filtrar por Tipo de Imóvel". At the bottom, there is a green button "Limpar Filtros" and a green button "Gerar PDF".

Fonte: Elaborado pelo autor (2024)

Na figura 39, está a representação da tela Relatório de Fiscalização, contendo filtros como, quarteirão, sequência, logradouro e tipo de imóvel, na aba “Listar” é possível visualizar vários checkbox para que possa ser selecionado conforme necessidade para que sejam listados os campos, utilizando o jspdf ao clicar em “Gerar PDF” será efetuado o download do relatório conforme filtros.

Ainda visando o relatório de fiscalizações, ao gerar o PDF, é efetuado download do arquivo denominado **relatorio_fiscalizacoes.pdf**, atendendo todos os campos sobre as fiscalizações (figura 40).

Figura 40: PDF do relatório de Fiscalizações



Relatório de Fiscalizações
Secretária Municipal de Saúde - Pinhalzinho SC
Endereço: Avenida Belém, 353 - Centro - Pinhalzinho
Telefone: (49) 3366-6640

Emitido por: Cristian Feldkicher
Data de emissão: 04/11/2024

| Quarteirão | Sequência | Logradouro | Número | Complemento | Hora de Entrada | Tipo de Imóvel |
|------------|-----------|-----------------------|--------|-----------------------|--------------------------------|----------------|
| 56 | 232 | Av Recife | 23 | Sala 2 | 3 de novembro de 2024 às 17:28 | C |
| 55 | 3 | Av Mato Grosso | 2 | Casa amarela | 3 de novembro de 2024 às 17:29 | R |
| 34 | 2 | Rua Rio de Janeiro | 433 | Barracão | 3 de novembro de 2024 às 17:29 | C |
| 55 | 435 | Rua João Pessoa | 55 | (Residencial Viacava) | 3 de novembro de 2024 às 19:56 | C |
| 55 | 2 | Rua São Luiz | 2 | 2 | 3 de novembro de 2024 às 19:56 | R |
| 55 | 3 | Avenida Paraná | 55 | Casa marrom | 4 de novembro de 2024 às 21:03 | R |
| 234 | 4 | NATAL | 433 | (Residencial Viacava) | 4 de novembro de 2024 às 21:05 | C |
| 55 | 7 | Rua Rio Grande do Sul | 4 | Casa 14 | 4 de novembro de 2024 às 21:11 | R |
| 55 | 2 | Rua São Luiz | 2345 | Casa Azul | 3 de novembro de 2024 às 20:27 | R |

Fonte: Elaborado pelo autor (2024)

Na figura 40, está a representação do relatório de fiscalizações, já em PDF, onde possui campos sobre as fiscalizações existentes e também cabeçalho com dados do município, usuário que gerou o relatório e data de emissão do relatório .

Para que a secretária possa obter um visão sobre todos os focos de dengue confirmados entre um determinado período, foi elaborada uma interface para visualização, onde os focos confirmados do período selecionado são visíveis, conforme figura 41.

Figura 41: Interface do relatório de focos



Fonte: Elaborado pelo autor (2024)

Na figura 41, está a representação dos resultados referente ao relatório de focos, o qual demonstra todos os focos do período selecionado, ele busca todos os focos que foram confirmados no período, mesmo que já tenham sido eliminados, trazendo assim uma visão abrangente dos principais bairros afetados.

Atendendo a necessidade de efetuar um controle eficaz sobre as denúncias, foi desenvolvido uma interface de controle de denúncias, essas denúncias são recebidas através do formulário disponível para a comunidade, avaliadas e posteriormente alteradas, conforme figura 42.

