Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Computação e Sistemas

Desafios da adoção: construção de um sistema para facilitar o processo de adoção de animais

Arthur Enrique Ribeiro Silva

TRABALHO DE CONCLUSÃO DE CURSO

ORIENTAÇÃO:
Igor Muzetti Pereira

Abril, 2025
João Monlevade–MG

Arthur Enrique Ribeiro Silva

Desafios da adoção: construção de um sistema para facilitar o processo de adoção de animais

Orientador: Igor Muzetti Pereira

Monografia apresentada ao curso de Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina "Trabalho de Conclusão de Curso II".

Universidade Federal de Ouro Preto

João Monlevade

Abril de 2025

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

S586d Silva, Arthur Enrique Ribeiro.
Desafios da adoção [manuscrito]: construção de um sistema para facilitar o processo de adoção de animais. / Arthur Enrique Ribeiro Silva. - 2025.
91 f.: il.: color..

Orientador: Prof. Dr. Igor Muzetti Pereira.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Sistemas de Informação .

1. Aplicações Web - Desenvolvimento. 2. Desenvolvimento ágil de software. 3. Engenharia de software. 4. Software de aplicação - Desenvolvimento. 5. Software livre. I. Pereira, Igor Muzetti. II. Universidade Federal de Ouro Preto. III. Título.

CDU 004.41

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTACAO E SISTEMAS

FOLHA DE APROVAÇÃO

Arthur Enrique Ribeiro Silva

Desafios da adoção: Construção de um sistema para facilitar o processo de adoção de animais

Monografia apresentada ao Curso de Sistemas de Informação da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação

Aprovada em 29 de Abril de 2025

Membros da banca

Dr. Igor Muzetti Pereira - Orientador (Universidade Federal de Ouro Preto)
Dra. Lucinéia Maia (Universidade Federal de Ouro Preto)
Dr. Fernando de Oliveira (Universidade Federal de Ouro Preto)

Igor Muzetti Pereira, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 29/04/2025

Documento assinado eletronicamente por **Igor Muzetti Pereira**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 29/04/2025, às 10:24, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0902405** e o código CRC **001434D1**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: (31)3808-0819 - www.ufop.br

Este trabalho é dedicado à minha família

Agradecimentos

Agradeço imensamente aos meus pais e familiares, pelo apoio incondicional e pela força que sempre me deram ao longo dessa jornada. Um agradecimento especial à minha mãe, que foi a pessoa que mais me deu suporte e motivação ao longo de toda a minha trajetória acadêmica. Sem o amor e a dedicação dela, nada disso seria possível.

Também sou grato aos amigos que fiz durante minha jornada na UFOP. As amizades construídas ao longo do curso tornaram essa caminhada mais leve e prazerosa. A ideia deste trabalho, inclusive, surgiu de uma conversa casual com eles, e sou muito grato por essa troca de ideias e apoio constante.

Por fim, agradeço ao Prof. Dr. Igor Muzetti Pereira, que me guiou no desenvolvimento deste trabalho. Suas orientações, discussões e reuniões semanais foram fundamentais para o aprimoramento deste projeto, contribuindo imensamente para o resultado final. Agradeço por toda a paciência, dedicação e sabedoria compartilhada ao longo desse processo.

Minha mais profunda gratidão a todos vocês que me apoiaram durante a realização deste trabalho.

"Science is more than a body of knowledge; it is a way of thinking."

— Carl Sagan (1934 – 1996),
in: The Demon-Haunted World: Science as a Candle in the Dark.

Resumo

Este trabalho apresenta o desenvolvimento da plataforma AdoteFácil, um sistema *web* projetado para simplificar e incentivar o processo de adoção de animais domésticos. O objetivo principal foi criar uma plataforma digital simplificada e gratuita para facilitar a conexão entre doadores e adotantes, especialmente em municípios menores e regiões com pouca infraestrutura voltada para proteção animal. A metodologia aplicada ao desenvolvimento foi baseada em práticas ágeis adaptadas à realidade de um projeto individual, utilizando Kanban para acompanhamento de tarefas. A plataforma foi desenvolvida com tecnologias como TypeScript, React, Next.js, Node.js e PostgreSQL, além da utilização da ferramenta Docker para conteinerização da aplicação. Como resultado, foi produzido um *software* totalmente funcional, com cobertura de testes unitários completa e disponibilizado publicamente como um projeto de *software* livre no GitHub.

Palavras-chave: Engenharia de *software*, Metodologias ágeis, Adoção de animais, Desenvolvimento *web*, *Software* livre.

Abstract

This work presents the development of the AdoteFácil platform, a web system designed to simplify and encourage the adoption of domestic animals. The main goal was to create a simplified and free digital platform to facilitate connections between donors and adopters, particularly in smaller municipalities and regions with limited infrastructure for animal protection. The development methodology was based on agile practices adapted to an individual project context, using Kanban to manage tasks. The platform was built with technologies such as TypeScript, React, Next.js, Node.js, and PostgreSQL, and containerized using Docker. As a result, a fully functional software with comprehensive unit test coverage was produced and made publicly available as free software on GitHub.

Key-words: *Software engineering, Agile methodologies, Animal adoption, Web development, Free software.*

Lista de ilustrações

Figura 1 – Página de *login* 16
Figura 2 – Menu principal 16
Figura 3 – Página de publicações de animais 17
Figura 4 – Página de adicionar publicação 17
Figura 5 – *Print* da plataforma Adote Petz 18
Figura 6 – *Print* da plataforma Amigo Não Se Compra filtrando os animais pela cidade de João Monlevade 20
Figura 7 – *Print* da plataforma Amigo Não Se Compra filtrando os animais pela cidade de Belo Horizonte 21
Figura 8 – *Print* da seção de animais disponíveis para adoção no *site* da prefeitura de João Monlevade 22
Figura 9 – Logo das tecnologias utilizadas no projeto 31
Figura 10 – *Print* do Projeto Adote Fácil no Jira 32
Figura 11 – Logotipo da plataforma 34
Figura 12 – Sugestões de logotipo criadas pela ferramenta Gemini 35
Figura 13 – Processo de criação do logotipo no Figma 35
Figura 14 – Protótipo da tela principal 36
Figura 15 – Protótipo da tela de informações do animal 36
Figura 16 – Lista de *commits* no repositório do GitHub 37
Figura 17 – Relatório de cobertura de testes no terminal 38
Figura 18 – Relatório de cobertura de testes no arquivo criado pelo Jest 39
Figura 19 – Tela de cadastro de usuário *web* 41
Figura 20 – Tela de *login* de usuário *web* 42
Figura 21 – Tela de cadastro de usuário *mobile* 42
Figura 22 – Tela de *login* de usuário *mobile* 42
Figura 23 – Tela de animais disponíveis *web* 43
Figura 24 – Tela de animais disponíveis *web* vazia 43
Figura 25 – Tela de animais disponíveis *mobile* 44
Figura 26 – Tela de animais disponíveis *mobile* vazia 44
Figura 27 – Tela de detalhes do animal *web* 44
Figura 28 – Tela de editar dados do usuário *web* 45
Figura 29 – Tela de detalhes do animal *mobile* 45
Figura 30 – Tela de editar dados do usuário *mobile* 45
Figura 31 – Tela de cadastro de animal para adoção *web* 46
Figura 32 – Tela de meus animais *web* 46
Figura 33 – Tela de cadastro de animal para adoção *mobile* 47

Figura 34 – Tela de meus animais *mobile* 47
Figura 35 – Tela de minhas conversas *web* 48
Figura 36 – Tela de *chat* entre usuários *web* 48
Figura 37 – Tela de minhas conversas *mobile* 49
Figura 38 – Tela de *chat* entre usuários *mobile* 49
Figura 39 – *Print* dos *logs* no terminal ao executar os contêineres do sistema no computador do LECoMP 51
Figura 40 – *Print* da tela de *login* do sistema executando no computador do LECoMP 52
Figura 41 – Protótipo da tela de cadastro 87
Figura 42 – Protótipo da tela de *login* 87
Figura 43 – Protótipo do cadastro *mobile* 88
Figura 44 – Protótipo do *login mobile* 88
Figura 45 – Protótipo da tela principal sem animais 88
Figura 46 – Protótipo do menu lateral 88
Figura 47 – Protótipo da tela de disponibilizar animal 89
Figura 48 – Protótipo da tela de meus animais 89
Figura 49 – Protótipo da tela de *chat* 89
Figura 50 – Protótipo da tela de conversas 89

Lista de abreviaturas e siglas

ACID Atomicidade, Consistência, Isolamento e Durabilidade

API *Application Programming Interface*

HTTP *Hypertext Transfer Protocol*

IA Inteligência Artificial

ICEA Instituto de Ciências Exatas e Aplicadas

LECoMP Laboratório de Estudos em Computação do Médio Piracicaba

ONGs Organizações Não Governamentais

ORM *Object-Relational Mapping*

SEO *Search Engine Optimization*

SQL *Structured Query Language*

SGBDR Sistema de Gerenciamento de Banco de Dados Relacional

SSR *Server Side Rendering*

TI Tecnologia da Informação

WIP *Work In Progress*

Sumário

1 INTRODUÇÃO 13
1.1 Identificação do problema 14
1.2 Objetivos 14
1.3 Organização do trabalho 15

2 REVISÃO BIBLIOGRÁFICA 16
2.1 Trabalhos correlatos 16
2.1.1 Aplicativo como plataforma de integração entre sociedade e animais domésticos 16
2.1.2 Programa Adote Petz 17
2.1.3 Plataforma Amigo Não se Compra 18
2.1.4 Espaço no *site* da Prefeitura de João Monlevade para adoção de animais resgatados 21
2.2 Abordagens e ferramentas no desenvolvimento do *software* 23
2.2.1 *Software* livre 23
2.2.2 Metodologias ágeis 24
2.2.3 Engenharia de requisitos 24
2.2.4 Histórias de usuário 25
2.2.5 Kanban 26
2.2.6 Testes unitários 27

3 DESENVOLVIMENTO 28
3.1 Documento de visão 28
3.2 Documento de requisitos 28
3.3 Definição de tecnologias para o desenvolvimento 29
3.4 Histórias de Usuário 31
3.5 Prototipação do sistema 34
3.6 Desenvolvimento do Sistema 36
3.7 Implementação dos testes unitários 38
3.8 Empacotamento do Sistema 39
3.9 Licença de *Software* Livre GNU/GPLv3 39

4 RESULTADOS 41
4.1 O sistema 41
4.2 Repositório no GitHub 49
4.3 Aplicação do tutorial para execução do sistema em um computador do LECoMP 50

5 CONSIDERAÇÕES FINAIS 53
5.1 Conclusão 53
5.2 Trabalhos Futuros 54

REFERÊNCIAS 55

APÊNDICES 57

APÊNDICE A – MATERIAIS ELABORADOS PELO AUTOR . . . 58
A.1 Documento de visão 58
A.2 Documento de requisitos 69
A.3 Histórias de usuário 79
A.3.1 Configuração inicial do projeto 79
A.3.2 Cadastro de clientes 79
A.3.3 Login de clientes 80
A.3.4 Atualização de dados do cliente 82
A.3.5 Disponibilização de animal para adoção 83
A.3.6 Listagem de animais disponíveis para adoção 84
A.3.7 Confirmação de adoção 84
A.3.8 *Chat* entre adotante e doador 85
A.3.9 Empacotamento do sistema 86
A.4 Protótipos 86

1 Introdução

A adoção de animais domésticos é uma prática essencial para reduzir o número de cães e gatos em situação de abandono, promovendo o bem-estar animal e incentivando a posse responsável. No Brasil, milhões de cães e gatos vivem nas ruas, expostos a diversos riscos, como fome, doenças e maus-tratos. Durante a pandemia de COVID-19, embora tenha havido um aumento de cerca de 30% na adoção de cães e gatos, o número de abandonos também cresceu significativamente (EXAME, 2021).

De acordo com Rosangela Gebara, gerente de projetos da Ampara Animal [1], em entrevista para a Exame: "o índice de abandono e de recolhimento de animais aumentou, em média, 61% entre julho de 2020 até o terceiro trimestre de 2021". A crise econômica e social exacerbada pela pandemia também contribuiu para esse aumento, levando muitos tutores a abandonarem seus animais devido a mudanças de casa, separações, perda de emprego e incapacidade financeira de mantê-los (EXAME, 2021).

Atualmente, o processo de adoção ocorre por meio de diferentes canais. Em algumas cidades, feiras de adoção organizadas por Organizações Não Governamentais (ONGs), clínicas veterinárias e *pet shops* são uma das principais formas de promover a adoção responsável. Essas feiras permitem que os adotantes conheçam os animais pessoalmente, recebam orientações sobre cuidados e passem por triagens para garantir que possuem condições adequadas para receber um novo *pet*. Além disso, redes sociais e *sites* especializados também desempenham um papel significativo na adoção, permitindo que doadores publiquem fotos e descrições de animais disponíveis, conectando-se com potenciais adotantes rapidamente.

Nos últimos anos, surgiram também plataformas *online* dedicadas exclusivamente à adoção de animais, funcionando como catálogos digitais onde doadores podem cadastrar animais e adotantes podem buscar por características específicas, como porte, idade, sexo e temperamento. O programa Adote Petz[2], por exemplo, foi criado para estreitar laços entre pessoas que têm o sonho de adotar um *pet* e ONGs e protetores parceiros, incentivando a adoção e conscientizando sobre a posse responsável (ADOTE PETZ, 2025). Algumas dessas plataformas contam com sistemas de triagem e acompanhamento pós-adoção, garantindo que os animais sejam entregues a lares preparados para recebê-los. No entanto, um dos principais desafios dessas ferramentas é a dificuldade de atender de maneira eficaz regiões menos populosas e cidades de menor porte, onde a infraestrutura e o número de animais disponíveis para adoção são limitados.

[1] https://institutoamparanimal.org.br/
[2] https://www.adotepetz.com.br/

1.1 Identificação do problema

Apesar das diversas iniciativas existentes, o processo de adoção ainda enfrenta desafios significativos, especialmente em cidades menores e regiões com pouca infraestrutura para a proteção animal. Enquanto em grandes centros urbanos há maior acesso a eventos e campanhas de adoção, em municípios menores a principal alternativa continua sendo a doação direta, feita por meio de redes de contatos pessoais, redes sociais ou grupos de mensagens. A falta de regulamentação e controle sobre esses processos pode resultar em adoções impulsivas, nas quais animais são entregues a pessoas que não estão preparadas para cuidar deles, aumentando os riscos de devolução ou novo abandono.

Outro grande obstáculo é a falta de apoio institucional. Embora existam ONGs e grupos de protetores independentes dedicados à causa animal, muitos enfrentam dificuldades financeiras e logísticas para manter os resgates e encaminhar os animais para adoção. O poder público, por sua vez, geralmente não dispõe de programas estruturados para o incentivo à adoção responsável, limitando-se a ações pontuais ou delegando essa responsabilidade à sociedade civil. Além disso, a superlotação de abrigos e a escassez de lares temporários fazem com que muitos animais resgatados permaneçam em condições inadequadas por longos períodos antes de serem adotados. Em Minas Gerais, por exemplo, o Governo Estadual tem investido na criação de políticas públicas para combater o abandono de animais, como campanhas de castração e estímulo à adoção responsável (AGÊNCIA MINAS, 2024). Entretanto, essas iniciativas ainda são limitadas e insuficientes para suprir a alta demanda por soluções eficazes.

Diante desse cenário, torna-se evidente a necessidade de soluções que facilitem a conexão entre doadores e adotantes, promovendo um processo mais eficiente e seguro, especialmente para municípios e localidades menos populosas. A criação de plataformas digitais modernas e bem estruturadas é essencial, pois pode desempenhar um papel crucial ao oferecer uma alternativa viável para suprir as lacunas existentes no atual modelo de adoção de animais no Brasil, garantindo que regiões com menor densidade populacional também tenham acesso a opções de adoção e apoio de qualidade.

1.2 Objetivos

O presente trabalho tem como objetivo principal o desenvolvimento e a implementação da plataforma *web* **AdoteFácil**, um *software* livre projetado para facilitar o processo de adoção de animais, conectando doadores e adotantes de maneira simples e eficiente. A proposta visa suprir lacunas existentes nas atuais iniciativas de adoção, proporcionando um meio digital intuitivo e funcional que auxilie na redução do número de animais abandonados.

Sendo assim, os principais objetivos específicos deste trabalho são:

- Desenvolver uma plataforma *web* responsiva que permita o cadastro de animais para adoção, a busca por *pets* disponíveis com filtros avançados e um sistema de comunicação entre adotantes e doadores;
- Aplicar métodos de engenharia de *software* no desenvolvimento da plataforma, incluindo levantamento e documentação de requisitos, prototipação, gerenciamento ágil com Kanban e implementação de testes unitários;
- Disponibilizar a plataforma como um projeto de *software* livre, permitindo que ONGs, prefeituras e outras instituições possam adaptar e implantar a solução conforme suas necessidades, fornecendo também documentação detalhada para auxiliar na implantação e manutenção do sistema.

Vale ressaltar que, embora o desenvolvimento da plataforma seja completo, sua implantação não faz parte do escopo do projeto. O foco principal deste trabalho é criar e disponibilizar a plataforma como um projeto de *software* livre, sem a responsabilidade de gerenciar o processo de implantação ou a operação contínua do *software* em um ambiente real.

1.3 Organização do trabalho

O restante deste trabalho é organizado como se segue. O Capítulo 2 apresenta os trabalhos correlatos e as abordagens e ferramentas utilizadas no desenvolvimento do *software*. O Capítulo 3 apresenta uma descrição do processo de desenvolvimento do sistema. O Capítulo 4 apresenta os resultados obtidos ao final do desenvolvimento, demonstrando diversas telas do sistema. O Capítulo 5 apresenta as conclusões obtidas após o desenvolvimento e implementação do trabalho. Por fim, o Apêndice A apresenta os materiais elaborados pelo autor ao longo do desenvolvimento do projeto.

2 Revisão bibliográfica

2.1 Trabalhos correlatos

2.1.1 Aplicativo como plataforma de integração entre sociedade e animais domésticos

Com o avanço das tecnologias digitais, diversas plataformas têm sido desenvolvidas para facilitar o processo de adoção de animais, promovendo uma maior integração entre a sociedade e os *pets* que necessitam de um lar. O estudo de Riva et al. (2022) apresenta o **PetLover.Find**, um aplicativo móvel desenvolvido com o objetivo de auxiliar na divulgação e busca por animais para adoção, além de permitir o registro de *pets* perdidos e encontrados na cidade de Canoas, Rio Grande do Sul.

O aplicativo permite que os usuários cadastrem animais disponíveis para adoção, visualizem publicações com informações detalhadas, como idade, porte e raça, e acessem os contatos dos responsáveis pela doação. Além disso, ele também permite o registro de animais que foram encontrados sem tutor ou que estão desaparecidos, ampliando as possibilidades de reconexão entre donos e *pets*. As Figuras 1, 2, 3 e 4 demonstram os protótipos da interface do aplicativo.

Figura 1 – Página de *login*

Fonte: Riva et al. (2022)

Figura 2 – Menu principal

Fonte: Riva et al. (2022)

Figura 3 – Página de publicações de animais

Fonte: Riva et al. (2022)

Figura 4 – Página de adicionar publicação

Fonte: Riva et al. (2022)

O estudo destaca a importância de plataformas digitais para facilitar a adoção responsável e a busca por animais desaparecidos. A iniciativa serve como referência para o desenvolvimento de soluções tecnológicas voltadas à causa animal, promovendo maior visibilidade para *pets* que necessitam de um lar e incentivando práticas mais organizadas e eficazes na adoção de animais (RIVA et al., 2022).

Entretanto, embora a plataforma ofereça uma solução para aprimorar o processo de adoção, ela não disponibiliza um meio de comunicação direto entre adotantes e doadores. Em vez disso, a plataforma fornece apenas informações de contato do doador, como telefone, endereço e outros dados relevantes.

2.1.2 Programa Adote Petz

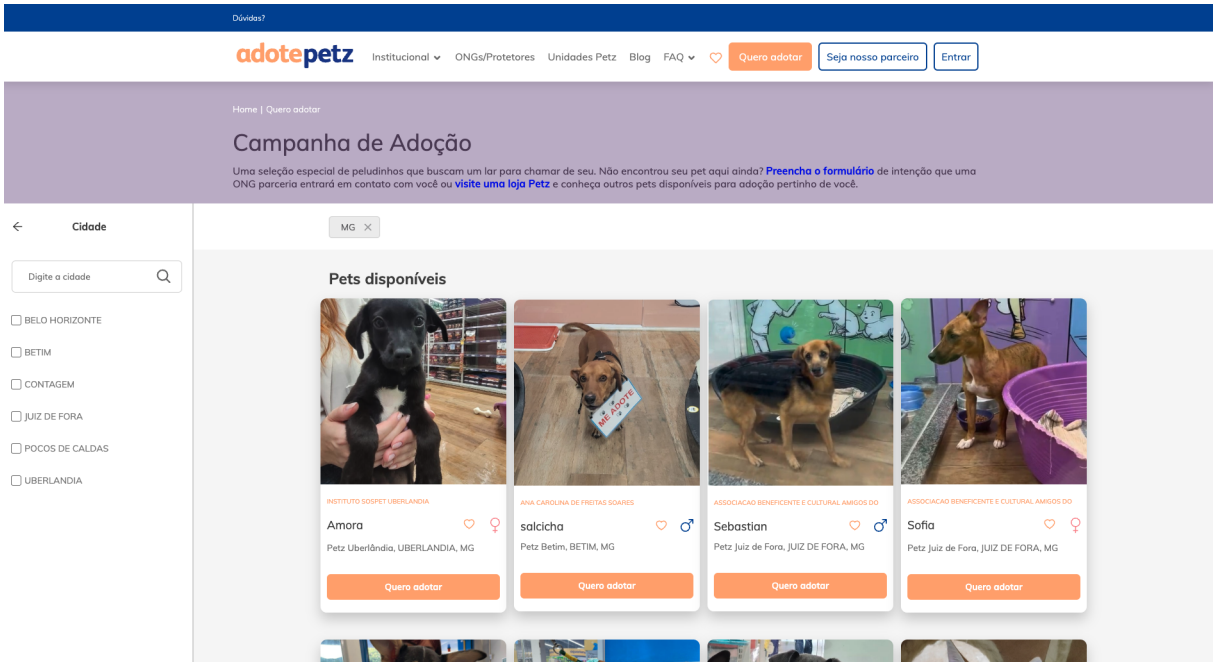
O Adote Petz é um programa desenvolvido pela rede de *pet shops* Petz com o objetivo de incentivar a adoção responsável de animais em parceria com ONGs e protetores independentes. A iniciativa busca conectar pessoas interessadas em adotar um *pet* com instituições que resgatam e cuidam de animais abandonados, proporcionando um ambiente estruturado para facilitar o processo de adoção (ADOTE PETZ, 2025).

A plataforma permite que os usuários conheçam os animais disponíveis para adoção por meio de eventos presenciais realizados nas lojas Petz, onde as ONGs cadastradas podem expor os *pets* e fornecer informações detalhadas sobre cada um deles. Além disso, a plataforma *online* do programa possibilita que os interessados visualizem cães e gatos de

ONGs e protetores selecionados diretamente pelo *site*, facilitando a escolha de um animal antes mesmo de comparecer a uma loja física. Dessa forma, a plataforma digital contribui para agilizar o processo de adoção, permitindo que os adotantes já iniciem a busca pelo *pet* ideal remotamente (ADOTE PETZ, 2025).

Apesar de sua abrangência nacional e da parceria com diversas ONGs, o programa ainda apresenta limitações quanto ao seu alcance geográfico. Como demonstrado na Figura 5, ao filtrar os animais disponíveis para adoção no estado de Minas Gerais, apenas seis municípios possuem *pets* cadastrados, restringindo as opções para adotantes de outras regiões. Isso indica que, embora o programa tenha um impacto significativo, ainda há desafios a serem superados para torná-lo acessível a um público maior, especialmente em cidades menores que não contam com unidades da Petz ou ONGs parceiras.

Figura 5 – *Print* da plataforma Adote Petz



Fonte: Adote Petz (2025)

2.1.3 Plataforma Amigo Não se Compra

O Amigo Não se Compra[1] é uma plataforma *online* de adoção de animais. De acordo com o *site* do projeto, ele é a maior plataforma *online* de adoção de animais do Brasil, fundada em 2012 com o objetivo de conectar cães e gatos que estão em abrigos ou sob os cuidados de protetores independentes a pessoas interessadas em adotar um novo amigo. A plataforma opera de forma totalmente *online*, sem possuir abrigos físicos ou

[1] https://www.amigonaosecompra.com.br/

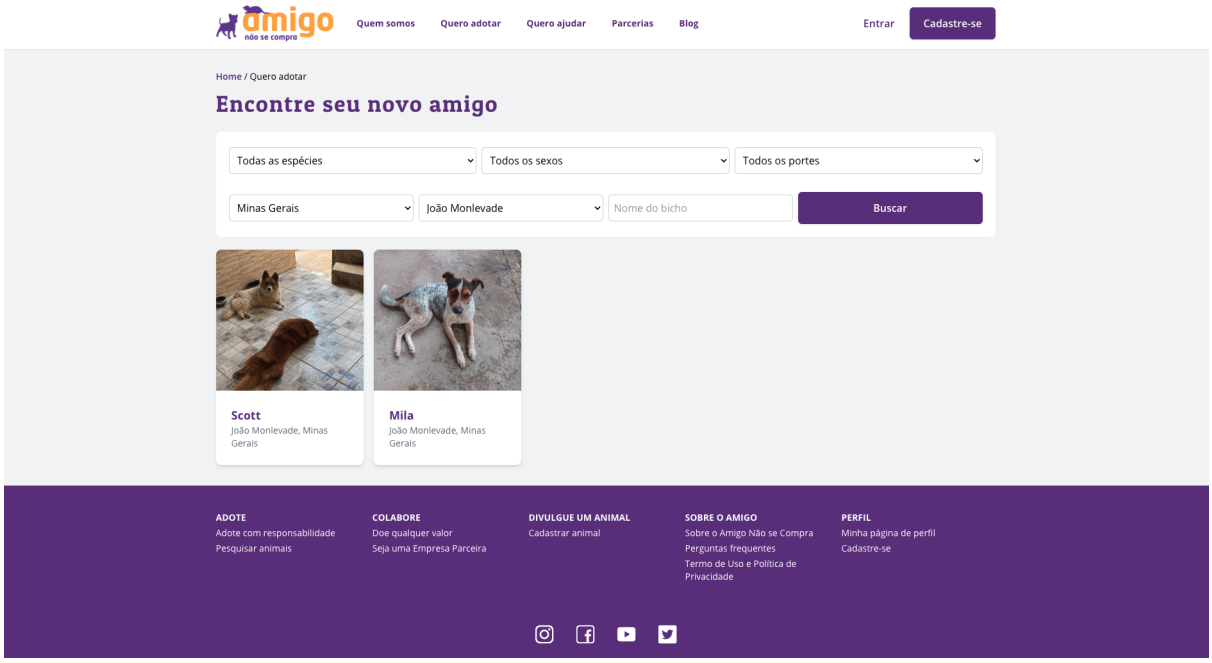
realizar resgates diretamente, focando na visibilidade dos animais que necessitam de um lar (AMIGO NÃO SE COMPRA, 2025).

A plataforma funciona da seguinte maneira: ONGs, protetores independentes ou indivíduos que precisam doar um animal realizam um cadastro gratuito e podem publicar informações detalhadas sobre os animais disponíveis para adoção, incluindo características físicas e traços de personalidade. Adotantes em potencial podem navegar pelo *site*, utilizar filtros para encontrar um animal que corresponda ao seu perfil e, ao encontrar o *pet* ideal, entrar em contato com o responsável pela publicação para acertar os detalhes da adoção (AMIGO NÃO SE COMPRA, 2025).

Até o momento, o Amigo Não se Compra já ajudou mais de 45 mil animais a encontrarem um lar, evidenciando o impacto positivo da plataforma na promoção da adoção responsável em todo o país. Além disso, a plataforma oferece orientações sobre adoção responsável e permite que os usuários contribuam de diversas formas, seja divulgando animais, tornando-se voluntários ou apoiando financeiramente a iniciativa (AMIGO NÃO SE COMPRA, 2025).

Diferentemente da plataforma Adote Petz, que apresenta limitações geográficas, o sistema do Amigo Não se Compra permite filtrar os animais por diversas cidades de Minas Gerais. Ao selecionar João Monlevade, uma cidade com cerca de 80 mil habitantes (IBGE, 2023), encontramos apenas dois animais disponíveis para adoção, como pode ser visto na Figura 6, que também demonstra a interface da plataforma.

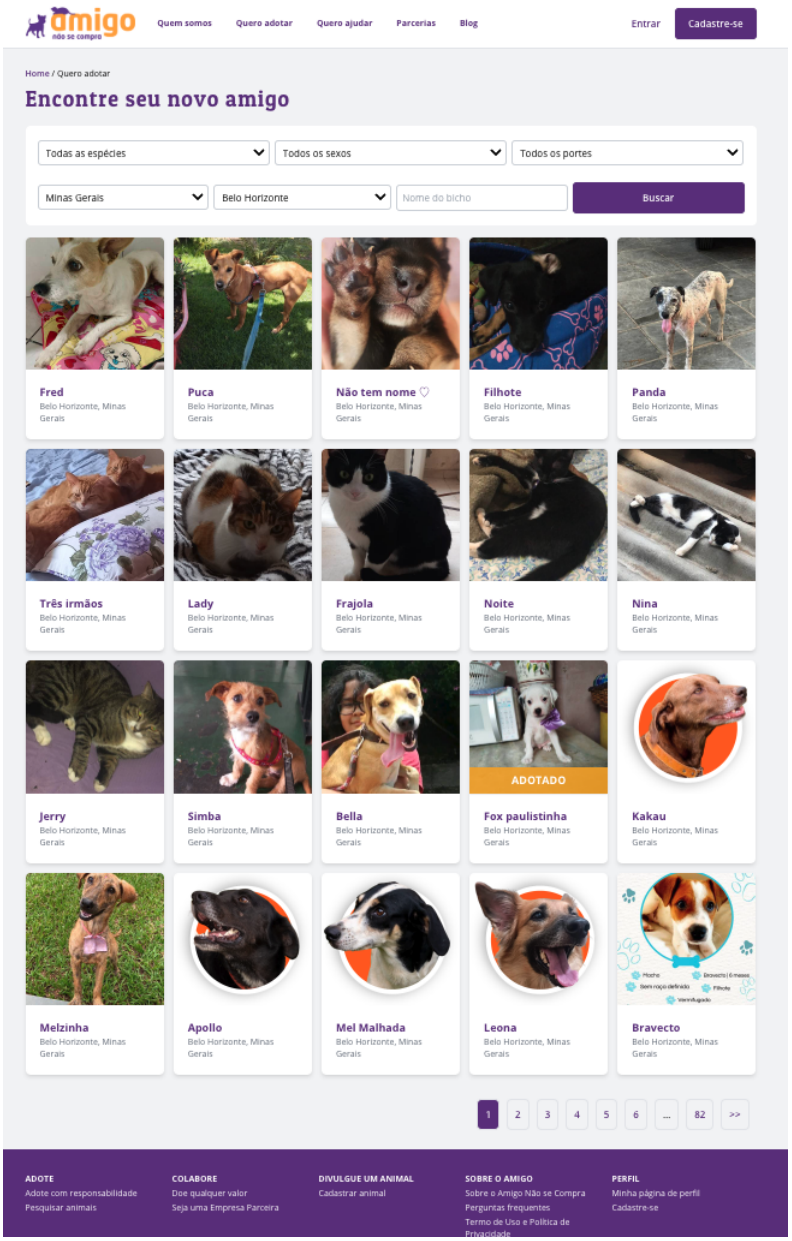
Figura 6 – *Print* da plataforma Amigo Não Se Compra filtrando os animais pela cidade de João Monlevade



Fonte: Amigo Não se Compra (2025)

Em contraste, ao realizar a mesma busca para Belo Horizonte, a capital de Minas Gerais, com uma população de aproximadamente 2,3 milhões de habitantes (IBGE, 2023), a plataforma apresenta um número significativamente maior de animais disponíveis para adoção, com aproximadamente 1.600 *pets* (Figura 7). Isso ilustra uma grande diferença entre as opções oferecidas em cidades com maior densidade populacional e aquelas em municípios menores. Embora a plataforma consiga fornecer mais opções em cidades maiores, como Belo Horizonte, a disponibilidade de animais ainda é limitada, evidenciando a escassez de adoções e a necessidade de maior alcance em áreas menos populosas.

Figura 7 – *Print* da plataforma Amigo Não Se Compra filtrando os animais pela cidade de Belo Horizonte



Fonte: Amigo Não se Compra (2025)

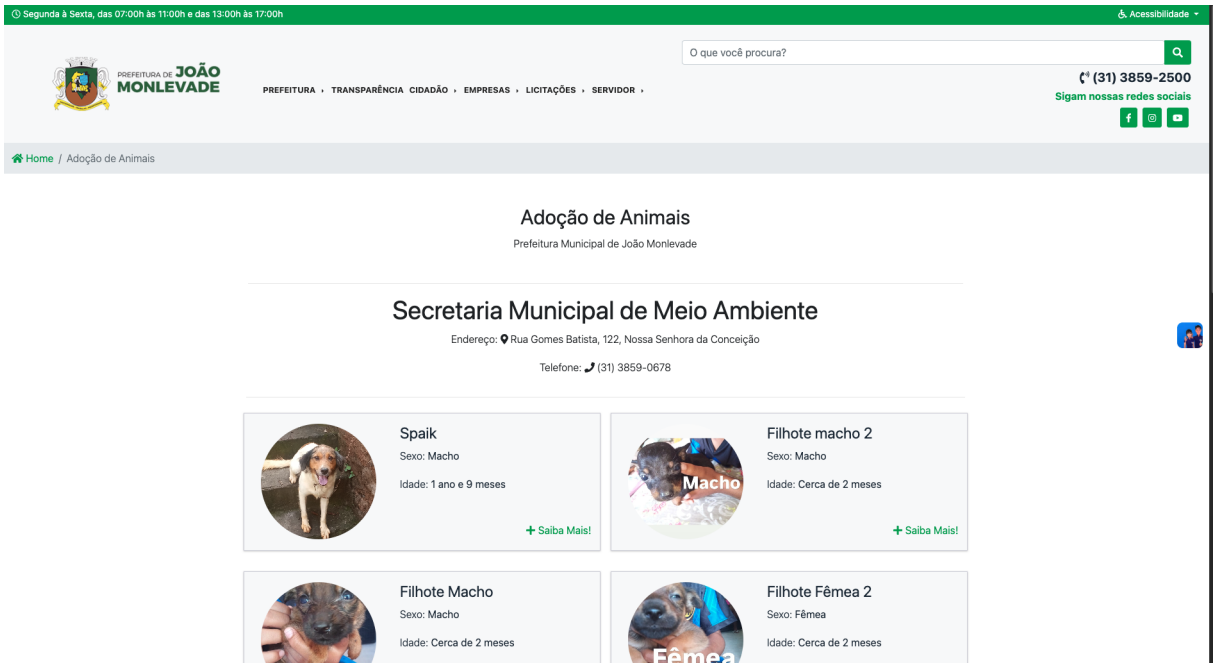
2.1.4 Espaço no *site* da Prefeitura de João Monlevade para adoção de animais resgatados

Em outubro de 2024, a Prefeitura de João Monlevade[2], por meio da Secretaria Municipal de Meio Ambiente – setor de Bem-Estar Animal, lançou em seu *site* oficial um espaço exclusivo para adoção de animais resgatados na cidade em situação de abandono.

[2] https://pmjm.mg.gov.br/

O objetivo é facilitar a conexão entre os animais que necessitam de um lar e os munícipes interessados em adotá-los (O POPULAR JM, 2024).

Figura 8 – *Print* da seção de animais disponíveis para adoção no *site* da prefeitura de João Monlevade



Fonte: Prefeitura Municipal de João Monlevade (2025)

O acesso à página é realizado a partir da aba “Cidadão” no *site* da prefeitura, selecionando a opção “Animais para Adoção”. Nessa seção, são apresentadas fotos e informações dos animais disponíveis, como pode ser visto na Figura 8. Ao clicar na imagem de um animal, o usuário encontra detalhes como sexo, raça, idade, estado de vacinação e castração, além do telefone/WhatsApp de contato da Secretaria de Meio Ambiente para iniciar o processo de adoção. Todo o procedimento é intermediado pela Secretaria, que realiza uma breve entrevista com os interessados para garantir que os animais sejam encaminhados a lares adequados (O POPULAR JM, 2024).

Inicialmente, estão sendo disponibilizados para adoção os animais que se encontram no canil municipal, todos resgatados em situação de abandono. Esses animais recebem cuidados, são vacinados e castrados antes de serem direcionados para um novo lar (O POPULAR JM, 2024). A iniciativa reforça o compromisso da administração municipal com o bem-estar animal e oferece uma oportunidade para que a comunidade participe ativamente na redução do número de animais abandonados na cidade.

Contudo, assim como a solução apresentada por Riva et al. (2022), a iniciativa da Prefeitura de João Monlevade ainda não oferece uma interação direta entre doadores e adotantes. Além disso, somente os animais do canil municipal estão disponíveis para adoção,

o que significa que indivíduos interessados em doar um animal não têm a possibilidade de disponibilizá-lo diretamente para adoção.

2.2 Abordagens e ferramentas no desenvolvimento do *software*

2.2.1 *Software* livre

Software livre é um conceito que se refere à liberdade dos usuários em executar, estudar, modificar e distribuir *software*. Um *software* é considerado livre quando o usuário tem a liberdade de acessar seu código-fonte e adaptá-lo conforme suas necessidades. Essa definição não diz respeito ao preço do *software*, mas sim à liberdade concedida ao usuário para fazer alterações e distribuir cópias. Assim, o termo "livre" está relacionado à liberdade, e não necessariamente à gratuidade do *software*. A principal vantagem do *software* livre é a autonomia dada ao usuário, permitindo-lhe controlar e modificar o programa de acordo com suas preferências e requisitos específicos (FREE SOFTWARE FOUNDATION, 2023).

Além disso, o *software* livre incentiva a colaboração e o compartilhamento de melhorias feitas por outros usuários. Com a disponibilidade do código-fonte, qualquer pessoa pode contribuir para o aprimoramento do *software*, criando um ambiente de desenvolvimento colaborativo. Isso resulta em um ciclo contínuo de melhorias, no qual os usuários podem beneficiar-se das modificações feitas por outros. Essa filosofia também favorece a transparência, já que o código-fonte pode ser auditado para garantir que não há comportamentos indesejados ou maliciosos embutidos no *software* (FREE SOFTWARE FOUNDATION, 2023).

De acordo com a definição de *software* livre feita pela Free Software Foundation (2023), os quatro princípios fundamentais que definem o *software* como livre são:

- **Liberdade de executar o programa**: o *software* pode ser executado para qualquer propósito;
- **Liberdade de estudar o programa**: o usuário pode estudar o código-fonte e modificar o *software* conforme necessário;
- **Liberdade de distribuir o programa**: o *software* pode ser compartilhado com outras pessoas;
- **Liberdade de melhorar o programa**: o usuário pode modificar o *software* e compartilhar as melhorias com a comunidade.

Esses princípios garantem que o *software* livre promova um ecossistema de colaboração e transparência, permitindo que todos os usuários possam se beneficiar e contribuir para o avanço do *software*. O movimento do *software* livre tem como objetivo não apenas

fornecer alternativas de *software*, mas também promover um modelo de desenvolvimento mais aberto e colaborativo (FREE SOFTWARE FOUNDATION, 2023).

2.2.2 Metodologias ágeis

No contexto moderno, as metodologias ágeis são amplamente adotadas no desenvolvimento de *software*, pois oferecem uma abordagem flexível que permite a entrega de *software* funcional de maneira iterativa e adaptável. Essas metodologias se baseiam na colaboração contínua com os clientes, possibilitando ajustes rápidos no sistema conforme novos requisitos surgem (SOMMERVILLE, 2019).

De acordo com Sommerville (2019), as principais características do desenvolvimento ágil de *software* incluem:

- O envolvimento contínuo do cliente ao longo de todo o processo de desenvolvimento, garantindo que suas necessidades sejam atendidas constantemente;
- A compreensão de que os requisitos do sistema podem evoluir durante o desenvolvimento, com a capacidade de adaptar o projeto às mudanças;
- A entrega incremental do *software*, dividindo o desenvolvimento em ciclos curtos para permitir ajustes rápidos e entregas frequentes de funcionalidades;
- O foco na simplicidade, tanto no design do *software* quanto nos processos de desenvolvimento, visando eficiência e redução de complexidade;
- A priorização das pessoas em relação aos processos, promovendo uma equipe colaborativa e flexível, em vez de se apegar a um processo rígido.

Essa abordagem é especialmente eficaz para o desenvolvimento de sistemas menores e médios, quando as equipes têm uma compreensão clara dos requisitos e o ambiente favorece a comunicação direta e a colaboração constante. Os métodos ágeis são ideais para situações em que os requisitos mudam frequentemente e o cliente deseja ver resultados contínuos e evolutivos ao longo do processo de desenvolvimento (SOMMERVILLE, 2019).

2.2.3 Engenharia de requisitos

Segundo Valente (2020), “Requisitos definem o que um sistema deve fazer e sob quais restrições”. A engenharia de requisitos, portanto, envolve um conjunto de atividades que vão desde a identificação até a documentação dos requisitos do sistema, sendo crucial para o sucesso do projeto, pois erros nesta fase podem resultar em retrabalho e insatisfação do cliente (VALENTE, 2020).

De acordo com Valente (2020), o processo de engenharia de requisitos é composto pelas seguintes etapas:

- **Levantamento de requisitos**: identificação e documentação das necessidades dos usuários e *stakeholders*, buscando compreender suas expectativas e problemas;
- **Análise de requisitos**: Avaliação da viabilidade, consistência e clareza das necessidades levantadas, garantindo que sejam bem definidas e compreendidas;
- **Especificação de requisitos**: detalhamento das funcionalidades do sistema, suas interações com o ambiente e as restrições impostas, resultando em uma documentação precisa que guiará o desenvolvimento;
- **Gerenciamento de requisitos**: monitoramento contínuo e controle das mudanças nos requisitos ao longo do ciclo de vida do projeto, assegurando que qualquer alteração seja adequadamente integrada e validada.

A comunicação eficaz com os *stakeholders* é crucial durante todo o processo de engenharia de requisitos. Por meio de reuniões e entrevistas, é possível obter informações detalhadas sobre as necessidades do sistema, ajustando o projeto conforme o *feedback* contínuo. A engenharia de requisitos deve ser encarada como um processo iterativo e dinâmico, que se adapta às mudanças e novas demandas ao longo do tempo. Ademais, a utilização de modelos de requisitos, como diagramas de casos de uso e protótipos, é uma prática comum para representar e validar as necessidades do sistema visual e interativamente, facilitando a compreensão de todos os envolvidos no projeto (VALENTE, 2020).

2.2.4 Histórias de usuário

As histórias de usuário são uma técnica fundamental dentro das metodologias ágeis, utilizadas para capturar e descrever as funcionalidades do sistema do ponto de vista dos usuários. Segundo Valente (2020), as histórias de usuário são formuladas de maneira simples, descrevendo uma necessidade ou funcionalidade específica, sem entrar em detalhes excessivos sobre a implementação. Elas são uma forma eficaz de comunicação entre os membros da equipe de desenvolvimento e os *stakeholders*, permitindo que todos compartilhem uma visão comum das funcionalidades esperadas no sistema (VALENTE, 2020).