Figura 42: Interface de controle de denúncias

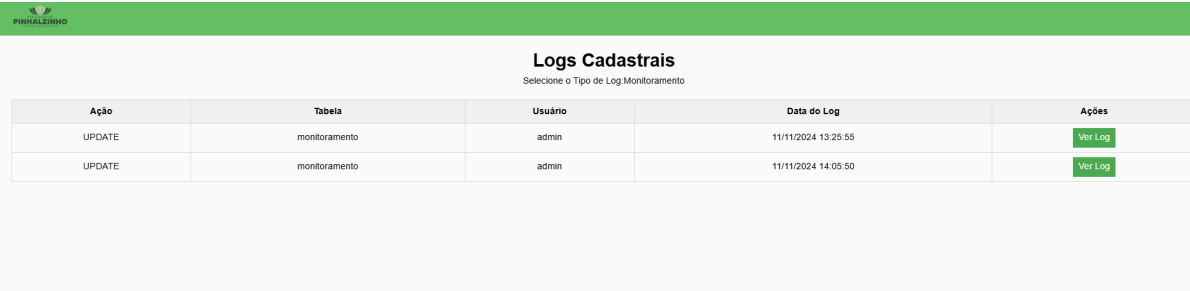
| ID | Nome Denunciante | Email | Telefone | Município | Logradouro | Descrição | Status | Ações |
|----|------------------|-------------------|----------------|-----------|-------------------|---|----------------|---|
| 12 | Anônima | Anônima | Anônima | 4507 | Av Mato Grosso | teste Av Mato Grosso | Não confirmado |     |
| 11 | Anônima | Anônima | Anônima | 4507 | Rua João Pessoa 2 | teste da rua joao pessoa | Não confirmado |     |
| 10 | Anônima | Anônima | Anônima | 4507 | Av Recife | Na avenida recife tem muita dengue | Não confirmado |     |
| 9 | Willian Heinen | willian@gmail.com | (49) 3732-7762 | 4507 | Parque Olaria | alguns pontos de dengue em todo parque da olaria | Confirmado |     |
| 8 | Elvis Machado | elvis@gmail.com | (47) 3513-0938 | 4507 | Av Mato Grosso | Estive passando na avenida e notei muito lixo em um certo ponto | Não confirmado |     |

Fonte: Elaborado pelo autor (2024)

Na figura 42, está representada a interface de controle de denúncias, tendo possibilidade de ser filtrada pelo nome do denunciante, email do denunciante, chave da denúncia e logradouro. A interface também possui os campos que são preenchidos na denúncia em suas respectivas colunas, além disso, na coluna de ações é possível editar a denúncia, visualizar o local aproximado, visualizar a imagem anexada na denúncia e excluir a denúncia.

Para melhor confiabilidade dos dados, foram elaborados registros de logs para que possa ser buscada alguma alteração indevida sobre algum registro (figura 43).

Figura 43: Interface de controle de logs



| Logs Cadastrais | | | | |
|---------------------------------------|---------------|---------|---------------------|-------------------------|
| Selecione o Tipo de Log Monitoramento | | | | |
| Ação | Tabela | Usuário | Data do Log | Ações |
| UPDATE | monitoramento | admin | 11/11/2024 13:25:55 | Ver Log |
| UPDATE | monitoramento | admin | 11/11/2024 14:05:50 | Ver Log |

Fonte: Elaborado pelo autor (2024)

Os logs ficam armazenados durante 90 dias, após isso são apagados automaticamente com um gatilho criado no banco de dados, os logs visualizados ao clicar no botão “Ver Log” que demonstra o registro antigo e o registro novo (figura 43).