Cada história de usuário normalmente segue um formato específico, que inclui a identificação do tipo de usuário, a necessidade ou objetivo do usuário, e o valor que a funcionalidade agregará ao sistema. O formato mais utilizado é: “Como [tipo de usuário], eu quero [funcionalidade] para [razão ou benefício]”. Esse formato simples ajuda a manter

o foco nas necessidades do usuário e no valor a ser entregue, sem complicar a descrição das funcionalidades (VALENTE, 2020).

A definição de histórias de usuário é uma atividade colaborativa, envolvendo os *stakeholders*, desenvolvedores e o cliente, que discutem e refinam as funcionalidades a serem implementadas. Após a criação, essas histórias são priorizadas de acordo com seu valor para o negócio e são transformadas em tarefas menores e mais gerenciáveis dentro de um quadro de trabalho ágil, como o Scrum ou Kanban (VALENTE, 2020).

Uma das vantagens das histórias de usuário é sua capacidade de fornecer um quadro flexível e adaptável, no qual os requisitos podem ser modificados à medida que novas informações são obtidas durante o processo de desenvolvimento. Além disso, elas promovem um entendimento compartilhado entre todos os membros da equipe, desde os desenvolvedores até os *stakeholders*, e ajudam a alinhar as expectativas em relação ao que será entregue em cada iteração (VALENTE, 2020).

2.2.5 Kanban

O Kanban tem suas origens no sistema de produção da Toyota, desenvolvido na década de 1940, com o objetivo de melhorar a eficiência e reduzir desperdícios no processo de fabricação. Sua adaptação para o desenvolvimento de *software* começou a ganhar popularidade no início dos anos 2000, especialmente com o aumento da adoção de práticas ágeis (AHMAD; MARKKULA; OIVO, 2013).

No contexto de *software*, o Kanban visa gerenciar o fluxo de trabalho de forma visual, permitindo que as equipes se concentrem na entrega contínua de valor ao cliente. Ao contrário de outras metodologias ágeis, como o Scrum, que se baseia em iterações ou *sprints*, o Kanban foca no fluxo contínuo de trabalho e na melhoria contínua dos processos (AHMAD; MARKKULA; OIVO, 2013).

A aplicação do Kanban no desenvolvimento de *software* envolve a visualização do trabalho e a limitação do número de tarefas em andamento, o que ajuda a identificar gargalos no processo e a melhorar o fluxo de trabalho. O sistema Kanban é tipicamente representado por um quadro dividido em colunas, em que cada coluna representa uma etapa do processo de desenvolvimento, como “A Fazer”, “Em Progresso” e “Concluído”. As tarefas são representadas por cartões, que são movidos entre as colunas à medida que avançam no fluxo de trabalho. Essa visualização permite que todos os membros da equipe acompanhem o progresso das tarefas e identifiquem facilmente áreas nas quais o fluxo está sendo bloqueado (AHMAD; MARKKULA; OIVO, 2013).

Uma das características essenciais do Kanban é a limitação do trabalho em progresso — *Work In Progress* (WIP). A ideia por trás da limitação de WIP é evitar sobrecarregar a equipe e garantir que cada tarefa receba a atenção necessária até sua conclusão. Limitar

o WIP ajuda a reduzir o tempo de ciclo, ou seja, o tempo necessário para concluir uma tarefa desde o início até a sua finalização. Ao reduzir o número de tarefas em andamento, as equipes podem se concentrar melhor nas tarefas prioritárias, melhorando a eficiência geral do processo de desenvolvimento (AHMAD; MARKKULA; OIVO, 2013).

Além disso, o Kanban promove a melhoria contínua por meio de métricas, como o tempo de ciclo e o *lead time*, que fornecem informações valiosas sobre o desempenho da equipe e a eficiência do processo. A análise dessas métricas permite que as equipes identifiquem gargalos e implementem mudanças para otimizar o fluxo de trabalho. A adaptação constante do processo com base nas métricas é uma das chaves para o sucesso do Kanban, permitindo que a equipe se torne cada vez mais eficiente ao longo do tempo (AHMAD; MARKKULA; OIVO, 2013).

2.2.6 Testes unitários

Os testes unitários são uma das práticas mais importantes para garantir a qualidade e a confiabilidade de sistemas de *software*. Essa técnica consiste em verificar o comportamento de unidades individuais de código, como métodos ou classes, isoladamente, para assegurar que cada parte do sistema execute conforme o esperado (VALENTE, 2020). Valente (2020) define que essa abordagem permite detectar falhas precocemente, o que contribui para um desenvolvimento mais ágil e menos propenso a erros, já que problemas são identificados antes de se propagarem para outras partes do sistema.

Ao realizar testes unitários, cada unidade do código é testada de forma independente, o que facilita a localização de falhas e melhora a manutenibilidade do *software*. Essa prática ajuda os desenvolvedores a manterem o código de alta qualidade, pois garante que alterações feitas em uma parte do sistema não quebrem outras funcionalidades. O uso de frameworks de testes automatiza a execução desses testes e facilita sua integração no processo de desenvolvimento contínuo (VALENTE, 2020).

Os testes unitários também oferecem uma maneira de documentar o sistema, já que eles descrevem o comportamento esperado de cada unidade do código. Isso proporciona um modo de especificação que, além de validar as funcionalidades, também serve como referência para novos desenvolvedores ou membros da equipe, esclarecendo como determinado componente do sistema deve se comportar sob diferentes condições (VALENTE, 2020).

3 Desenvolvimento

3.1 Documento de visão

O projeto foi iniciado com o preenchimento do documento de visão, com a finalidade de definir claramente as metas e as funcionalidades que a plataforma de adoção de animais deveria atender. O documento descreveu os objetivos e o escopo do sistema, as principais funcionalidades e os requisitos de alto nível, além de apresentar a visão geral do produto e as necessidades do usuário.

O objetivo principal do sistema, conforme descrito no documento de visão, é facilitar o processo de adoção de animais. A plataforma foi projetada para permitir que usuários se conectem de maneira simples e eficiente, seja para adotar ou para doar animais. Além disso, o sistema foi desenvolvido para ser totalmente gratuito para todas as partes envolvidas, sem custos para adoção ou divulgação dos animais. O sistema também foi disponibilizado como um projeto de *software* livre, permitindo que outras organizações e prefeituras o utilizem e adaptem conforme suas necessidades, ampliando seu alcance e impacto.

O escopo do projeto foi restrito no documento ao desenvolvimento da plataforma de adoção sem incluir o processo de implantação ou a manutenção do sistema após o término do desenvolvimento. O foco foi criar uma plataforma *web*, com funcionalidades básicas para cadastro de animais, busca por características específicas e comunicação entre adotantes e doadores. Assim, o sistema não tem responsabilidade sobre o acompanhamento do animal após a adoção.

O documento de visão também detalhou os perfis de usuários, que incluem tanto os doadores quanto os adotantes de animais. A plataforma foi projetada para ser acessada por meio de computadores e dispositivos móveis, o que amplia a gama de usuários potenciais. O documento pode ser conferido na íntegra no Apêndice A.1.

3.2 Documento de requisitos

Segundo Valente (2020), o levantamento de requisitos envolve a identificação das necessidades dos usuários, a documentação dessas necessidades e a análise da viabilidade de seu atendimento. No projeto AdoteFácil, o documento de requisitos foi desenvolvido com base nas necessidades dos usuários, os quais são contemplados por doadores e adotantes de animais. Durante esse processo, foram identificadas as funcionalidades essenciais que a plataforma deveria oferecer, como o cadastro de animais disponíveis para adoção, a busca por características específicas e a comunicação direta entre adotantes e doadores por meio

de um sistema de *chat*.

O documento de requisitos também definiu outras funcionalidades, como a atualização de dados dos usuários e a confirmação de adoção pelos doadores. A decisão de não incluir um sistema de acompanhamento pós-adoção foi tomada para garantir que o foco do sistema permanecesse exclusivamente na simplificação do processo de adoção de animais.

Ademais, o documento de requisitos abordou aspectos importantes como o desempenho e a usabilidade da plataforma. O sistema foi projetado para ter tempos de resposta mínimos, para garantir uma experiência satisfatória aos usuários. Além disso, a interface do sistema é simples e responsiva, portanto há a possibilidade de acessá-lo via computadores e dispositivos móveis. A responsividade foi priorizada levando em consideração que muitos usuários, especialmente em regiões com menos infraestrutura tecnológica, acessam plataformas *online* principalmente por meio de *smartphones*. O documento pode ser conferido na íntegra no Apêndice A.2.

3.3 Definição de tecnologias para o desenvolvimento

A escolha das tecnologias para o desenvolvimento do sistema AdoteFácil foi orientada principalmente pela experiência do autor, além de considerar as tendências de mercado e as principais tecnologias utilizadas no desenvolvimento *web*. O projeto foi desenvolvido utilizando Typescript[1] como linguagem principal, que é extensivamente adotada para o desenvolvimento de sistemas *web* modernos, devido à sua robustez, tipagem estática e suporte a ferramentas de desenvolvimento avançadas.

Para o *frontend*, foi escolhida a biblioteca React[2] em conjunto com o *framework* Next.js[3]. React é uma das tecnologias mais populares para a construção de interfaces de usuário dinâmicas e reativas, permitindo a criação de componentes reutilizáveis e facilitando a gestão do estado da aplicação. O Next.js, por sua vez, é um *framework* que complementa o React, oferecendo recursos como *Server Side Rendering* (SSR)[4] , otimização automática e roteamento simplificado. Isso permite que o sistema tenha um desempenho superior em termos de *Search Engine Optimization* (SEO)[5] e tempo de carregamento, ao mesmo tempo em que facilita a construção de páginas dinâmicas e de fácil manutenção.

[1] https://www.typescriptlang.org/
[2] https://react.dev/
[3] https://nextjs.org/
[4] O SSR no Next.js é um processo em que o conteúdo da página é gerado no servidor antes de ser enviado para o cliente, permitindo que o conteúdo da página seja renderizado mais rapidamente em relação à renderização do lado do cliente.
[5] SEO é o conjunto de práticas e técnicas utilizadas para otimizar a visibilidade de um *site* nos motores de busca, como o Google, com o objetivo de melhorar o posicionamento do *site* nos resultados orgânicos, atraindo mais tráfego e, consequentemente, aumentando a relevância e o alcance da página na *internet*.

No *backend*, foi escolhido o Node.js[6] em conjunto com o Express[7], criando uma *Application Programming Interface* (API) *Restful* para gerenciar as requisições e a comunicação entre o *frontend* e o banco de dados. Node.js é amplamente utilizado para construir aplicações escaláveis e de alto desempenho, especialmente em ambientes de desenvolvimento ágil, enquanto o Express simplifica a criação de rotas e o gerenciamento de requisições HTTP, proporcionando uma base sólida e flexível para o *backend*.

Para o banco de dados foi escolhido o PostgreSQL[8] devido à sua robustez, confiabilidade e conformidade com os padrões de *Structured Query Language* (SQL). Trata-se de um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR) de código aberto, amplamente utilizado em projetos de *software* devido à sua flexibilidade, escalabilidade e suporte a consultas complexas. O PostgreSQL oferece suporte a transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade), o que garante integridade e segurança dos dados. Além disso, sua capacidade de lidar com grandes volumes de dados e realizar consultas de alto desempenho torna-o uma escolha ideal para sistemas *web* dinâmicos e em constante crescimento.

O Prisma[9] foi escolhido como o *Object-Relational Mapping* (ORM) para o sistema devido à sua simplicidade, desempenho e recursos avançados que facilitam a interação com o banco de dados PostgreSQL. O Prisma é uma ferramenta moderna e eficiente, amplamente utilizada no ecossistema Node.js, que permite mapear e acessar o banco de dados de maneira intuitiva e sem a complexidade de escrever SQL manualmente. Ele oferece uma API poderosa e de fácil uso para realizar operações de leitura e escrita, além de permitir a validação de dados antes da inserção no banco. O Prisma também suporta migrações de banco de dados de modo automatizado, o que facilita a manutenção e evolução do esquema do banco de dados ao longo do ciclo de vida do projeto.

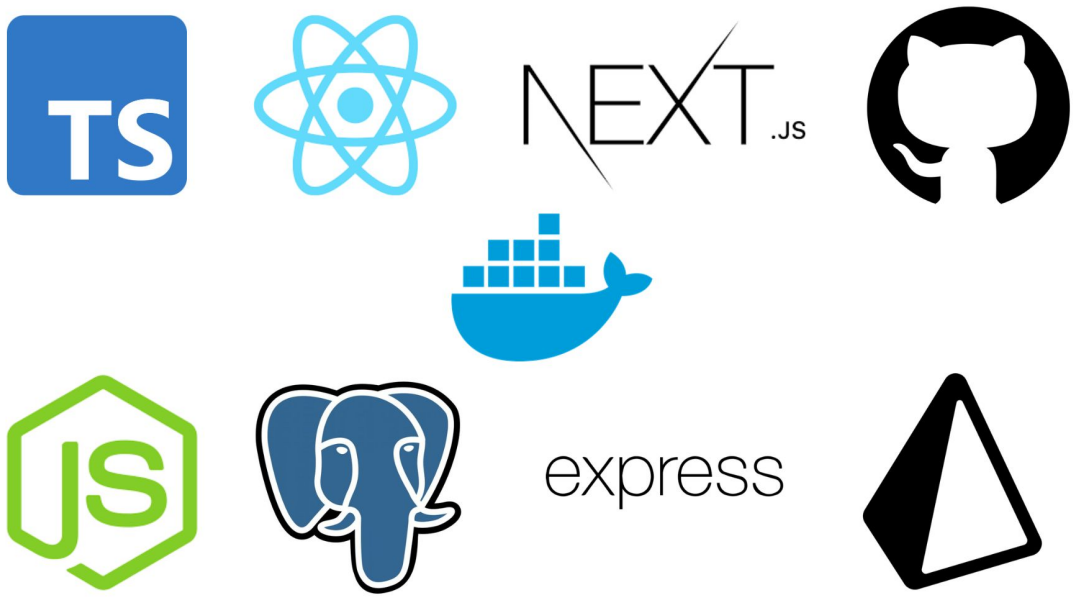
Além disso, o sistema foi conteinerizado utilizando Docker[10], o que facilita a execução e a implantação do ambiente de desenvolvimento e produção, garantindo que o *software* funcione de consistentemente em diferentes plataformas e configurações. O uso de Docker permite que o sistema seja facilmente distribuído e implantado em diferentes ambientes, seja para uso local ou em servidores de produção. Para orquestrar os contêineres, foi utilizado também o Docker Compose, que simplifica o processo de configuração e gerenciamento dos múltiplos contêineres necessários para o funcionamento da aplicação, garantindo uma integração eficiente entre os serviços e um ambiente de desenvolvimento mais ágil.

Por fim, para o controle de versão do projeto, foi escolhido o GitHub, uma plataforma amplamente utilizada na indústria de *software* devido à sua robustez e facilidade de

[6] https://nodejs.org/pt
[7] https://expressjs.com/pt-br/
[8] https://www.postgresql.org/
[9] https://www.prisma.io/
[10] https://www.docker.com/

integração com diversas ferramentas. O GitHub permite um controle eficiente de versões, possibilitando a colaboração e o gerenciamento de diferentes ramificações do código de forma simples e segura. Além disso, o GitHub foi a escolha para disponibilizar o código do projeto publicamente, permitindo que a comunidade possa acessar, utilizar, adaptar e contribuir para a evolução da plataforma. A Figura 9 demonstra as logos das principais tecnologias utilizadas no desenvolvimento.

Figura 9 – Logo das tecnologias utilizadas no projeto



Fonte: Elaborado pelo autor

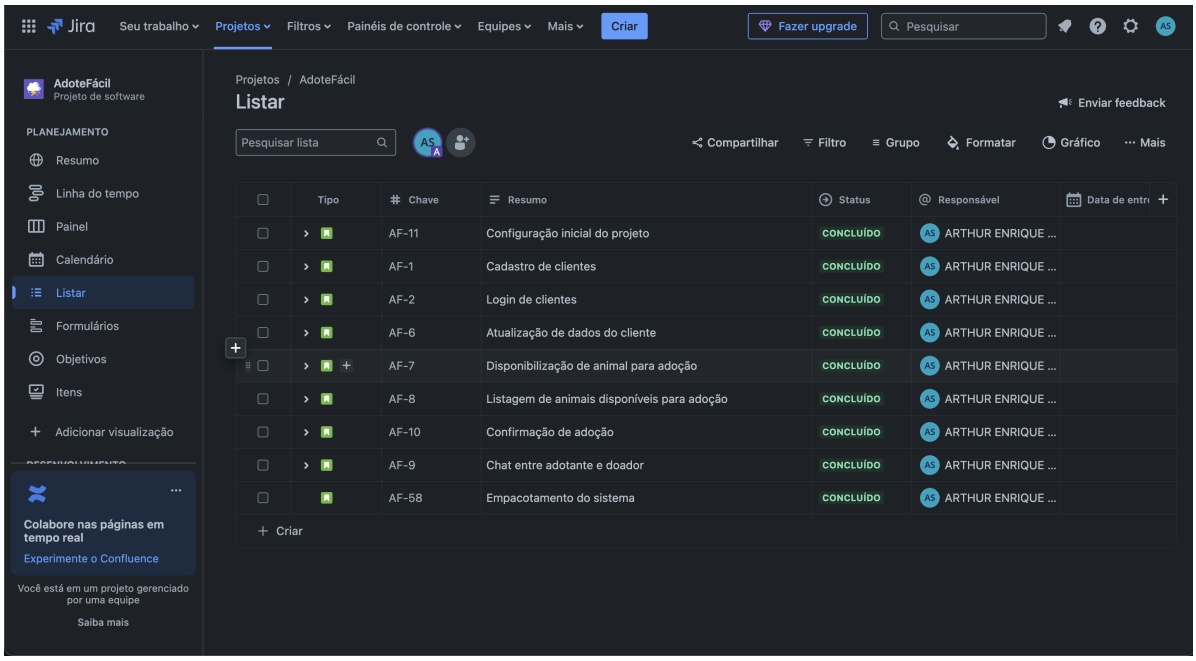
3.4 Histórias de Usuário

Para a escrita das histórias de usuário, foi adotado o padrão descrito no artigo de Pereira (2014), que propõe a estrutura: "COMO [papel] EU GOSTARIA DE [funcionalidade] PARA CONSEGUIR [objetivo]". Esse formato ajuda a manter o foco nas necessidades do usuário e no valor que a funcionalidade trará. Para organizar e gerenciar as histórias de usuário, foi escolhida a ferramenta Jira[11], uma plataforma amplamente reconhecida no mercado para o monitoramento de projetos e itens, além de ser altamente eficaz na gestão ágil de tarefas. No Jira, foi criado um projeto dedicado ao AdoteFácil, onde todas as histórias de usuário, junto com suas subtarefas, foram registradas e puderam ser acompanhadas por meio de um quadro Kanban.

[11] https://www.atlassian.com/br/software/jira

Infelizmente, a funcionalidade para tornar o projeto público no Jira não estava disponível, pois essa opção é uma característica paga da plataforma. Como resultado, não foi possível compartilhar o projeto para exibição pública. No entanto, a Figura 10 apresenta uma captura de tela do projeto na plataforma, ilustrando as histórias de usuário e seus respectivos status.

Figura 10 – *Print* do Projeto Adote Fácil no Jira



Fonte: Elaborado pelo autor

É importante ressaltar que as histórias de usuário foram escritas antes de começar o desenvolvimento, portanto diversos detalhes de implementação foram alterados ao longo do desenvolvimento do *software*.

A seguir está descrita e detalhada a história de usuário de cadastro de cliente, com seus respectivos cenários de teste e subtarefas. As demais histórias de usuário descritas no projeto podem ser encontradas na íntegra no Apêndice A.3

Descrição:

```
COMO uma pessoa comum
EU GOSTARIA DE poder me cadastrar no sistema AdoteFácil
PARA CONSEGUIR acessar as funcionalidades do sistema
```

Cenários de teste:

```
Cenário principal - Usuário se cadastra com informações válidas
```

```
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir suas informações no formulário de cadastro
    (nome, email e senha) e pressionar o botão "Cadastrar"
    ENTÃO a aplicação deve salvar as informações da pessoa no banco
    (com a senha criptografada) e redirecioná-la para a tela de login

Cenário alternativo - Usuário insere email inválido
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir um email inválido no formulário de cadastro
    ENTÃO a aplicação deve informar à pessoa que o email é inválido e
    impedir que o botão "Cadastrar" seja clicado.

Cenário alternativo - Usuário insere senhas diferentes no input de senha
e confirmação de senha
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir senhas diferentes no input de senha e
    de confirmação de senha no formulário de cadastro
    ENTÃO a aplicação deve informar à pessoa que as senhas estão
    divergentes e impedir que o botão "Cadastrar" seja clicado.

Cenário alternativo - Usuário insere email já existente
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir suas informações no formulário de cadastro,
    pressionar o botão "Cadastrar" e o email inserido já estiver
    cadastrado no sistema
    ENTÃO a aplicação deve informar um erro ao cliente com a mensagem
    "Email já cadastrado no sistema" e se manter na tela de cadastro
```

Subtarefas:

- Protótipo do cadastro *mobile*
- Protótipo do cadastro *web*
- Tela de cadastro
- Rota de cadastro no *backend*
- Integração da tela de cadastro com *backend*

3.5 Prototipação do sistema

Após a escrita das histórias de usuário, foi feita a prototipação inicial do sistema. A ferramenta utilizada para prototipação foi o Figma[12], amplamente utilizado por *designers* no mercado para prototipação de *software*. Foi criado um projeto na plataforma (que pode ser acessado neste link), utilizando como *template* o protótipo de um projeto feito pela Rocketseat[13] chamado *Ignite Feed*, uma rede social voltada para a área de Tecnologia da Informação (TI).

Após a escolha das principais cores da marca, foi necessário desenvolver um logotipo para a plataforma. O logotipo apresentado na Figura 11 foi criado com base em sugestões geradas pela ferramenta de Inteligência Artificial (IA) Gemini[14]. A Figura 12 exibe algumas das opções iniciais propostas pela ferramenta, que levou em consideração as cores da marca para criar variações do logotipo.

Figura 11 – Logotipo da plataforma

Fonte: Elaborado pelo autor

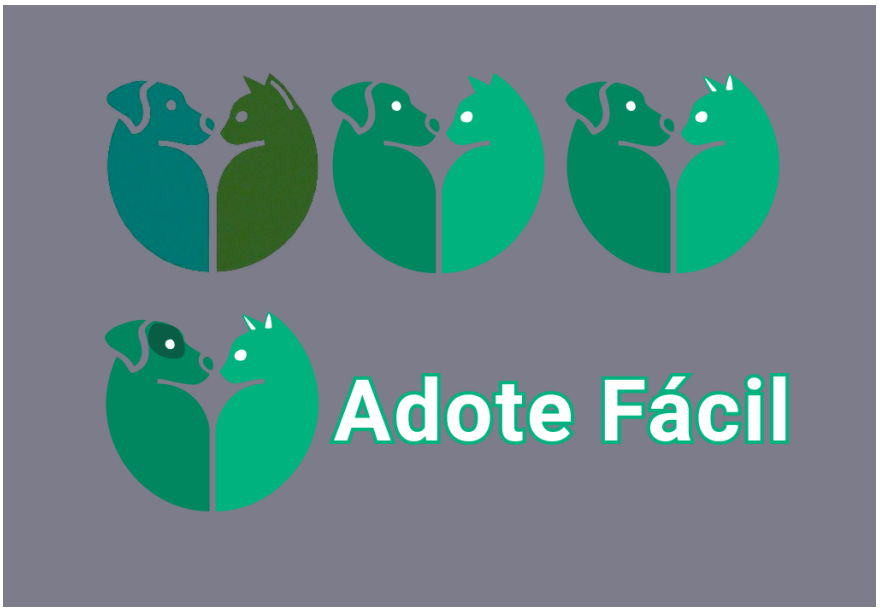
Posteriormente, o logotipo foi desenhado no Figma utilizando elipses, com ajustes no formato para alcançar um círculo perfeito. Após refinamentos no estilo e alguns ajustes finais, chegou-se ao resultado final. A Figura 13 apresenta alguns resultados parciais neste processo de refinamento do logo, assim como o resultado final, já apresentado na Figura 11.

[12] https://www.figma.com
[13] https://www.rocketseat.com.br/
[14] https://gemini.google.com/app?hl=pt-BR

Figura 12 – Sugestões de logotipo criadas pela ferramenta Gemini

Fonte: Elaborado pelo autor

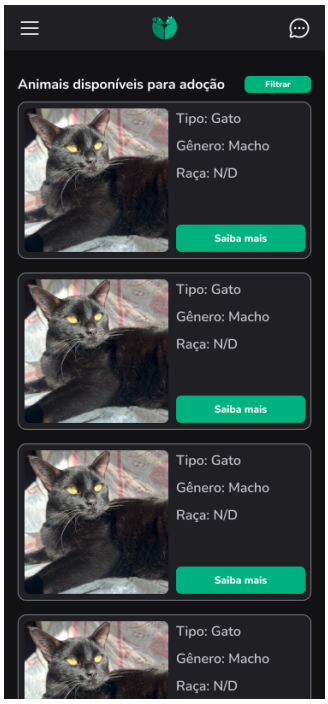
Figura 13 – Processo de criação do logotipo no Figma



Fonte: Elaborado pelo autor

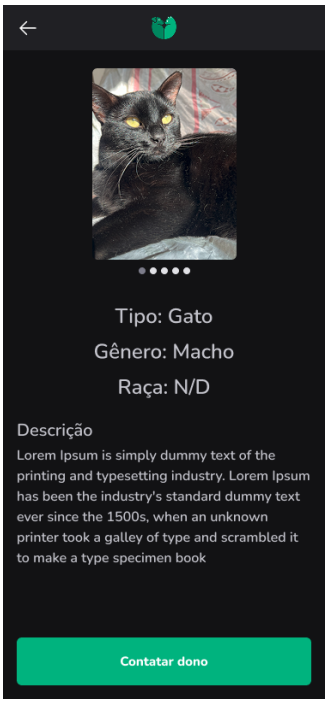
Com o logotipo da plataforma criado, partiu-se para o desenvolvimento dos protótipos das principais telas do sistema. A seguir estão demonstrados os protótipos da tela principal e da tela de informações do animal (Figuras 14 e 15, respectivamente). Os protótipos das demais telas do sistema podem ser encontrados no Apêndice A.4.

Figura 14 – Protótipo da tela principal



Fonte: Elaborado pelo autor

Figura 15 – Protótipo da tela de informações do animal



Fonte: Elaborado pelo autor

Assim como as histórias de usuário, os protótipos foram criados para orientar o desenvolvimento do projeto, oferecendo uma visão inicial da interface do sistema. No entanto, durante o progresso do desenvolvimento, novas ideias e ajustes foram surgindo, fazendo com que o sistema final não fosse totalmente fiel aos protótipos iniciais. Isso é perfeitamente comum no desenvolvimento de *software*, uma vez que os protótipos servem como rascunhos e não têm a intenção de refletir com precisão o produto final, mas sim fornecer uma base para evolução e melhorias contínuas.

3.6 Desenvolvimento do Sistema

O desenvolvimento do sistema Adote Fácil seguiu um processo adaptado às necessidades do projeto, levando em consideração que o trabalho foi realizado de forma individual, utilizando o Kanban no Jira para acompanhamento do desenvolvimento. O Kanban ajudou a visualizar o progresso das tarefas, com histórias e subtarefas sendo atualizadas conforme eram iniciadas e concluídas. Esse fluxo de trabalho simplificado permitiu acompanhar o andamento do projeto de maneira clara e sem sobrecarga administrativa.

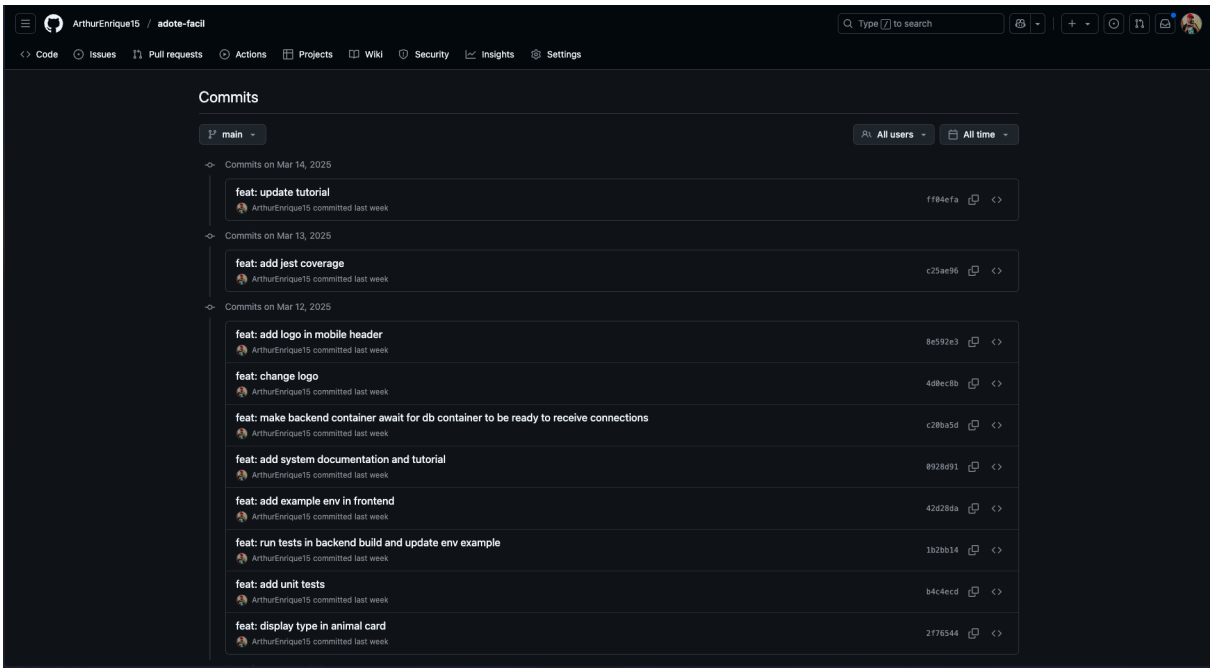
Inicialmente, o plano era desenvolver toda a interface da aplicação primeiro, para depois focar nas funcionalidades do *backend* e na integração entre as partes. No entanto, após algumas semanas de desenvolvimento, foi identificado que essa abordagem não estava sendo eficaz, pois nenhuma funcionalidade era totalmente finalizada, fazendo com que não

houvessem demonstráveis reais das entregas, apenas demonstrações da interface. Assim, a estratégia foi alterada para um desenvolvimento mais incremental, focando em finalizar uma funcionalidade completamente antes de partir para a próxima.

Dessa forma, a primeira funcionalidade implementada foi o cadastro e *login* de usuários, seguido pela disponibilização de animais para adoção, que incluía o cadastro de informações e a exibição dos animais disponíveis. Em seguida, foi desenvolvida a listagem de animais, tanto os disponíveis quanto os do próprio usuário, e o sistema de *chat* entre adotantes e doadores. Por fim, a última funcionalidade implementada foi a confirmação de adoção, que permitiu que o processo fosse concluído com sucesso. A estratégia de desenvolver *features* completas uma de cada vez provou ser mais eficaz, pois permitiu maior foco nas funcionalidades e na integração do sistema como um todo.

Ao longo de todo o desenvolvimento, o código foi versionado incrementalmente, com *commits* realizados à medida que cada parte das funcionalidades era concluída. Esses *commits* foram regularmente enviados para o repositório no GitHub[15], garantindo o controle de versão eficiente e permitindo o acompanhamento do progresso do projeto de forma contínua. A Figura 16 demonstra a lista de *commits* no repositório.

Figura 16 – Lista de *commits* no repositório do GitHub



Fonte: Elaborado pelo autor

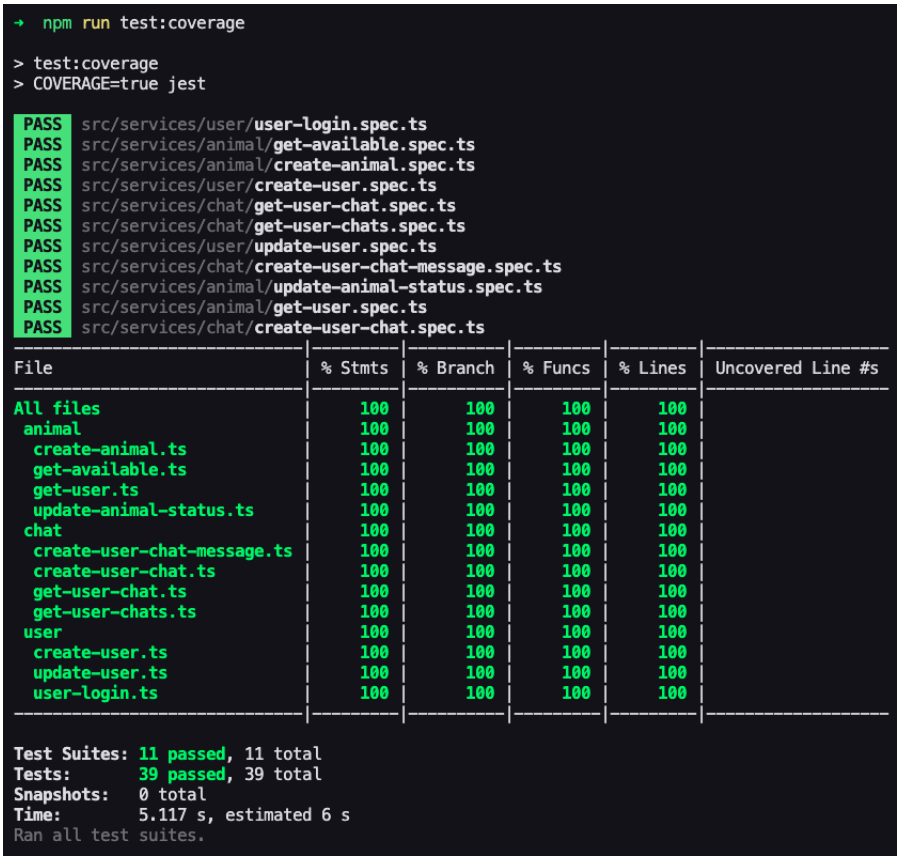
[15] https://github.com/ArthurEnrique15/adote-facil

3.7 Implementação dos testes unitários

Após a conclusão do desenvolvimento do sistema, foi realizada a implementação dos testes unitários para os *services* do *backend*. Para essa tarefa, foi escolhido o Jest[16], um *framework* de testes amplamente utilizado na comunidade Javascript, conhecido pela sua simplicidade e eficiência na escrita e execução de testes. O código dos testes também foi enviado ao repositório do GitHub.

Uma das funcionalidades essenciais do Jest é o relatório de cobertura de testes, que fornece a porcentagem de linhas de código cobertas pelos casos de teste. Conforme ilustrado nas Figuras 17 e 18, após a implementação de todos os testes unitários, foi alcançada uma cobertura de 100% do código, garantindo que todas as funcionalidades críticas do sistema fossem completamente testadas.

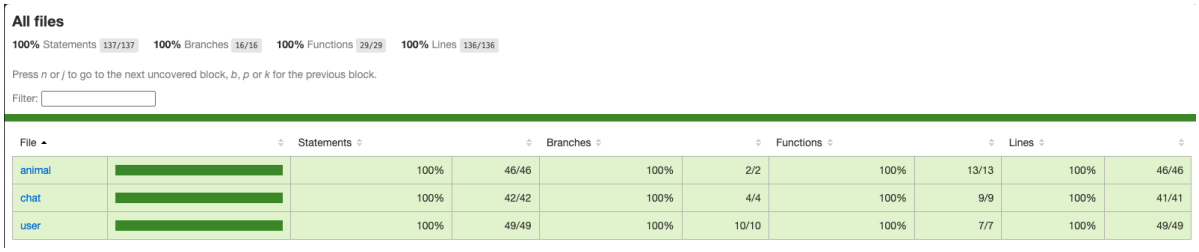
Figura 17 – Relatório de cobertura de testes no terminal



Fonte: Elaborado pelo autor

[16] https://jestjs.io/pt-BR/

Figura 18 – Relatório de cobertura de testes no arquivo criado pelo Jest



Fonte: Elaborado pelo autor

3.8 Empacotamento do Sistema

Para facilitar a implantação e garantir a consistência do ambiente de desenvolvimento e produção, o sistema foi conteinerizado utilizando Docker, com a orquestração dos contêineres realizada pelo Docker Compose. Na configuração do Docker Compose, foi criado um contêiner com a imagem oficial do PostgreSQL 14 para gerenciar o banco de dados. Além disso, foram configurados dois arquivos Dockerfile: um para o *frontend* e outro para o *backend*, ambos utilizando a imagem do Node.js 20 Alpine. Esses contêineres foram configurados para garantir que o *backend* só inicie após o banco de dados estar pronto para receber conexões, evitando problemas de dependência durante a inicialização.

Assim, com a execução do comando ***docker compose up***, todos os contêineres são iniciados automaticamente, permitindo que a API e o banco de dados sejam executados em praticamente qualquer dispositivo sem a necessidade de configurações complexas, simplificando o processo de implantação e garantindo que o sistema funcione de maneira consistente, independentemente do ambiente em que está sendo executado.

3.9 Licença de *Software* Livre GNU/GPLv3

O conceito de *software* livre, abordado anteriormente no Capítulo 2 Seção 2.2.1, está relacionado à liberdade concedida aos usuários para executar, estudar, modificar e distribuir o *software*. Essas liberdades permitem que o código-fonte seja acessível e adaptável, possibilitando que qualquer pessoa possa contribuir e compartilhar melhorias (FREE SOFTWARE FOUNDATION, 2023).

No desenvolvimento deste projeto, foi escolhida a licença GNU General Public License v3 (GPLv3) para garantir que o *software* fosse disponibilizado como livre e de código aberto. A GPLv3 permite que qualquer pessoa utilize, modifique e distribua o código, desde que as mesmas liberdades sejam preservadas nas versões modificadas (FREE SOFTWARE FOUNDATION, 2022).

A licença também exige que o código-fonte seja disponibilizado para qualquer pessoa que receba o *software*, garantindo transparência e permitindo que o código seja auditado. Além disso, a GPLv3 protege contra práticas como o uso de patentes para restringir as liberdades do *software* livre e impede que outras adições restritivas sejam impostas ao *software*. Um aspecto importante da GPLv3 é a exigência de que qualquer *software* derivado seja distribuído sob a mesma licença, garantindo que o código continue livre, mesmo nas versões modificadas (FREE SOFTWARE FOUNDATION, 2022).

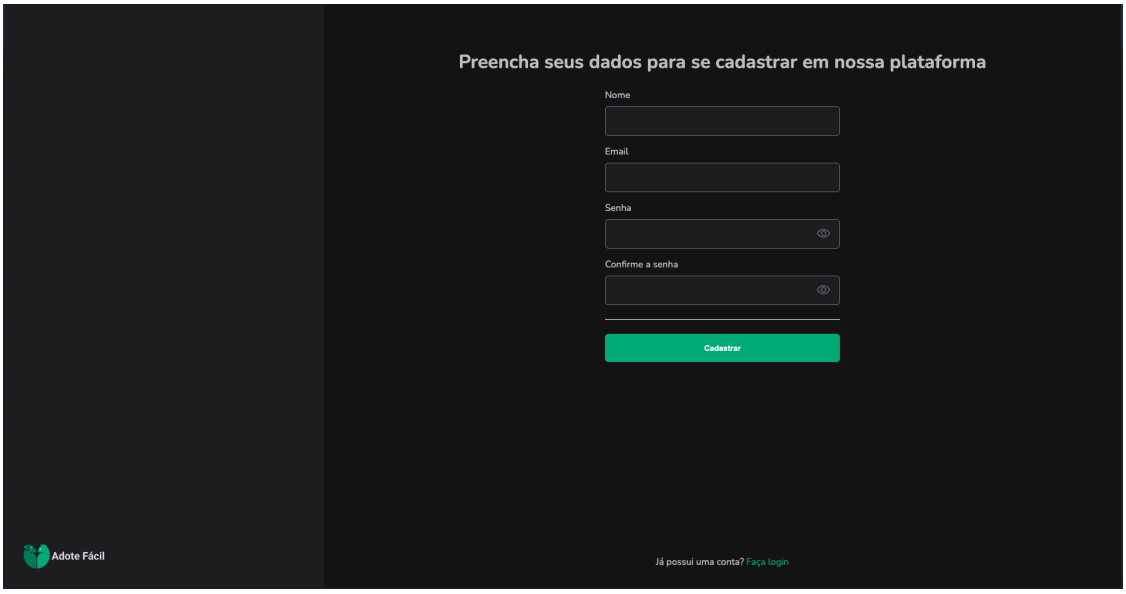
Com isso, ao liberar o AdoteFácil sob a licença GPLv3, buscou-se permitir que outras organizações, prefeituras e desenvolvedores utilizem e alterem o código da plataforma livremente e sem custos, ampliando seu alcance e impacto no processo de adoção de animais. A licença também pode ser encontrada no repositório do GitHub.

4 Resultados

4.1 O sistema

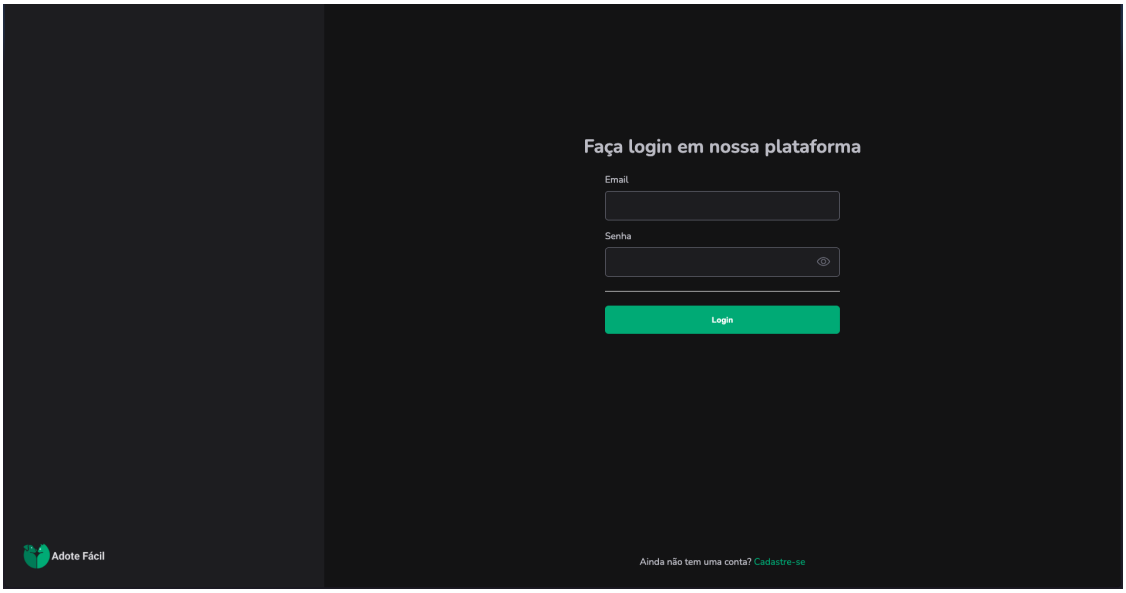
A seguir, serão demonstradas todas as telas do sistema, em suas versões *web* e *mobile*. Inicialmente, o usuário acessa a tela de cadastro (Figuras 19 e 21), tendo também a opção na parte inferior de ir para a página de *login* (Figuras 20 e 22).