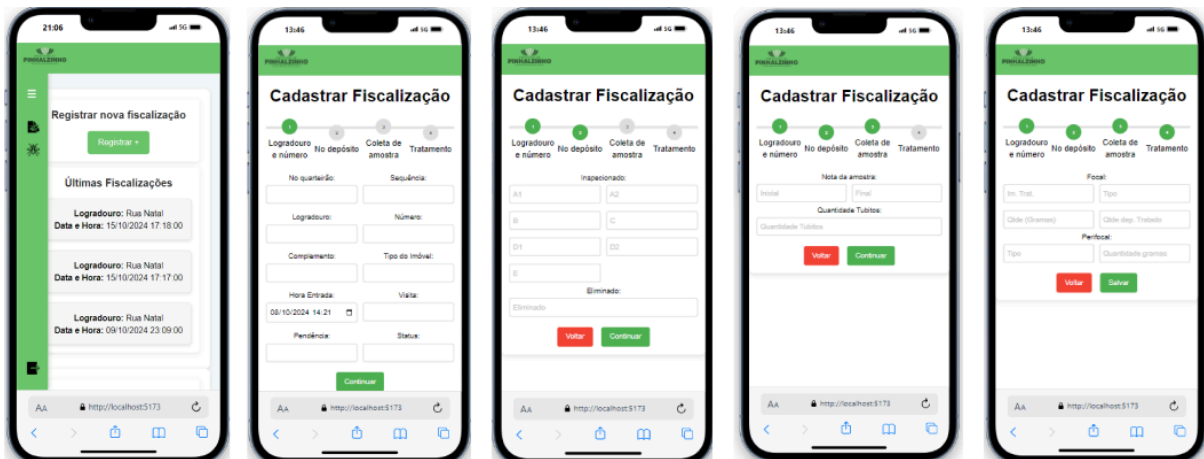
6.3.4.3 Interfaces para ACE's (mobile)

Nesta etapa será descrito e representado em imagens todas as interfaces sobre o uso dos Agentes de Combate à Dengue (ACE 's). As interfaces podem ser acessadas tanto por mobile quanto web.

Para uso dos ACE's, foi elaborado um controle de acesso, tendo como disponível, opções de cadastro de logradouros, cadastro de municípios, cadastro de fiscalização, cadastro de denúncias, inclusão de focos e remoção de focos.

Na figura abaixo (figura 44), elaboramos o cadastro de fiscalização, esse cadastro permite com que o agente deixe de usar o documento físico (papel), proporcionando um preenchimento de informações de forma mais eficiente e menos desgastante.