Figura 19 – Tela de cadastro de usuário *web*



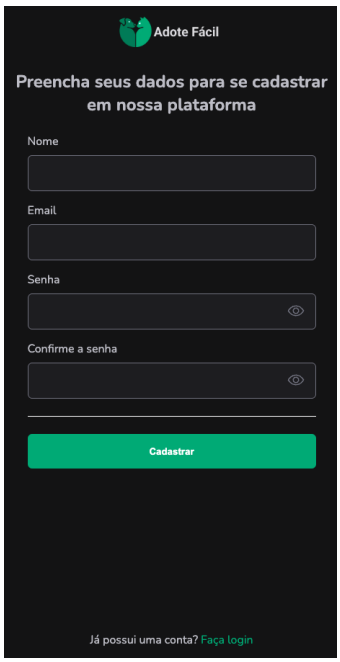
Fonte: Elaborado pelo autor

Figura 20 – Tela de *login* de usuário *web*



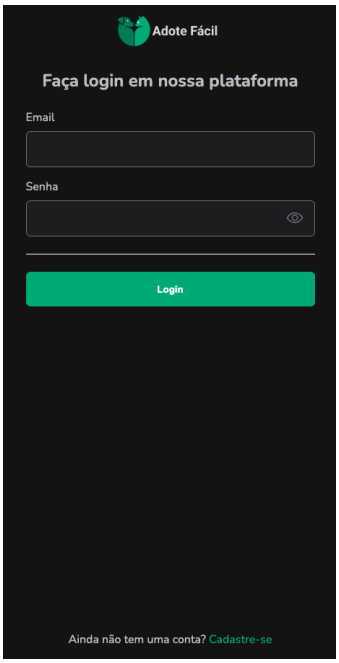
Fonte: Elaborado pelo autor

Figura 21 – Tela de cadastro de usuário *mobile*



Fonte: Elaborado pelo autor

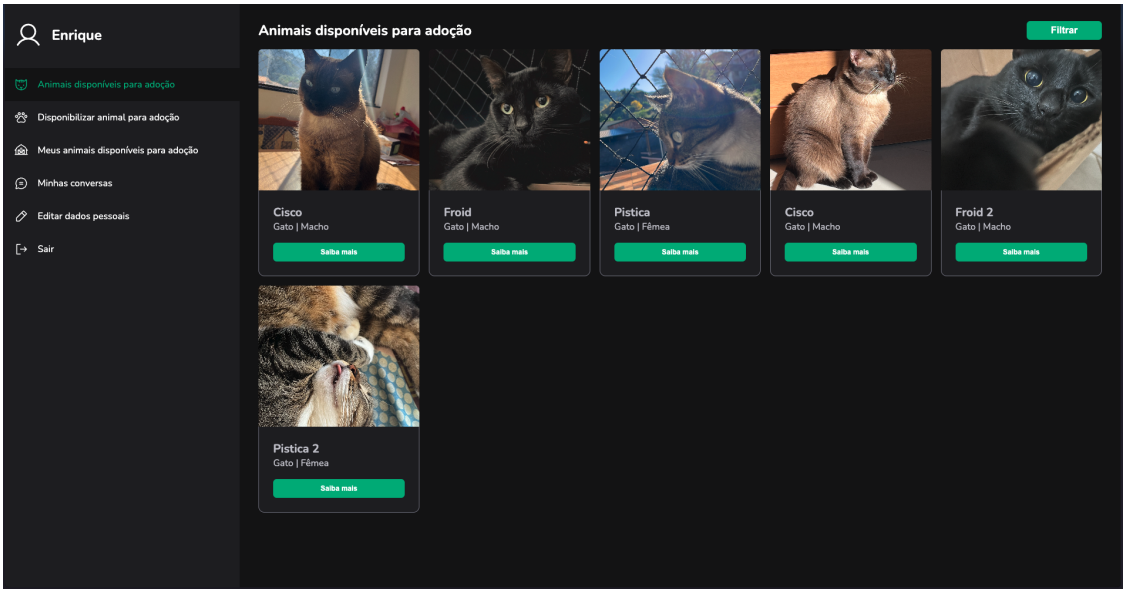
Figura 22 – Tela de *login* de usuário *mobile*



Fonte: Elaborado pelo autor

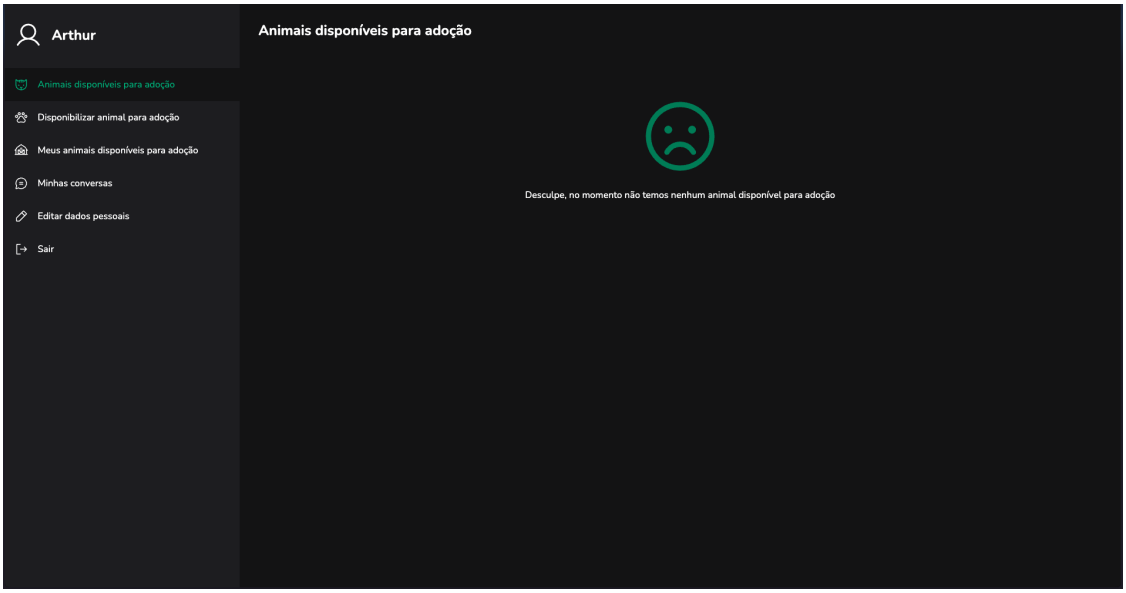
Ao acessar a plataforma, o usuário se depara com a tela de animais disponíveis para adoção (Figuras 23 e 25). Caso não tenham animais disponíveis, o sistema exibe a versão demonstrada nas Figuras 24 e 26.

Figura 23 – Tela de animais disponíveis *web*



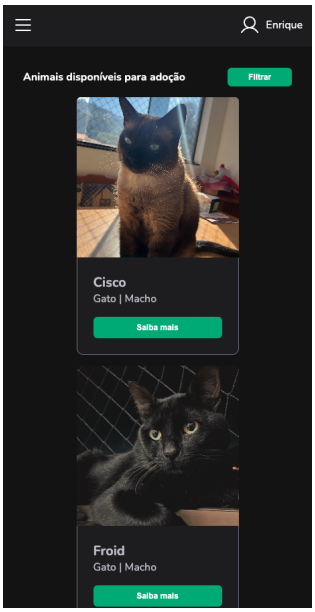
Fonte: Elaborado pelo autor

Figura 24 – Tela de animais disponíveis *web* vazia



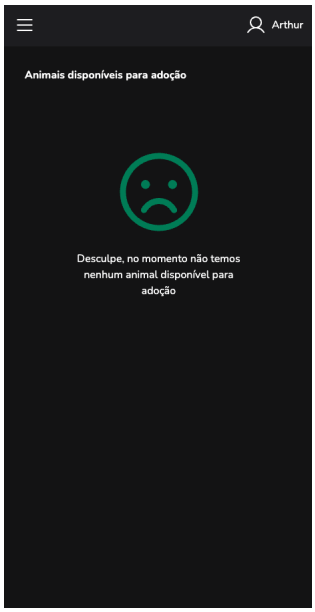
Fonte: Elaborado pelo autor

Figura 25 – Tela de animais disponíveis *mobile*



Fonte: Elaborado pelo autor

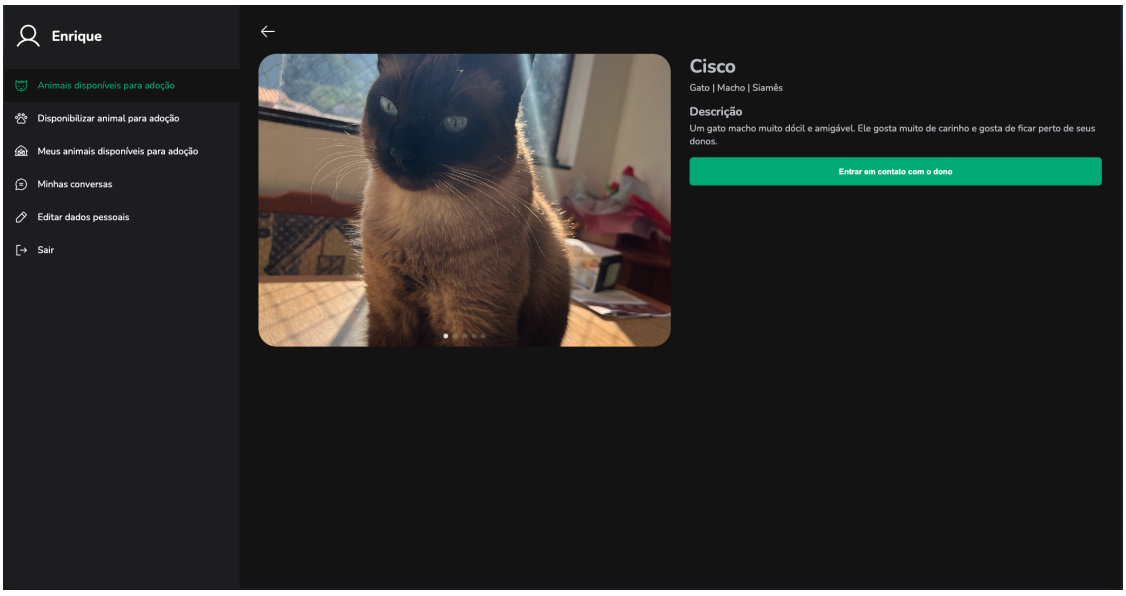
Figura 26 – Tela de animais disponíveis *mobile* vazia



Fonte: Elaborado pelo autor

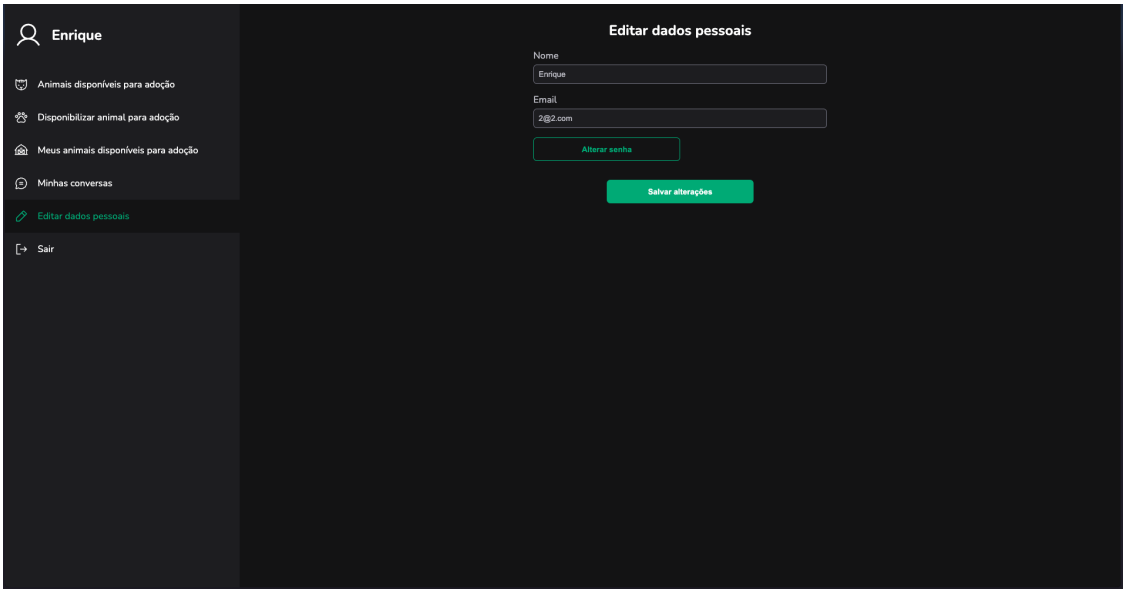
Ao clicar no botão de 'Saiba mais' exibido no *card* de cada animal, o usuário é redirecionado para a tela de detalhes do animal (Figuras 27 e 29). Caso queira, o usuário pode editar seus dados, tendo a possibilidade de alterar seu email, nome ou senha, clicando na opção 'Editar dados pessoais' no menu lateral. A tela de edição de dados pode ser conferida nas Figuras 28 e 30.

Figura 27 – Tela de detalhes do animal *web*



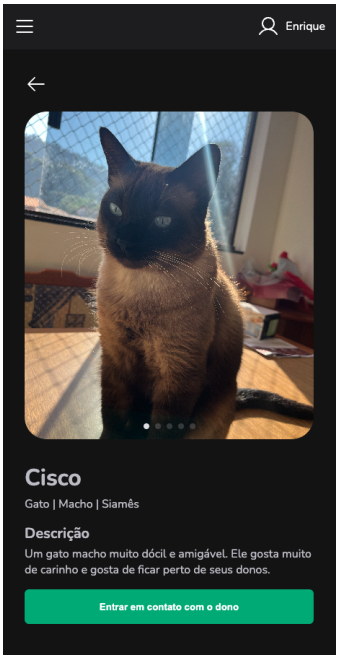
Fonte: Elaborado pelo autor

Figura 28 – Tela de editar dados do usuário *web*



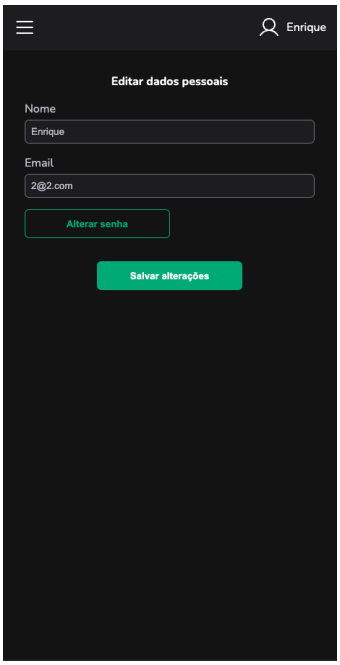
Fonte: Elaborado pelo autor

Figura 29 – Tela de detalhes do animal *mobile*



Fonte: Elaborado pelo autor

Figura 30 – Tela de editar dados do usuário *mobile*

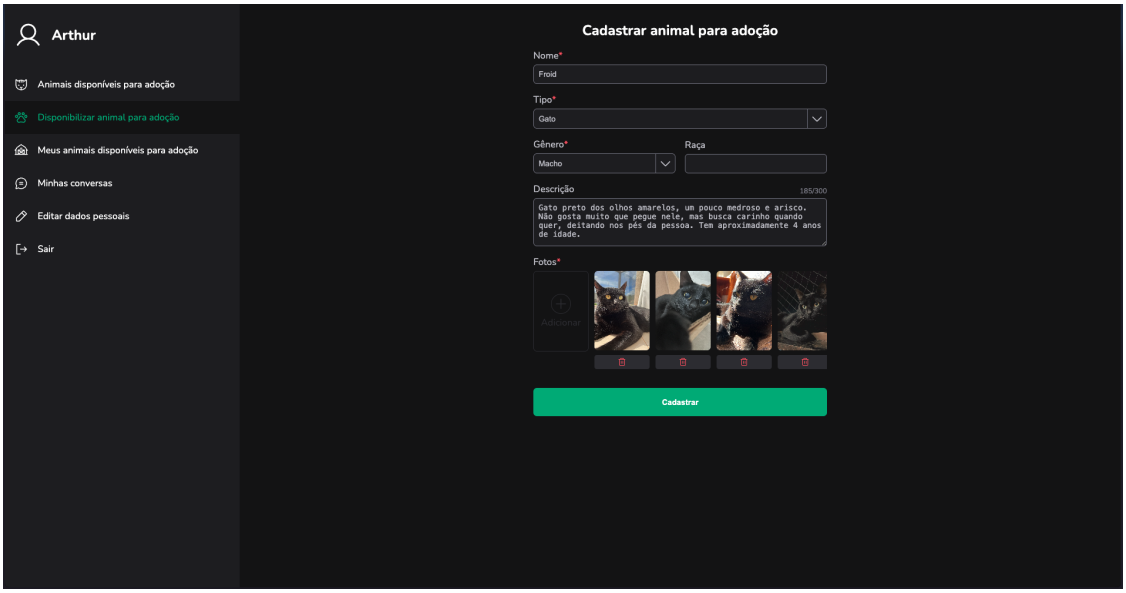


Fonte: Elaborado pelo autor

Para os usuários que desejam doar animais, o sistema apresenta a funcionalidade de cadastro de animais. Pelo menu lateral, o usuário pode selecionar a opção 'Disponibilizar animal para adoção' para ser redirecionado para o formulário de cadastro de animal (Figuras 31 e 33). Há também a listagem dos animais cadastrados pelo usuário, na opção

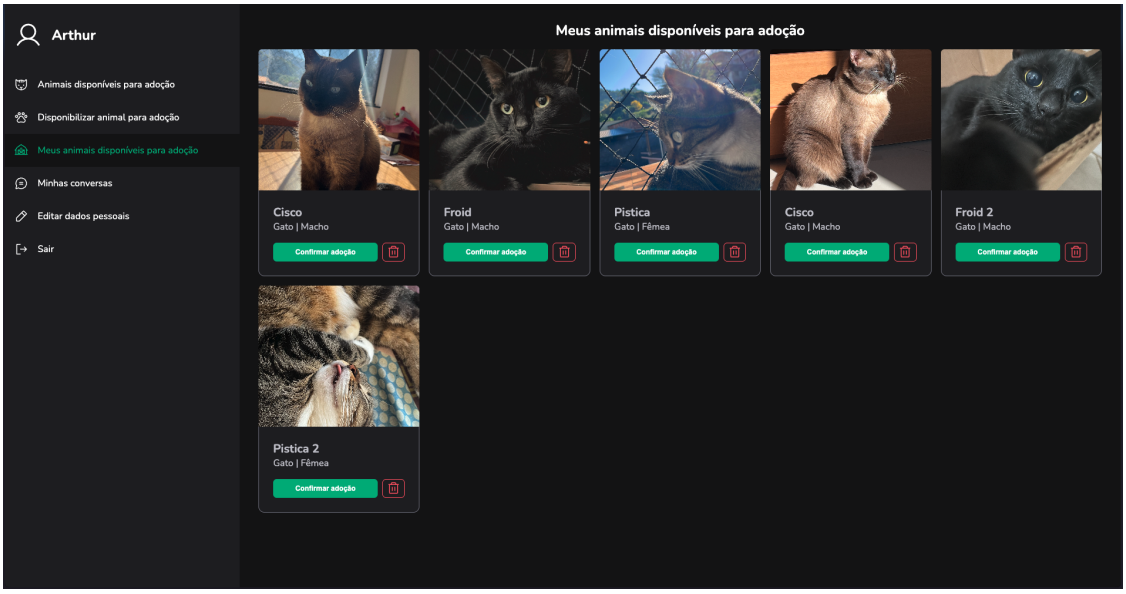
'Meus animais disponíveis para adoção' (Figuras 32 e 34). Essa listagem também apresenta uma versão vazia similar à tela de listagem de animais quando não há animais disponíveis para adoção, demonstrada nas Figuras 24 e 26.