Figura 44: Interfaces para Cadastrar Fiscalização

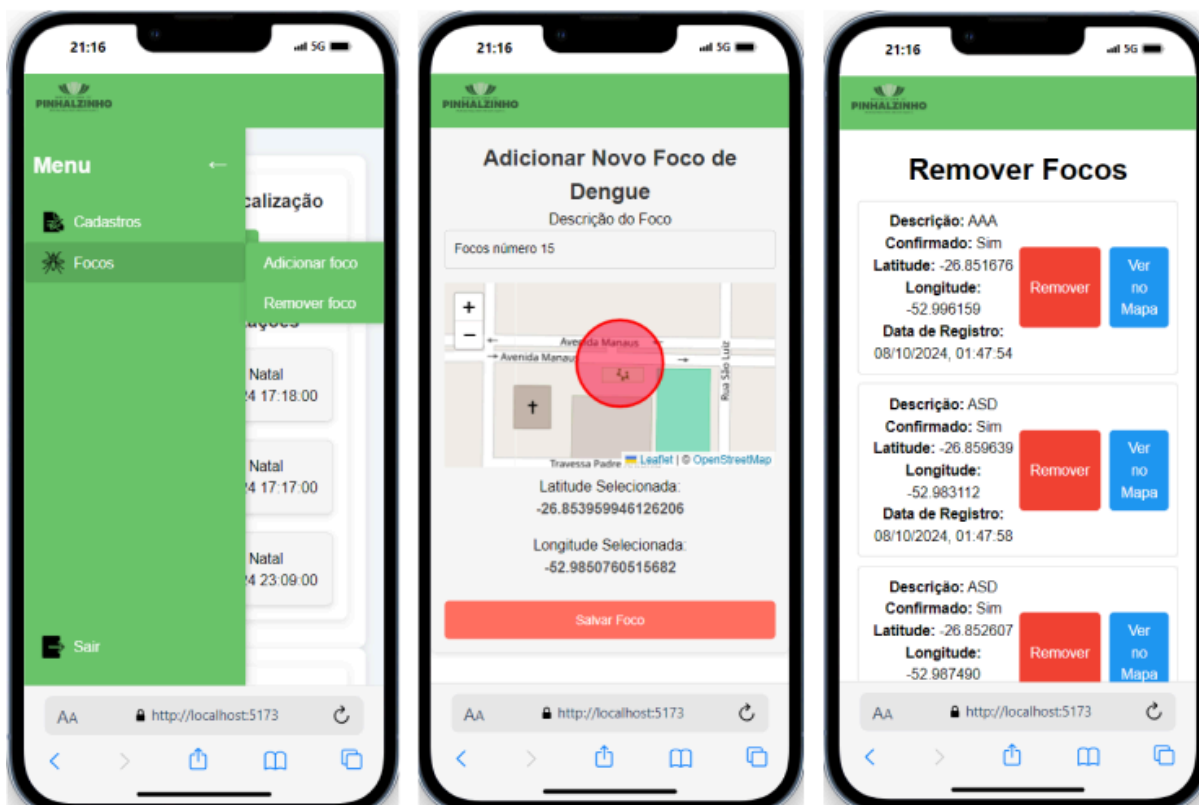


Fonte: Elaborado pelo autor (2024)

A figura 44, possui como objetivo, ser utilizada pelos agentes em visitas a residências e comércios, a fiscalização é efetuada em 4 etapas, na primeira etapa é informado dados mais importantes, **quarteirão, seqüência, logradouro, número, complemento, tipo do imóvel, hora de entrada, tipo de visita, pendência e status da fiscalização**. Já na segunda etapa, contém duas seções de dados, no inspeccionado é referente aos campos **A1, A2, B, C, D1, D2 e E**, na segunda seção da etapa número 2, é somente o campo **eliminado**. Na etapa número 3, são informados nota de **Amostra Inicial**, nota de **Amostra Final** e **Quantidade de Tubitos**. Na seção de número 4, refere-se à parte de tratamento, Focal tendo os campos **Im. Trat, Tipo, Qtde (Gramas), Qtde dep. Tratado** e Perifocal com os campos, **Tipo e Quantidade gramas**.

Para atender a necessidade de poder estar confirmando um novo foco de dengue no ato de estar efetuando uma fiscalização, o agente de fiscalização pode estar acessando pelo menu, a opção de adicionar e de remover focos de dengue conforme figura 45.

Figura 45: Interfaces para inclusão e remoção de focos



Fonte: Elaborado pelo autor (2024)

Na figura 45, está representada a interface de adição e de remoção de focos de dengue, ela possibilita adicionar um novo foco que será apresentado como confirmado, posteriormente irá ser demonstrado na página inicial acessada pela população, possibilidade também de remoção dos focos, removendo diretamente da página inicial acessada pela população.

7 CONSIDERAÇÕES FINAIS

Diante dos significativos aumentos de focos de dengue no município de Pinhalzinho SC, e também em todo Brasil, a implementação de um software de controle de focos, fiscalização e denúncias se torna extremamente desafiador, tendo em vista todo o gerenciamento, desde receber denúncias, repassar resultados para a comunidade e também obter gestão sobre fiscalizações. Visto isso, o atual trabalho apresenta um estudo do cenário da atualidade dos casos de Dengue no Brasil, além do software desenvolvido para o município de Pinhalzinho SC.

As tecnologias utilizadas no atual software, como, PostgreSQL, Vue.js, Node.js e outras, buscam de forma intuitiva e eficaz proporcionar para a secretária de saúde, uma versão aprimorada do processo de gestão.

Além disso, o desenvolvimento das API's possibilitam uma integração eficiente com outros sistemas ou ferramentas utilizadas na área da saúde, dessa forma permitindo troca de informações entre sistemas, essa integração fica ainda mais prática com utilização das documentações onde estão todos os endpoints da API.

O software desenvolvido, atende os principais objetivos estabelecidos referente a gestão sobre focos de dengue em todo município de Pinhalzinho, possibilitando que seja feito de forma intuitiva, controle de denúncias, controle de fiscalizações e focos de dengue além de demonstrar estatísticas para a comunidade.