Figura 31 – Tela de cadastro de animal para adoção *web*



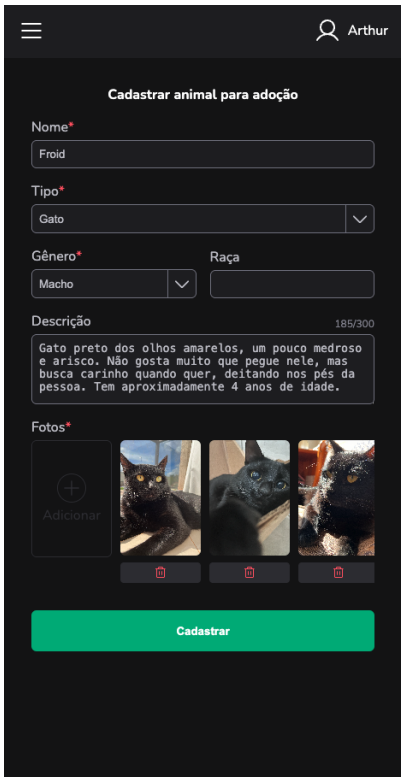
Fonte: Elaborado pelo autor

Figura 32 – Tela de meus animais *web*



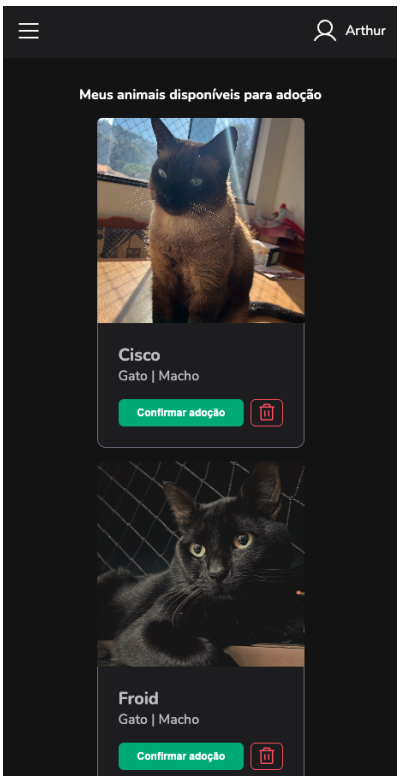
Fonte: Elaborado pelo autor

Figura 33 – Tela de cadastro de animal para adoção *mobile*



Fonte: Elaborado pelo autor

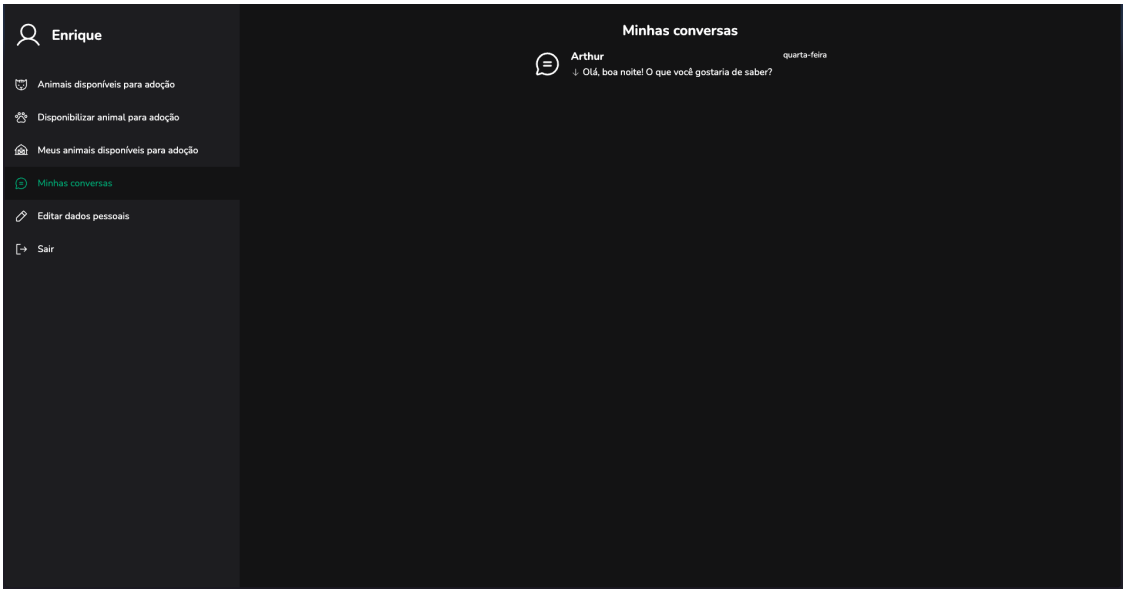
Figura 34 – Tela de meus animais *mobile*



Fonte: Elaborado pelo autor

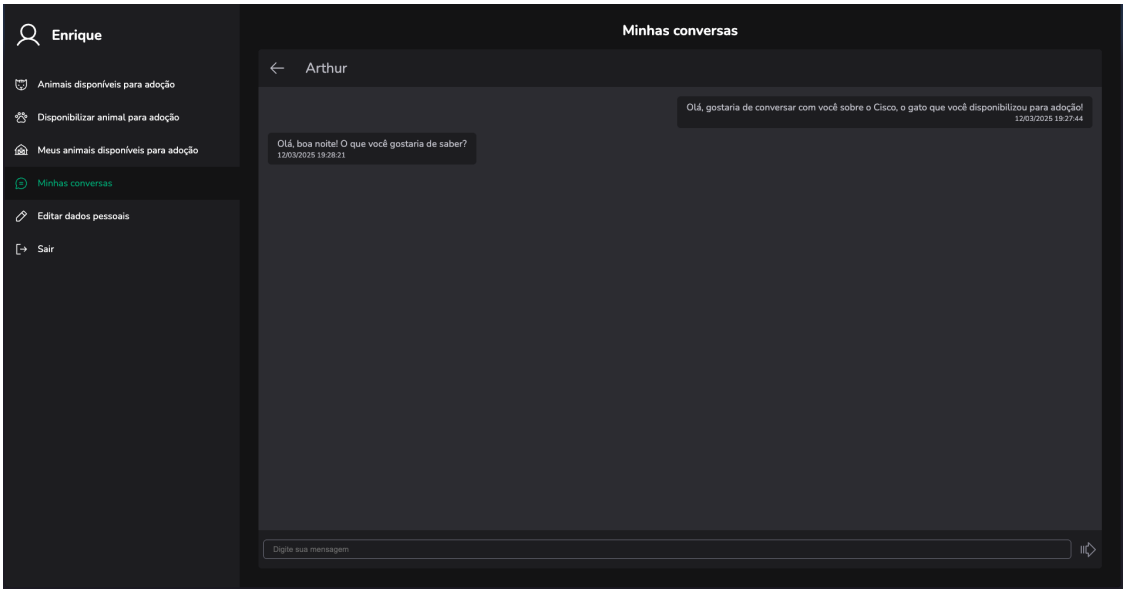
Os usuários também têm a opção de conversar entre si pela plataforma. Ao visualizar os detalhes do animal, o usuário pode clicar no botão 'Entrar em contato com o dono', para que o sistema o redirecione para um chat entre ele e o dono do animal. O usuário pode visualizar todas as suas conversas com outros usuários na tela de 'Minhas conversas' (Figuras 35 e 37), e ao clicar em uma conversa específica, ele visualiza o histórico de mensagens e pode enviar novas mensagens na tela de *chat* (Figuras 36 e 38).

Figura 35 – Tela de minhas conversas *web*



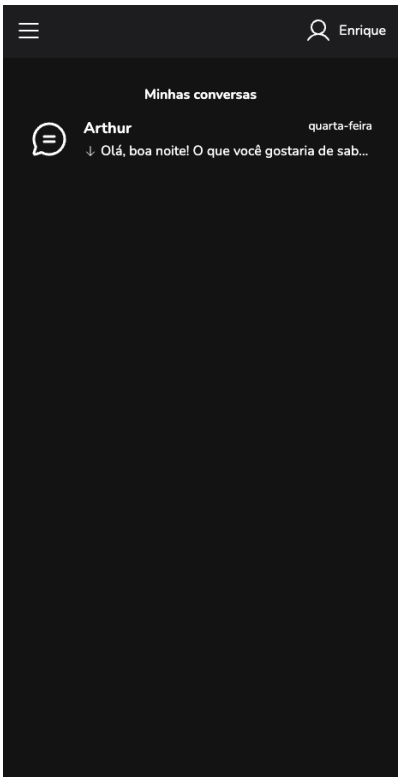
Fonte: Elaborado pelo autor

Figura 36 – Tela de *chat* entre usuários *web*



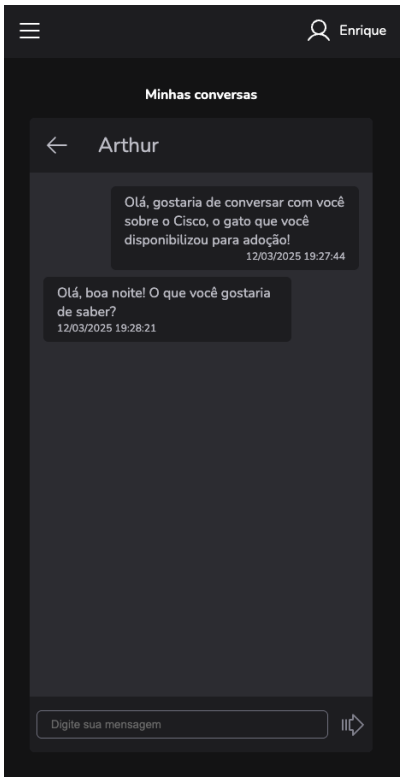
Fonte: Elaborado pelo autor

Figura 37 – Tela de minhas conversas *mobile*



Fonte: Elaborado pelo autor

Figura 38 – Tela de *chat* entre usuários *mobile*



Fonte: Elaborado pelo autor

Como demonstrado pelas imagens das telas do sistema, foi desenvolvido um *software* que atende ao objetivo inicial de simplificar o processo de adoção de animais. A plataforma facilita a visualização dos animais disponíveis para adoção e promove a comunicação direta entre adotantes e doadores por meio de um sistema de *chat*, tornando o processo mais ágil e eficiente.

Embora alguns detalhes definidos nos documentos de visão, requisitos, histórias de usuário e protótipos tenham sido ajustados ao longo do desenvolvimento, a maior parte do sistema manteve-se fiel às características inicialmente estabelecidas. Isso reflete um planejamento eficaz e um levantamento de requisitos bem executado, que permitiram uma execução sólida do projeto, com mudanças pontuais e controladas.

4.2 Repositório no GitHub

Como mencionado anteriormente, um repositório foi criado no GitHub[1] para armazenar o código-fonte do sistema publicamente, permitindo o acesso e a colaboração da comunidade. Nele, é possível encontrar todo o código do sistema, além de uma licença de *software* livre GNU/GLP-3.0, que garante sua livre utilização e modificação. Também

[1] https://github.com/ArthurEnrique15/adote-facil

está disponível um arquivo README, que contém a documentação das rotas do *backend* e um tutorial detalhado para a implantação do sistema, facilitando a integração e o uso da plataforma por outros desenvolvedores ou organizações.

4.3 Aplicação do tutorial para execução do sistema em um computador do LECoMP

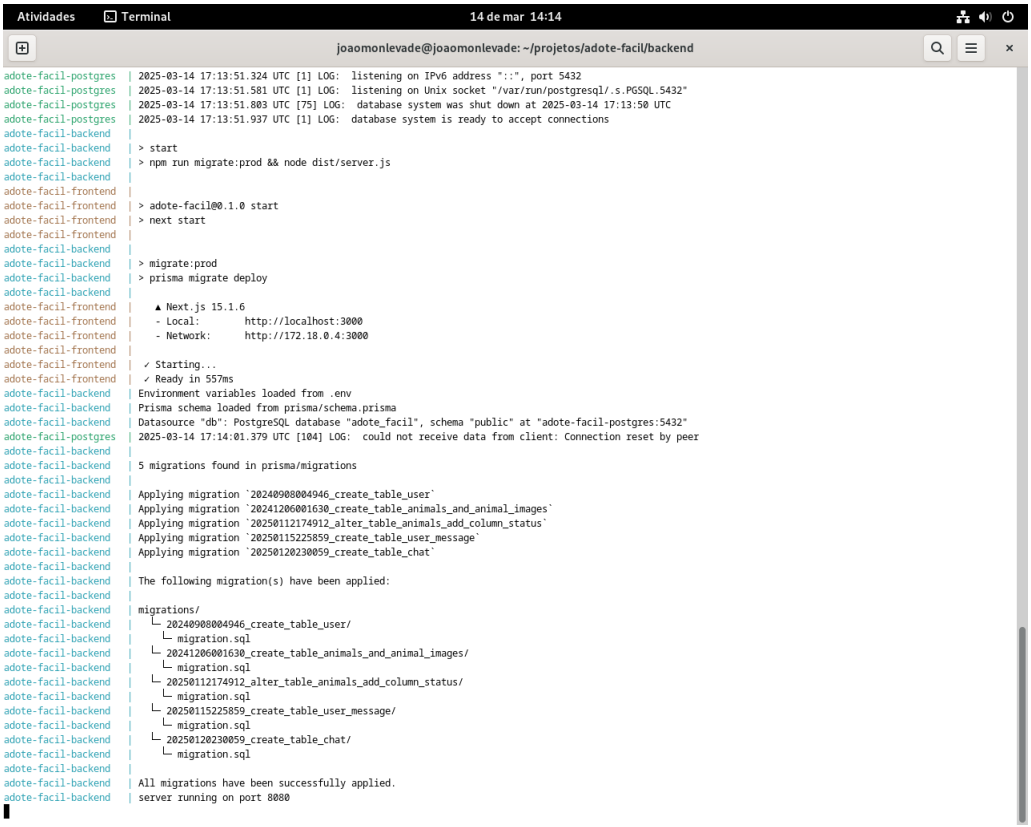
Após a finalização do sistema e do tutorial de implantação, foi decidido testar o processo de instalação seguindo o tutorial para executar o sistema em uma máquina com sistema operacional Debian do Laboratório de Estudos em Computação do Médio Piracicaba (LECoMP), no Instituto de Ciências Exatas e Aplicadas (ICEA).

Primeiramente, foram instaladas as ferramentas necessárias para a aplicação do tutorial, incluindo Git, Docker e o editor Visual Studio Code. Em seguida, o tutorial foi seguido, começando pelo clone do repositório, criação das variáveis de ambiente e execução do comando ***docker compose up*** no diretório ***/backend***. As Figuras 39 e 40 ilustram, respectivamente, os contêineres do sistema em execução no terminal e a interface do sistema sendo exibida no navegador Firefox[2].

Com isso, pode-se concluir que tanto o tutorial de implantação quanto a contêinerização do sistema foram bem-sucedidos. Apesar de o sistema ter sido desenvolvido em uma máquina com MacOS, o tutorial de implantação permitiu a execução perfeita da API em uma máquina com Debian GNU/Linux, comprovando a portabilidade e a eficácia da abordagem de contêinerização.

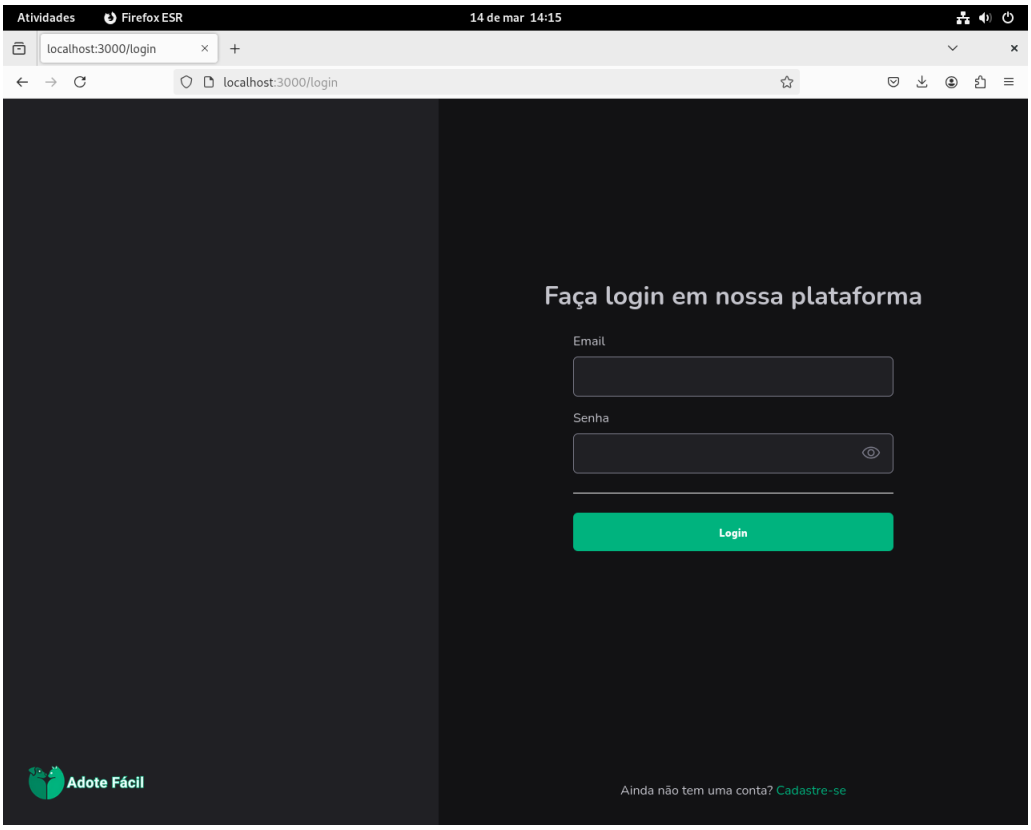
[2] https://www.mozilla.org/pt-BR/firefox/new/

Figura 39 – *Print* dos *logs* no terminal ao executar os contêineres do sistema no computador do LECoMP



Fonte: Elaborado pelo autor

Figura 40 – *Print* da tela de *login* do sistema executando no computador do LECoMP



Fonte: Elaborado pelo autor

5 Considerações finais

5.1 Conclusão

O objetivo deste trabalho foi o desenvolvimento da plataforma AdoteFácil, uma solução digital para facilitar o processo de adoção de animais, conectando adotantes e doadores de forma eficiente e simplificada. Ao longo do desenvolvimento, foi possível aplicar conceitos de engenharia de *software*, metodologias ágeis e tecnologias modernas, resultando em uma plataforma totalmente funcional.

O levantamento de requisitos foi um processo fundamental para a execução do projeto, realizado com base em uma análise detalhada das necessidades dos usuários, tanto doadores quanto adotantes de animais. O documento de visão, o documento de requisitos e as histórias de usuário forneceram uma base sólida para a implementação das funcionalidades, ao passo que os protótipos no Figma serviram de rascunho para visualizar como seria a interface do sistema. O progresso do desenvolvimento foi acompanhado utilizando a metodologia ágil Kanban por meio da ferramenta Jira, acompanhando os status das tarefas, o que permitiu que o sistema fosse desenvolvido de forma incremental, com foco na entrega de funcionalidades completas.

A escolha das tecnologias para o desenvolvimento do sistema foi orientada principalmente pela proficiência técnica do autor. As principais tecnologias utilizadas foram TypeScript, React, Next.js, Node.js, Express, PostgreSQL, PrismaORM e Jest, assim como Docker e Docker Compose para conteinerização da aplicação. O sistema foi concluído com a implementação de todas as funcionalidades essenciais para o processo de adoção de animais, incluindo o cadastro de animais, a busca por características específicas e a comunicação entre adotantes e doadores. Ao final do desenvolvimento, a plataforma foi disponibilizada publicamente como um projeto de *software* livre no GitHub, com uma documentação do sistema e tutorial de implantação, permitindo que outros indivíduos, grupos ou organizações possam adaptá-la às suas necessidades e incrementá-la com novas funcionalidades.

Em termos de contribuição para o campo de estudo, o projeto AdoteFácil foi construído utilizando um processo no qual foram aplicados conceitos teóricos e práticos em engenharia de *software*, como engenharia de requisitos, metodologias ágeis, testes, versionamento de código e conteinerização. Como resultado, foi produzida uma plataforma totalmente funcional que pode ser implantada e incrementada por outros indivíduos ou entidades, com potencial de ajudar a solucionar um problema social significativo: o abandono de animais. Portanto, conclui-se que o desenvolvimento do sistema AdoteFácil

alcançou os objetivos propostos, aplicando conceitos de engenharia de *software* para criar uma plataforma que atende às necessidades da adoção de animais.

5.2 Trabalhos Futuros

Como possibilidades para continuidade deste trabalho, destacam-se algumas iniciativas importantes. Inicialmente, sugere-se estabelecer parcerias com prefeituras, ONGs e outras entidades relacionadas à proteção animal, visando à implantação da plataforma em contextos reais, permitindo avaliar sua eficácia e impacto na prática. Além disso, pode-se realizar o desenvolvimento de módulos adicionais, adicionando funcionalidades que possam enriquecer ainda mais a experiência dos usuários e ampliar o alcance da plataforma. Por fim, outra possibilidade interessante é expandir o escopo da plataforma, integrando funcionalidades voltadas não somente para adoção, mas também para o bem-estar animal de maneira mais ampla, abrangendo denúncias, campanhas educativas e serviços relacionados ao cuidado dos animais.

Referências

ADOTE PETZ. *Adote Petz*. [S.l.], 2025. Disponível em: <https://www.adotepetz.com.br/>. Acesso em: 09 fev. 2025. Citado 3 vezes nas páginas 13, 17 e 18.

AGÊNCIA MINAS. *Governo de Minas promove políticas públicas que reforçam combate ao abandono de animais domésticos*. [S.l.], 2024. Disponível em: <https://www.agenciaminas.mg.gov.br/noticia/governo-de-minas-promove-politicas-publicas-que-reforcam-combate-ao-abandono-de-animais-domesticos>. Acesso em: 09 fev. 2025. Citado na página 14.

AHMAD, M. O.; MARKKULA, J.; OIVO, M. *Kanban in Software Development: A Systematic Literature Review*. [S.l.], 2013. Disponível em: <https://www.researchgate.net/publication/260739586_Kanban_in_Software_Development_A_Systematic_Literature_Review>. Acesso em: 15 fev. 2025. Citado 2 vezes nas páginas 26 e 27.

AMIGO NÃO SE COMPRA. *Amigo Não se Compra*. [S.l.], 2025. Disponível em: <https://www.amigonaosecompra.com.br>. Acesso em: 09 fev. 2025. Citado 3 vezes nas páginas 19, 20 e 21.

EXAME. *Abandono de animais aumentou cerca de 60% durante a pandemia*. [S.l.], 2021. Disponível em: <https://exame.com/bussola/abandono-de-animais-aumentou-cerca-de-60-durante-a-pandemia/>. Acesso em: 09 fev. 2025. Citado na página 13.

FREE SOFTWARE FOUNDATION. *A Quick Guide to the GNU General Public License (GPL) v3*. [S.l.], 2022. Disponível em: <https://www.gnu.org/licenses/quick-guide-gplv3.html>. Acesso em: 23 mar. 2025. Citado 2 vezes nas páginas 39 e 40.

FREE SOFTWARE FOUNDATION. *What is Free Software?* [S.l.], 2023. Disponível em: <https://www.gnu.org/philosophy/free-sw.html>. Acesso em: 23 mar. 2025. Citado 3 vezes nas páginas 23, 24 e 39.

IBGE. *Panorama do Censo 2022*. Brasil, 2023. Disponível em: <https://censo2022.ibge.gov.br/panorama/>. Acesso em: 12 fev. 2025. Citado 2 vezes nas páginas 19 e 20.

O POPULAR JM. *Prefeitura de João Monlevade lança espaço dedicado a adoção de animais em seu site*. [S.l.], 2024. Disponível em: <https://opopularjm.com.br/prefeitura-de-joao-monlevade-lanca-espaco-dedicado-a-adocao-de-animais-em-seu-site>. Acesso em: 09 fev. 2025. Citado na página 22.