7.1 TRABALHOS FUTURO

O levantamento de dados possibilitou diversos cenários de melhorias presentes para o sucesso do projeto em questão, como as seguintes:

- Desenvolver possibilidade de publicação de boletins mensais da secretaria.
- Implementação de firewalls para controlar e filtrar o tráfego de entrada e saída no servidor onde o sistema está hospedado, garantindo proteção contra acessos não autorizados.
- Desenvolver envio de denúncia por e-mail ao denunciante, para que no ato da criação da denúncia, seja recebido uma confirmação da criação.
- Desenvolver uma interface para que a comunidade possa pesquisar denúncias.

REFERÊNCIAS BIBLIOGRÁFICAS:

COUTINHO, Thiago. Express.js: entenda as características desse framework backend!. Disponível em: <https://voitto.com.br/blog/artigo/o-que-e-expressjs>. Acesso em: 07 junho 2024.

DEVMEDIA. O que é UML e Diagramas de Caso de Uso: Introdução Prática à UML. Disponível em: <https://www.devmedia.com.br/o-que-e-uml>. Acesso em: 15 outubro, 2024.

DIVE. ORIENTAÇÕES TÉCNICAS PARA PESSOAL DE CAMPO. Disponível em: <https://dive.sc.gov.br/phocadownload/doencas-agrivos>. Acesso em: 26 maio 2024.

FEITOSA, Juliana Aparecida Corrêa Nunes. Reflexão sobre a participação da comunidade no combate a dengue. Disponível em: <https://www.nescon.medicina.ufmg.br/biblioteca>. Acesso em: 03 junho 2024.

FLANAGAN, David. JavaScript: The Definitive Guide. Disponível em: <https://pepa.holla.cz/wp-content/uploads>. Acesso em: 07 junho 2024.

FOX IOT. Node.js: curiosidades e tendências. Disponível em: <https://foxiot.com.br/es/node-js-curiosidades-e-tendencias/>. Acesso em: 07 junho 2024.

FRANÇA, Lays Santos, et al. Desafios para o controle e prevenção do mosquito aedes aegypti. Disponível em: <https://doi.org/10.5205/1981-8963>. Acesso em: 09 junho 2024.

GIL, Carlos. Métodos e Técnicas de Pesquisa Social. 7. ed. São Paulo: Grupo GEN, 2019.

GitHub. (2008). GitHub. Disponível em: <https://github.com>. Acesso em: 10 maio 2024.

GOMES, K. W. L. et al. Organização do processo de trabalho no manejo da dengue em uma capital do Nordeste. Disponível em: <http://www.scielo.br/pdf>. Acesso em: 09 junho 2014.

GOV. MS. Dicas simples podem ajudar na prevenção contra a dengue na sua residência. Disponível em: <https://www.saude.ms.gov.br/dicas-simples> Acesso em: 26 maio 2024.

GOV. PR. População tem papel importante no combate à dengue. Disponível em: <https://www.saude.pr.gov.br/Noticia>. Acesso em: 03 junho 2024.

GUIMARÃES, Willyan. O que é uma interface em programação?. Disponível em: <https://medium.com/experiencecode/o-que-%C3%A9-uma-interface-em-programacao>. Acesso em: 31 outubro, 2024.

IBGE. Pesquisa de informações básicas municipais. Disponível em: <http://cidades.ibge.gov.br/brasil/sc/pinhalzinho/panorama>. Acesso em: 27 maio 2024.

IBM. Diagramas de Classes. Disponível em: <https://www.ibm.com/diagramas-de-classes>. Acesso em: 15 outubro, 2024.

IBM. Swagger. Disponível em: <https://www.ibm.com/docs>. Acesso em: 07 junho 2024.

INSTITUTO OSWALDO CRUZ. Dengue: uma visão histórica no Brasil. Disponível em: <https://www.ioc.fiocruz.br/dengue/textos/longatraje.html>. Acesso em: 23 maio 2024.

INSTITUTO RENÉ RACHOU FIOCRUZ MINAS. Dengue e sua história. Disponível em: <https://www.conquista.mg.gov.br/noticia>. Acesso em: 23 maio 2024.

LUCIDCHART. Diagrama de caso de uso UML: O que é, como fazer e exemplos.. Disponível em: <https://www.lucidchart.com/pages/pt/diagrama-de-caso-de-uso-uml>. Acesso em: 16 outubro 2024.