PEREIRA, I. M. *Domain Driven Design: Trabalhando com histórias de usuário*. [S.l.], 2014. Disponível em: <https://www.devmedia.com.br/engenharia-de-requisitos-agil/31871>. Acesso em: 12 mar. 2025. Citado na página 31.

PREFEITURA MUNICIPAL DE JOÃO MONLEVADE. *Seção de animais disponíveis para adoção no site da prefeitura municipal de João Monlevade*. João Monlevade, MG, Brasil, 2025. Disponível em: <https://pmjm.mg.gov.br/adocao>. Acesso em: 09 fev. 2025. Citado na página 22.

RIVA, A. et al. *Aplicativo como plataforma de integração entre sociedade e animais domésticos*. Porto Alegre, RS, Brasil, 2022. 21–31 p. Disponível em: <https://sol.sbc.org.br/index.php/ercomprs/article/view/20403>. Acesso em: 09 fev. 2025. Citado 3 vezes nas páginas 16, 17 e 22.

SOMMERVILLE, I. *Engenharia de Software*. São Paulo, Brasil, 2019. Disponível em: <https://plataforma.bvirtual.com.br/Leitor/Publicacao/168127/pdf/0>. Acesso em: 15 fev. 2025. Citado na página 24.

VALENTE, M. T. *Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade*. São Paulo, Brasil, 2020. Disponível em: <https://engsoftmoderna.info/>. Acesso em: 15 fev. 2025. Citado 5 vezes nas páginas 24, 25, 26, 27 e 28.

Apêndices

APÊNDICE A – Materiais elaborados pelo autor

A.1 Documento de visão

As próximas páginas são um apêndice do documento de visão elaborado no projeto.

Universidade Federal de Ouro Preto

AdoteFácil
Visão

Versão 0.1

Histórico da Revisão

Data	Versão	Descrição	Autor
15/04/2024	0.1	Versão inicial	Arthur Enrique

Índice

1. Introdução 5
    1.1 Objetivo 5
    1.2 Escopo 5
    1.3 Definições, Acrônimos e Abreviações 5
    1.4 Referências 5
    1.5 Visão Geral 5
2. Posicionamento 5
    2.1 Oportunidade de Negócio 5
    2.2 Declaração do Problema 5
    2.3 Declaração da Posição do Produto 5
3. Descrições do Investidor e do Usuário 6
    3.1 Demográficos de Mercado 6
    3.2 Resumo do Investidor 6
    3.3 Resumo de Usuários 6
    3.4 Ambiente do Usuário 6
    3.5 Perfis de Usuários 6
        3.5.1 Doador 7
        3.5.2 Adotante 7
    3.6 Necessidades Principais do Investidor ou Usuário 7
    3.7 Alternativas e Competição 7
4. Visão Geral do Produto 8
    4.1 Perspectiva do Produto 8
    4.2 Resumo de Recursos 8
    4.3 Premissas e Dependências 8
    4.4 Custo e Preço 8
    4.5 Licença e Instalação 8
5. Recursos do Produto 8
    5.1 Cadastro de usuários 8
    5.2 Atualização de informações do usuário 8
    5.3 Login de usuários 8
    5.4 Cadastro de animais disponíveis para adoção 8
    5.5 Exibir animais disponíveis para adoção 8
    5.6 Contato entre adotante e doador 8
    5.7 Confirmação de adoção de animal 9
6. Restrições 9
7. Intervalos de Qualidade 9
8. Precedência e Prioridade 9
9. Outros Requisitos de Produto 9

9.1 Padrões Aplicáveis 9
9.2 Requisitos do Sistema 9
9.3 Requisitos de Desempenho 9
9.4 Requisitos Ambientais 9
10. Requisitos de Documentação 9
10.1 Manual do Usuário 9
10.2 Ajuda On-line 9
10.3 Guias de Instalação, Configuração e Arquivo LEIA-ME 10
10.4 Etiquetagem e Empacotamento 10

Visão

1. Introdução

1.1 Objetivo

O objetivo deste documento é coletar, analisar e definir necessidades e recursos de alto nível do sistema AdoteFácil. Ele está focalizado nos recursos necessários aos usuários de destino e por que essas necessidades existem. Os detalhes de como o AdoteFácil atende essas necessidades são explicados no documento de requisitos e suplementares.

1.2 Escopo

Este documento Visão está associado apenas ao projeto AdoteFácil e não possui outros projetos associados.

1.3 Definições, Acrônimos e Abreviações

- ONG: Organização Não Governamental
- MVC: Model, View, Controller

1.4 Referências

Este documento não menciona ou refere outros documentos.

1.5 Visão Geral

O documento descreve a resolução do problema que o sistema apresenta, quais os potenciais usuários, uma visão geral do sistema, quais recursos o software terá, e por fim alguns requisitos como padrões, desempenho, etc.

2. Posicionamento

2.1 Oportunidade de Negócio

O processo de adoção de animais é extremamente complicado, principalmente da parte do doador, que precisa muitas vezes pagar para que uma clínica veterinária coloque um animal para adoção, e muitas vezes tais clínicas nem aceitam o animal. Tendo isso em vista, o projeto surge com o objetivo de simplificar esse processo e por consequência aumentar o número de adoções, melhorando a qualidade de vida dos animais em situações vulneráveis. Vale ressaltar que o foco do projeto é no processo de adoção em si, portanto não haverá qualquer tipo de suporte ou acompanhamento de animais que já passaram pelo processo de adoção na plataforma.

2.2 Declaração do Problema

O problema de	dificuldade no processo de adoção de animais
afeta	a sociedade no geral
o impacto é o seguinte	mais animais em situação de vulnerabilidade quando poderiam estar sendo cuidados em algum lar
uma solução bem-sucedida seria	simplificação do processo de adoção, não cobrando taxas para adoção de animais, oferecendo uma plataforma para divulgação de animais disponíveis para adoção

2.3 Declaração da Posição do Produto

Para	cidadãos maiores de idade no geral

Quem	pessoas que querem adotar ou doar um animal doméstico
O AdoteFácil	é uma plataforma online (site)
Que	permite a disponibilização de animais para adoção e ao mesmo tempo a adoção de animais que estiverem disponíveis
Diferente	gratuito
Nosso produto	facilita o processo de adoção e doação de animais, de forma totalmente gratuita para todas as partes envolvidas

3. Descrições do Investidor e do Usuário

3.1 Demográficos de Mercado

Demográficos de mercado não se aplicam ao projeto.

3.2 Resumo do Investidor

O projeto não possui investidores.

3.3 Resumo de Usuários

Doadores de animais	Usuários principais, sem eles o sistema não tem utilidade	Disponibiliza algum animal para adoção. Também pode adotar.	
Adotantes de animais	Também são usuários principais, juntamente com os doadores	Adota algum animal colocado para adoção por algum doador, porém como os doadores também pode colocar animais para adoção	

3.4 Ambiente do Usuário

O usuário acessa o sistema de qualquer lugar que desejar, uma vez que é um site e pode ser acessado de qualquer lugar desde que exista uma conexão com a internet. O foco principal é o acesso via celulares, uma vez que os animais disponíveis para adoção precisam de fotos, e pensando na usabilidade, é mais fácil realizar o upload de fotos na plataforma via smartphone.

Cada usuário pode ser adotante ou doador ou ambos. O sistema deve permitir que as tarefas executadas pelos usuários possam ser concluídas por eles mesmos, de forma simples. Dado que as tarefas são relativamente simples, não devem ser necessárias mais pessoas ou outras formas de ajuda para o usuário conseguir usar o sistema.

Com relação ao tempo, as tarefas podem ser executadas relativamente rápido, em cerca de 10-15 minutos, dado que não são muito complexas. Consistem basicamente em criar uma conta, logar e disponibilizar um animal para adoção ou navegar pelos animais disponíveis até encontrar um que seja do agrado do usuário. A depender do usuário, estas últimas tarefas, principalmente a de adotar um animal, podem demorar mais tempo, uma vez que adotar um animal é uma decisão importante, então o usuário pode querer consultar outras pessoas e pensar mais tempo a respeito de sua decisão.

3.5 Perfis de Usuários

3.5.1 Doador

Representante	Pessoa comum interessada em doar um animal.
Descrição	Pessoa que possui um animal doméstico que deseja doar para outra pessoa.
Tipo	Por ser extremamente abrangente, é possível considerar que este usuário possui pouca afinidade com tecnologias, com pouco ou nenhum background técnico, e sendo um usuário casual.
Responsabilidades	Disponibiliza algum animal para adoção. Também pode adotar.
Critérios de Êxito	O êxito deste usuário ao utilizar o sistema é conseguir que alguém adote os animais que o mesmo disponibilizou para adoção.
Envolvimento	Não está envolvido.
Distribuíveis	Não.
Comentários / Problemas	A pouca afinidade com tecnologia, dado que o público alvo é qualquer pessoa que queira adotar ou doar um animal, pode ser um problema na usabilidade do usuário.

3.5.2 Adotante

Representante	Pessoa comum interessada em adotar um animal.
Descrição	Pessoa que deseja adotar um animal doméstico.
Tipo	Por ser extremamente abrangente, é possível considerar que este usuário possui pouca afinidade com tecnologias, com pouco ou nenhum background técnico, e sendo um usuário casual
Responsabilidades	Adota algum animal colocado para adoção por algum doador, porém como os doadores também pode colocar animais para adoção
Critérios de Êxito	O êxito deste usuário ao utilizar o sistema é conseguir adotar um ou mais animais.
Envolvimento	Não está envolvido.
Distribuíveis	Não.
Comentários / Problemas	A pouca afinidade com tecnologia, dado que o público alvo é qualquer pessoa que queira adotar ou doar um animal, pode ser um problema na usabilidade do usuário.

3.6 Necessidades Principais do Investidor ou Usuário

O principal problema dos usuários do sistema é a dificuldade de doar ou adotar animais domésticos na cidade de João Monlevade. As clínicas veterinárias da cidade não disponibilizam seus espaços para abrigar animais em situação de vulnerabilidade sem cobrar taxas por isso, além de também cobrarem taxas para colocar algum animal para adoção, os quais só podem ser filhotes, pois as clínicas não aceitam animais adultos para doação, uma vez que é mais difícil um animal mais velho ser adotado.

Atualmente, existem algumas ONGs na cidade que tentam por meio de vaquinhas e divulgação nas redes sociais conseguir ajuda para animais em situação de vulnerabilidade, ou até mesmo achar algum adotante para cuidar do animal. Porém, essa solução não é muito viável, pois as ONGs não possuem muito alcance nas redes, o que torna extremamente difícil conseguir ajuda para os animais.

Com este projeto, a ideia é facilitar esse processo de adoção de animais na cidade, dando autonomia para que as pessoas interessadas em doar ou adotar animais tenham uma forma de se comunicar facilmente.

3.7 Alternativas e Competição

Não se aplica ao projeto.

4. Visão Geral do Produto

4.1 Perspectiva do Produto

O produto é um sistema totalmente independente, e não se comunica ou interage com outros sistemas.

4.2 Resumo de Recursos

Tabela 4-1 - Sistema para adoção de animais

Benefício do Cliente	Recursos de Suporte
Os clientes conseguem adotar e doar animais de forma muito simples e prática.	O sistema permite que em poucos passos os clientes possam doar ou adotar animais.
Aumento significativo de diversidade na hora de escolher um animal para adotar	A visualização de vários animais cadastrados pelos doadores faz com que os clientes possam filtrar e escolher um animal com as características desejadas
Comunicação facilitada entre adotante e doador	Clientes podem visualizar informações dos donos dos animais disponíveis para adoção e entrar em contato

4.3 Premissas e Dependências

- O sistema estará disponível a todo momento
- Os usuários do sistema estão interessados em adotar ou doar animais
- Existem muitos animais em situação de abandono, sem um lar

4.4 Custo e Preço

Não haverá custos uma vez que o sistema não será efetivamente implantado, apenas desenvolvido e disponibilizado livremente para implantação.

4.5 Licença e Instalação

O software deve ter uma licença de software livre que será emitida próximo ao fim do desenvolvimento.

5. Recursos do Produto

5.1 Cadastro de usuários

Um cadastro simples que permita aos usuários criar uma conta na plataforma, preenchendo suas informações como email, senha, e alguns dados de contato como telefone. O cadastro pode ser feito também integrado com a conta do google.

5.2 Atualização de informações do usuário

Caso o cliente deseje, também pode atualizar suas informações de contato.

5.3 Login de usuários

Login dos usuários na conta criada na plataforma.

5.4 Cadastro de animais disponíveis para adoção

Os usuários que desejarem doar algum animal podem cadastrá-los na plataforma, preenchendo informações como idade, nome se houver, adicionar fotos, etc.

5.5 Exibir animais disponíveis para adoção

Os usuários que desejarem adotar algum animal podem navegar pelos animais disponíveis para adoção e filtrar a exibição para encontrar um animal com as características que quiserem.

5.6 Contato entre adotante e doador

A plataforma terá um chat que poderá ser usado entre o adotante e o doador para se comunicarem.

5.7 Confirmação de adoção de animal

Quando um animal for adotado, o doador pode atualizar seu status na plataforma, confirmando que ele foi adotado.

6. Restrições

- Evitar uma interface muito complexa.
- O sistema não será implantado.
- Manter a documentação simples e de fácil entendimento, porém completa.
- O escopo do projeto está restrito ao processo de adoção desde o contato do adotante com o doador até que o animal esteja nas mãos de seu novo dono, portanto o projeto e o sistema não se responsabilizam pelo animal após a adoção

7. Intervalos de Qualidade

O sistema deve priorizar os seguintes atributos de qualidade: confiabilidade, portabilidade, usabilidade e eficiência.

8. Precedência e Prioridade

Ordem de prioridade dos recursos do sistema:

1. Cadastro de usuários
2. Login de usuários
3. Cadastro de animais disponíveis para adoção
4. Exibir animais disponíveis para adoção
5. Confirmação de adoção de animal
6. Atualização de informações do usuário
7. Contato entre adotante e doador

9. Outros Requisitos de Produto

9.1 Padrões Aplicáveis

O sistema deve estar em conformidade com os padrões MVC.

9.2 Requisitos do Sistema

Os requisitos para acessar a plataforma são os mesmos para executar um navegador, uma vez que o sistema será projetado para a web, e portanto também é necessário ter acesso à internet.

9.3 Requisitos de Desempenho

O sistema não pode ter tempos de carregamento e tempos de resposta muito longos para não afetar a usabilidade do usuário final. Além disso, deve suportar o acesso simultâneo de milhares de pessoas sem travamentos ou bugs.

9.4 Requisitos Ambientais

O sistema, ao ser implantado, pode ser acessado de qualquer lugar com acesso à internet, então não existem muitas restrições relacionadas ao ambiente de acesso do usuário. Com relação a erros, inicialmente não haverá formas avançadas de tratá-los e recuperá-los, apenas logs para identificação.

10. Requisitos de Documentação

10.1 Manual do Usuário

Haverá um vídeo de apresentação do sistema explicando que problema ele resolve e quais as principais funcionalidades.

10.2 Ajuda On-line

Não haverá um sistema de ajuda online.

10.3 Guias de Instalação, Configuração e Arquivo LEIA-ME

O arquivo LEIAME será uma documentação extremamente detalhada do software, composto por uma descrição de todas as suas funcionalidades e um tutorial de implantação do sistema.

10.4 Etiquetagem e Empacotamento

O sistema deve ter uma identidade visual de acordo com os padrões visuais do logotipo corporativo. Os ícones também devem ser padronizados, assim como a UI no geral, mantendo sempre os elementos gráficos no mesmo padrão.

A.2 Documento de requisitos

As próximas páginas são um apêndice do documento de requisitos elaborado no projeto.

Universidade Federal de Ouro Preto

AdoteFácil

Especificação de Requisitos de Software

Versão <0.1>

Histórico da Revisão

Data	Versão	Descrição	Autor
01/05/2024	0.1	Versão inicial	Arthur Enrique
26/05/2024	0.2	Remoção da funcionalidade de demonstrar interesse no animal e adição da funcionalidade de listagem de animais	Arthur Enrique

Índice

1. Introdução **5**
1.1 Objetivo **5**
1.2 Escopo **5**
1.3 Definições, Acrônimos e Abreviações **5**
1.4 Referências **5**
1.5 Visão Geral **5**
2. Descrição Geral **5**
3. Requisitos Específicos **6**
3.1 Funcionalidade **6**
3.1.1 Cadastro de usuário **6**
3.1.2 Login de usuário 6
3.1.3 Atualização de informações do usuário 6
3.1.4 Disponibilização de animal para adoção 6
3.1.5 Listagem de animais disponíveis para adoção 6
3.1.6 Chat entre possível adotante e doador 6
3.1.7 Confirmação de adoção 7
3.2 Utilidade **7**
3.2.1 Tempo de Treinamento 7
3.2.2 Tempo de Tarefa Mensurável 7
3.3 Confiabilidade **7**
3.4 Desempenho **7**
3.5 Suportabilidade **7**
3.5.1 Padrões de código 7
3.6 Restrições de Design **8**
3.6.1 Padrão da API 8
3.6.2 Linguagem de programação 8
3.6.3 Padrão de código 8
3.6.4 Padrão de deploy 8
3.7 Documentação do Usuário On-line e Requisitos do Sistema de Ajuda **8**
3.8 Componentes Comprados **8**
3.9 Interfaces **8**
3.9.1 Interfaces com o Usuário **8**
3.9.2 Interfaces de Hardware **8**
3.9.3 Interfaces de Software **8**
3.9.4 Interfaces de Comunicações **8**

3.10 Requisitos de Licença **8**

3.11 Observações Legais, sobre Direitos Autorais e Outras Observações **8**

3.12 Padrões Aplicáveis **9**

4. Informações de Suporte **9**

Especificação de Requisitos de Software

1. Introdução

1.1 Objetivo

Este documento descreve completamente o comportamento externo do sistema AdoteFácil. Descreve também requisitos não funcionais, restrições de design e outros fatores necessários para fornecer uma descrição completa e abrangente dos requisitos para o software. O objetivo deste documento é fornecer informações detalhadas sobre o sistema, e servir como um guia e base para o desenvolvimento do projeto.

1.2 Escopo

O sistema ao qual este documento se refere será criado para resolver a dificuldade no processo de adoção, fornecendo uma plataforma para que possíveis adotantes possam buscar animais que desejam adotar, enquanto possíveis doadores podem cadastrar animais para adoção. A plataforma também disponibilizará uma forma de contato entre ambas as partes, simplificando o processo de adoção como um todo.

1.3 Definições, Acrônimos e Abreviações

O documento não apresenta definições, acrônimos e abreviações.

1.4 Referências

Este documento não menciona ou refere outros documentos.

1.5 Visão Geral

O documento especifica detalhadamente os requisitos do sistema a ser desenvolvido, apresentando requisitos funcionais, de utilidade, confiabilidade, desempenho, entre outros, com o objetivo de servir de base para o desenvolvimento do projeto.

2. Descrição Geral

O processo de adoção de animais é extremamente complicado, principalmente da parte do doador, que precisa muitas vezes pagar para que uma clínica veterinária coloque um animal para adoção, e muitas vezes tais clínicas nem aceitam o animal. Tendo isso em vista, o projeto surge com o objetivo de simplificar esse processo e por consequência aumentar o número de adoções, melhorando a qualidade de vida dos animais em situações vulneráveis.

Assim, os usuários do sistema são pessoas comuns que desejam adotar ou doar animais. Os doadores disponibilizam animais para adoção na plataforma, enquanto os adotantes sinalizam interesse em adotar algum animal disponível e podem entrar em contato com o doador por dentro da plataforma mesmo.

As principais funções do sistema são o cadastro e login de usuários, a disponibilização de animais para adoção, o contato entre doador e adotante por meio de um chat de mensagens, e a confirmação de adoção de um animal.

O sistema não será implantado, ele será de código aberto e possuirá uma documentação completa (porém simples e de fácil entendimento) para que possíveis organizações (como por exemplo uma prefeitura) possam ler a documentação e implantar o software caso queira.

3. Requisitos Específicos

3.1 Funcionalidade

3.1.1 Cadastro de usuário

Para se cadastrar na plataforma, os usuários têm duas opções: preencher um formulário com suas informações de email e senha ou utilizar uma conta do Google. No primeiro caso, o sistema deve fornecer um formulário de registro padrão, onde os usuários inserem seu email e senha desejados. As senhas devem ser criptografadas antes de serem armazenadas no banco de dados para garantir a segurança dos dados. Além disso, é essencial realizar a validação dos campos para garantir que o email seja único e esteja em um formato válido. Para a segunda opção, o sistema deve oferecer a funcionalidade de cadastro através do Google, permitindo que os usuários autorizem o acesso às suas informações básicas, como email e nome, para facilitar o processo de registro.

3.1.2 Login de usuário

Após o cadastro, os usuários podem acessar a plataforma através do login. Existem duas formas de login disponíveis: via email e senha ou utilizando a conta do Google. Para o login via email e senha, os usuários devem ser direcionados para uma página de login onde inserem suas credenciais previamente cadastradas. O sistema deve autenticar o usuário comparando as informações inseridas com as armazenadas no banco de dados. No caso do login via Google, os usuários devem ter a opção de se autenticar utilizando sua conta do Google, sendo redirecionados para a página de autorização do Google para validar suas credenciais.

3.1.3 Atualização de informações do usuário

Os usuários devem ter a capacidade de atualizar suas informações de perfil a qualquer momento. Isso inclui a possibilidade de editar o email, senha e as preferências de adoção/doação. As alterações realizadas pelos usuários devem ser refletidas imediatamente no banco de dados, garantindo que suas informações estejam sempre atualizadas. Essa funcionalidade permite que os usuários mantenham seu perfil atualizado de acordo com suas necessidades e preferências.