LUTINSKI, Junir Antonio, et. al. Environmental factors associated to dengue fever occurrence in the Chapecó municipality, Santa Catarina State. Disponível em: <https://dx.doi.org/>. Acesso em: 31 maio 2024.

MARTINS, Antonio. Documentação de APIs: você conhece o Swagger?. Disponível em: <https://medium.com/inside-swagger>. Acesso em: 07 junho 2024.

MATIAS-PEREIRA, José. Manual de Metodologia da Pesquisa Científica. 4. ed. São Paulo: Atlas, 2016.

MERKEL, Diogo. Docker: containers Linux leves para desenvolvimento e implantação consistentes. In: Merkel, Diogo. Docker: containers Linux leves para desenvolvimento e implantação consistentes. Linux Journal, 2014(239), p.

Ministério da Saúde. Secretaria de Vigilância em Saúde. Coordenação-Geral de desenvolvimento da Epidemiologia em Serviços. Guia de vigilância em saúde. Disponível em: <http://portal.arquivos.saude.gov.br>. Acesso em 26 maio 2024.

MINISTÉRIO DA SAÚDE: INFORME TÉCNICO: DENGUE. Disponível em: <https://www.saude.sp.gov.br/resources/cve-centro-de-vigilancia>. Acesso em: 05 maio, 2024.

MOSHE, Lucca. Prototipar: o que é, como fazer e sua importância? Disponível em: <https://escolakoru.com.br/blog/prototipar-o-que-e-como-fazer/>. Acesso em: 03 outubro, 2024.

OPUS. Node.js – O que é, como funciona e quais as vantagens. Disponível em: <https://www.opus-software.com.br/insights/node-js/>. Acesso em: 07 junho, 2024.

PEDRO, Wagner. O que é UML?. Disponível em: <https://tecnoblog.net/o-que-e-uml/>. Acesso em: 15 outubro, 2024.

PEREIRA, Arlindo. Mapas na web com Leaflet.js. Disponível em: <https://medium.com/leaflet.js>. Acesso em: 10 outubro, 2024.

PERES, A. C. Aedes: ampliando o foco. Comunicação e Saúde Revista Radis. Escola Nacional de Saúde Pública - ENSP Fiocruz, n. 161, fev., 2016.

PINHO. Diego. Gerando arquivos PDF com JavaScript. Disponível em: <https://medium.com/code-prestige/gerando-arquivos-pdf-com-javascript>. . Acesso em: 10 outubro, 2024.

PREFEITURA DE CURIPA. Governo municipal desenvolve projeto de combate ao Aedes aegypti. Disponível em: <http://cupira.pe.gov.br/2016/02>. Acesso em: 24 maio, 2024.

SCHMIDT, Mario. Node.js 6.x Blueprints: Develop, test, and deploy robust web applications using the Express framework and Node.js. In: SCHMIDT, Mario. Packt Publishing Ltd, 2016.

Secretaria de Vigilância em Saúde. O Agente Comunitário de Saúde no controle da dengue. Disponível em: <https://bvsms.saude.gov.br>. Acesso em: 14 abril 2024.

SILVA, Liliam. B. Comunicação sazonal sobre a dengue em grupos socioeducativos na atenção primária à saúde. Disponível em: <http://www.scielo.br/scielo.php>. Acesso em: 09 junho 2024.

TOTVS. Front end: O que é, como funciona e qual a importância. Disponível em: <https://www.totvs.com/blog/developers/front-end/>. Acesso em: 31 outubro, 2024.

UFMA et. al. Segunda do Português: "Aedes aegypti". Disponível em: <https://portalpadrao.ufma.br/site/servicos-a-comunidade> Acesso em: 15 abril, 2024.

VERSIANI, Rafael. O que é o Postman?. Disponível em: <https://enotas.com.br/postman>. Acesso em: 07 junho 2024.

Vue.js. Vue.js Documentation. Disponível em: <https://vuejs.org/v2/guide/>. Acesso em: 09 maio 2024.