3.1.4 Disponibilização de animal para adoção

Para disponibilizar um animal para adoção, os usuários devem preencher um formulário com informações detalhadas sobre o animal, incluindo espécie, idade, sexo, raça, temperamento, histórico médico, entre outros. Além disso, deve ser possível fazer o upload de imagens do animal para fornecer uma representação visual. Essa informação será crucial para os potenciais adotantes conhecerem melhor o animal e decidirem se desejam adotá-lo.

3.1.5 Listagem de animais disponíveis para adoção

Os usuários da plataforma devem ser capazes de navegar por uma listagem detalhada de animais disponíveis para adoção, utilizando filtros como espécie, idade, sexo, raça e tamanho para refinar a busca. Cada animal tem um perfil completo com descrições e fotos, proporcionando informações essenciais para os adotantes.

3.1.6 Chat entre possível adotante e doador

Para facilitar a comunicação entre os possíveis adotantes e os doadores de animais, o sistema deve fornecer um sistema de mensagens interno na plataforma. Isso permitirá que os usuários conversem entre si de forma privada, discutindo detalhes sobre o animal, trocando informações e esclarecendo dúvidas. Essa funcionalidade é essencial para promover uma interação eficaz entre as partes envolvidas no processo de adoção.

3.1.7 Confirmação de adoção

Após o adotante e o doador concordarem com os termos da adoção através do sistema de mensagens internas da plataforma, o doador terá a opção de confirmar a adoção diretamente pelo sistema. Esta confirmação será uma etapa simples, onde o doador indicará no sistema que a adoção foi acordada e aceita. Esta abordagem simplificada visa agilizar o processo de adoção, tornando-o mais conveniente para ambas as partes envolvidas.

3.2 Utilidade

3.2.1 Tempo de Treinamento

O software deve ser intuitivo o suficiente para que um usuário normal se familiarize com as operações básicas em até 15 minutos de treinamento. Isso inclui as funcionalidades essenciais, como cadastro, login e busca por animais disponíveis. Por outro lado, um usuário potente, que busca utilizar funcionalidades mais avançadas, como o chat entre adotante e doador, pode precisar de até 30 minutos para se tornar produtivo. Esse tempo de treinamento deve ser considerado durante o desenvolvimento do sistema, garantindo que a curva de aprendizado seja suave e eficiente para todos os tipos de usuários.

3.2.2 Tempo de Tarefa Mensurável

O tempo médio para um usuário cadastrar um animal para adoção não deve exceder 5 minutos. Isso significa que o processo de cadastro deve ser simplificado e otimizado para que os usuários possam completá-lo rapidamente, sem complicações desnecessárias. Da mesma forma, o tempo médio para um usuário demonstrar interesse em um animal disponível para adoção não deve ultrapassar 2 minutos, enquanto o doador deve confirmar uma adoção pelo sistema em até 1 minuto. Esses tempos de tarefa ajudam a manter o fluxo do processo de adoção de forma eficiente e sem demoras.

3.3 Confiabilidade

É sempre importante que um sistema tenha formas eficazes de tratar erros. Porém, como o projeto não será implantado, conceitos que abrangem tempo entre falhas e tempo de recuperação de falhas não se aplicam ao projeto.

3.4 Desempenho

Muitos requisitos de desempenho estão mais relacionados com o uso final do software, ao ambiente que ele será implantado. Como o projeto não engloba esse escopo, podemos apenas definir que seria ideal para os usuários ter um tempo de resposta relativamente baixo, com um tempo médio inferior a 2 segundos e máximo de 5 segundos, para qualquer operação realizada dentro do sistema. Conceitos como capacidade de processamento e uso de recursos dependem da quantidade de usuários finais do sistema e quanto tempo quem estiver implantando deseja que o sistema esteja disponível, portanto não se aplicam aqui.

3.5 Suportabilidade

3.5.1 Padrões de código

Utilizar o padrão MVC para codificação do sistema. Sempre usar classes no backend e evitar o uso de funções soltas, uma vez que a linguagem de programação usada será Typescript, que permite o uso de programação funcional. Manter a nomenclatura dos arquivos o mais próxima possível do nome da classe do arquivo, e escrever o nome todo em minúsculas com as palavras separadas por hífens. Nos nomes de classes, variáveis, funções e etc, seguir o padrão de camelcase. Utilizar a biblioteca ESLint para padronizar estilo do código.

3.6 Restrições de Design

3.6.1 Padrão da API

Construir uma API HTTP no padrão RestFul, em NodeJS utilizando a biblioteca Express para as requisições HTTP. No frontend, utilizar React para a construção da interface.

3.6.2 Linguagem de programação

Utilizar o Typescript em ambiente de desenvolvimento com transpilação para o Javascript.

3.6.3 Padrão de código

Utilizar o padrão MVC para estruturação de classes, arquivos e pastas.

3.6.4 Padrão de deploy

Utilizar o Docker para a construção de um container para execução da API e da interface.

3.7 Documentação do Usuário On-line e Requisitos do Sistema de Ajuda

Deve haver um vídeo de apresentação do sistema explicando quais problemas o sistema resolve e quais suas principais funcionalidades. Não haverão outros sistemas de ajuda ao usuário.

3.8 Componentes Comprados

Não existem componentes comprados.

3.9 Interfaces

3.9.1 Interfaces com o Usuário

A API será um sistema web responsivo que pode ser acessado via qualquer navegador, portanto as interfaces com o usuário englobam dispositivos como computadores, notebooks, smartphones, etc.

3.9.2 Interfaces de Hardware

O sistema será desenvolvido em NodeJS e React, então qualquer hardware que suporte tais ferramentas deve suportar o sistema.

3.9.3 Interfaces de Software

Será utilizado NodeJs para o desenvolvimento do backend do sistema, e React para o frontend.

3.9.4 Interfaces de Comunicações

Não serão necessárias.

3.10 Requisitos de Licença

O sistema deve possuir uma licença de software livre.

3.11 Observações Legais, sobre Direitos Autorais e Outras Observações

Como é um sistema de software livre, não existem observações a serem adicionadas aqui.

3.12 Padrões Aplicáveis

Não se aplica.

4. Informações de Suporte

Não se aplica.

A.3 Histórias de usuário

Seguem todas as histórias de usuário descritas no começo do desenvolvimento do projeto, com suas descrições, cenários de teste e subtarefas.

A.3.1 Configuração inicial do projeto

Descrição:

```
COMO sistema
EU GOSTARIA DE ter um repositório público no Github
E as APIs de frontend e backend devidamente configuradas
PARA CONSEGUIR armazenar seguramente o código do sistema, realizando
controle de versão com o Git
E permitir o desenvolvimento das funcionalidades da plataforma nas APIs
```

Para esta história, não foram criados cenários de teste.

Subtarefas:

- Criação do repositório
- Configuração *backend*
- Configuração *frontend*

A.3.2 Cadastro de clientes

Descrição:

```
COMO uma pessoa comum
EU GOSTARIA DE poder me cadastrar no sistema AdoteFácil
PARA CONSEGUIR acessar as funcionalidades do sistema
```

Cenários de teste:

```
Cenário principal - Usuário se cadastra com informações válidas
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir suas informações no formulário de cadastro
    (nome, email e senha) e pressionar o botão "Cadastrar"
    ENTÃO a aplicação deve salvar as informações da pessoa no banco
    (com a senha criptografada) e redirecioná-la para a tela de login
```

```
Cenário alternativo - Usuário insere email inválido
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir um email inválido no formulário de cadastro
    ENTÃO a aplicação deve informar à pessoa que o email é inválido e
    impedir que o botão "Cadastrar" seja clicado.

Cenário alternativo - Usuário insere senhas diferentes no input de senha
e confirmação de senha
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir senhas diferentes no input de senha e
    de confirmação de senha no formulário de cadastro
    ENTÃO a aplicação deve informar à pessoa que as senhas estão
    divergentes e impedir que o botão "Cadastrar" seja clicado.

Cenário alternativo - Usuário insere email já existente
    DADO que uma pessoa comum deseja se cadastrar como usuário no sistema
    QUANDO a pessoa inserir suas informações no formulário de cadastro,
    pressionar o botão "Cadastrar" e o email inserido já estiver
    cadastrado no sistema
    ENTÃO a aplicação deve informar um erro ao cliente com a mensagem
    "Email já cadastrado no sistema" e se manter na tela de cadastro
```

Subtarefas:

- Protótipo do cadastro *mobile*
- Protótipo do cadastro *web*
- Tela de cadastro
- Rota de cadastro no *backend*
- Integração da tela de cadastro com *backend*

A.3.3 Login de clientes

Descrição:

```
COMO um usuário do sistema AdoteFácil
EU GOSTARIA DE poder logar na plataforma
PARA CONSEGUIR acessar as funcionalidades do sistema
```

Cenários de teste:

```
Cenário principal - Usuário insere informações válidas no login
    DADO que um usuário deseja fazer login na plataforma
    QUANDO o usuário inserir suas informações corretas de email e senha
    no formulário de login e pressionar o botão "Login"
    ENTÃO a aplicação deve redirecionar o usuário para a tela principal

Cenário alternativo - Usuário insere email inválido
    DADO que um usuário deseja fazer login na plataforma
    QUANDO o usuário inserir um email inválido no formulário de login
    ENTÃO a aplicação deve informar ao usuário que o email é inválido
    e impedir que o botão "Login" seja clicado.

Cenário alternativo - Usuário insere email não cadastrado
    DADO que um usuário deseja fazer login na plataforma
    QUANDO o usuário preencher o formulário de login com um email não
    cadastrado no sistema e pressionar o botão "Login"
    ENTÃO a aplicação deve informar um erro ao cliente com a mensagem
    "Credenciais inválidas" e se manter na tela de login

Cenário alternativo - Usuário insere senha incorreta
    DADO que um usuário deseja fazer login na plataforma
    QUANDO o usuário preencher o formulário de login com um email
    cadastrado mas com a senha diferente da senha salva e pressionar
    o botão "Login"
    ENTÃO a aplicação deve informar um erro ao cliente com a mensagem
    "Credenciais inválidas" e se manter na tela de login
```

Subtarefas:

- Protótipo do login *mobile*
- Protótipo do login *web*
- Tela de login
- Rota de login no *backend*
- Integração tela de login com *backend*

A.3.4 Atualização de dados do cliente

Descrição:

```
COMO um usuário do sistema AdoteFácil
EU GOSTARIA DE poder atualizar minhas informações na plataforma
PARA CONSEGUIR manter meus dados mais atualizados
```

Cenários de teste:

```
Cenário principal - Usuário atualiza seus dados de login com dados válidos
    DADO que um usuário deseja atualizar seus dados de login
    QUANDO o usuário acessar a tela de atualização de dados, inserir
    os dados a serem atualizados no formulário (email, senha e nome
    podem ser atualizados) e pressionar o botão "Atualizar"
    ENTÃO a aplicação deve salvar os dados atualizados no banco
    (com a senha criptografada), informar visualmente que os dados
    foram atualizados e se manter na mesma tela.

Cenário alternativo - Usuário insere email inválido
    DADO que um usuário deseja atualizar seus dados de login
    QUANDO o usuário inserir um email inválido no formulário
    ENTÃO a aplicação deve informar à pessoa que o email é inválido e
    impedir que o botão "Atualizar" seja clicado.

Cenário alternativo - Usuário insere senhas diferentes no input de senha
e confirmação de senha
    DADO que um usuário deseja atualizar seus dados de login
    QUANDO o usuário inserir senhas diferentes no input de senha e de
    confirmação de senha no formulário
    ENTÃO a aplicação deve informar à pessoa que as senhas estão
    divergentes e impedir que o botão "Atualizar" seja clicado.

Cenário alternativo - Usuário insere email já existente
    DADO que um usuário deseja atualizar seus dados de login
    QUANDO o usuário inserir suas informações no formulário, pressionar
    o botão "Atualizar" e o email inserido já estiver cadastrado no sistema
    ENTÃO a aplicação deve informar um erro ao cliente com a mensagem
    "Email já cadastrado no sistema" e se manter na tela de atualização
    de dados
```

Subtarefas:

- Protótipo da edição de dados cadastrais *mobile*
- Protótipo da edição de dados cadastrais *web*
- Rota de atualização de dados no *backend*
- Tela de atualização de dados do cliente
- Integração tela de atualização de dados com *backend*

A.3.5 Disponibilização de animal para adoção

Descrição:

```
COMO um usuário do sistema AdoteFácil
EU GOSTARIA DE poder disponibilizar um animal para adoção, com suas
informações e fotos
PARA QUE outra pessoa possa adotá-lo
```

Cenários de teste:

```
Cenário principal - Usuário preenche informações do animal
    DADO que um usuário deseja disponibilizar um animal para adoção
    QUANDO o usuário acessar a página de cadastro de animais para adoção,
    preencher as informações do animal corretamente (nome, tipo, gênero,
    raça, descrição e fotos do animal) e pressionar o botão "Cadastrar"
    ENTÃO a aplicação deve salvar os dados do animal no banco, informar
    ao usuário que o animal foi cadastrado e transitar para a tela
    de "Animais cadastrados"
```

Subtarefas:

- Protótipo da disponibilização de animal para adoção *web*
- Protótipo da disponibilização de animal para adoção *mobile*
- Rota de cadastro de animal para adoção no *backend*
- Integração do *backend* com o *frontend*
- Telas de disponibilização de animal para adoção

A.3.6 Listagem de animais disponíveis para adoção

Descrição:

```
COMO um usuário do sistema AdoteFácil
EU GOSTARIA DE poder visualizar os animais disponíveis para adoção na
plataforma e filtrá-los
PARA QUE eu possa encontrar o animal ideal para adotar
```

Cenários de teste:

```
Cenário principal - Usuário acessa página de "Animais para adoção"
    DADO que um usuário deseja adotar um animal pela plataforma
    QUANDO o usuário acessar a página de "Animais para adoção"
    ENTÃO a aplicação deve exibir os animais disponíveis, com opções de
    filtro (nome, tipo, gênero, e raça) para o usuário

Cenário alternativo - Usuário deseja filtrar animais disponíveis
    DADO que um usuário deseja adotar um animal pela plataforma
    QUANDO o usuário acessar a página de "Animais para adoção" e
    selecionar algum filtro de exibição
    ENTÃO a aplicação deve exibir os animais disponíveis com os filtros
    selecionados aplicados à listagem
```

Subtarefas:

- Protótipo da listagem de animais disponíveis *web*
- Protótipo da listagem de animais disponíveis *mobile*
- Rota no *backend*
- Tela de listagem de animais
- Integração tela de listagem de animais com rota no *backend*

A.3.7 Confirmação de adoção

Descrição:

```
COMO um usuário do sistema AdoteFácil que possui animais disponíveis
para adoção
```

```
EU GOSTARIA DE confirmar a adoção de um animal
PARA impedir que um animal que já foi adotado seja exibido como
disponível para adoção na plataforma
```

Cenários de teste:

```
Cenário principal - Usuário atualiza um animal disponível para adoção
para doado
    DADO que um usuário com um animal disponível para adoção chegou em
    um acordo com um adotante, doou o animal e agora deseja atualizar
    o animal na plataforma para que não seja exibido como disponível
    para adoção
    QUANDO o usuário clicar no botão de "Confirmar adoção"
    ENTÃO a aplicação deve atualizar os dados do animal no banco e não
    exibí-lo mais na listagem de animais disponíveis para adoção para
    os usuários da plataforma
```

Subtarefas:

- Protótipo da confirmação de adoção de animal *web*
- Protótipo da confirmação de adoção de animal *mobile*
- Rota de confirmar adoção de animal no *backend*
- Confirmação de adoção de animal no *frontend*

A.3.8 *Chat* entre adotante e doador

Descrição:

```
COMO um usuário do sistema AdoteFácil
EU GOSTARIA DE ter um chat de mensagens com o doador de um animal que
desejo adotar
PARA QUE possamos ter um contato mais próximo e fácil e combinar os termos
da adoção
```

Cenários de teste:

```
Cenário principal - Usuário que deseja adotar um animal acessa chat
com o adotante
```

```
DADO que um usuário deseja adotar um animal pela plataforma
QUANDO o usuário encontrar um animal que deseja, acessar a página do
animal e clicar no botão "Entrar em contato com o doador"
ENTÃO a aplicação deve transitar para uma página de chat do usuário
adotante com o doador do animal
```

Subtarefas:

- Protótipo do chat entre adotante e doador *web*
- Protótipo do chat entre adotante e doador *mobile*
- Rota de inserir mensagem de um usuário para outro no *backend*
- Rota de listar mensagens entre dois usuários no *backend*
- Implementação do chat no *frontend*

A.3.9 Empacotamento do sistema

Descrição:

```
COMO sistema
EU GOSTARIA DE ter uma configuração de conteinerização
PARA CONSEGUIR ser implantado em dispositivos diferentes independente
do ecossistema
```

Para esta história, não foram criados cenários de teste.

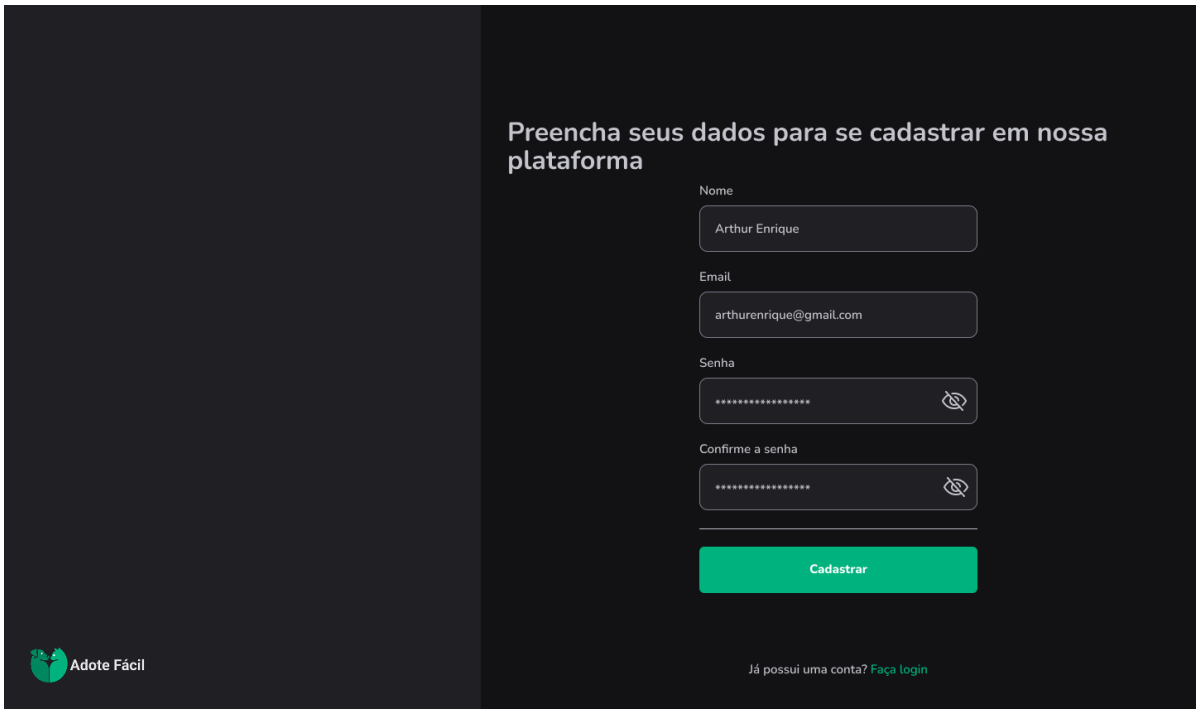
Subtarefas:

- Container do *backend*
- Container do *frontend*
- Orquestrar contêineres junto com o contêiner do banco de dados

A.4 Protótipos

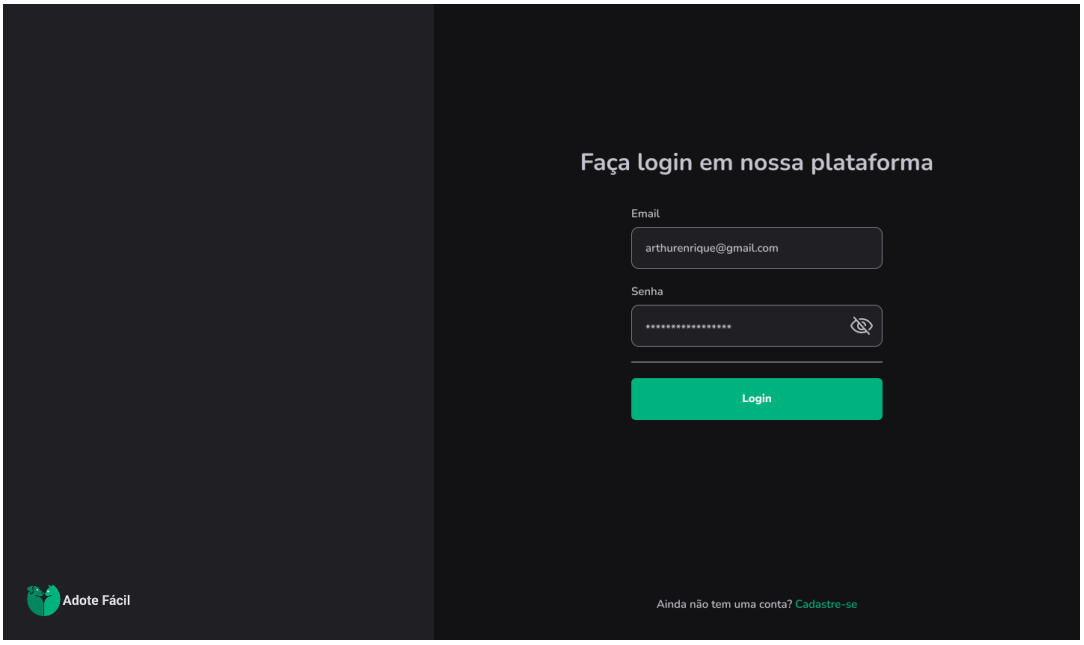
A seguir estão as imagens dos demais protótipos desenvolvidos no projeto.

Figura 41 – Protótipo da tela de cadastro



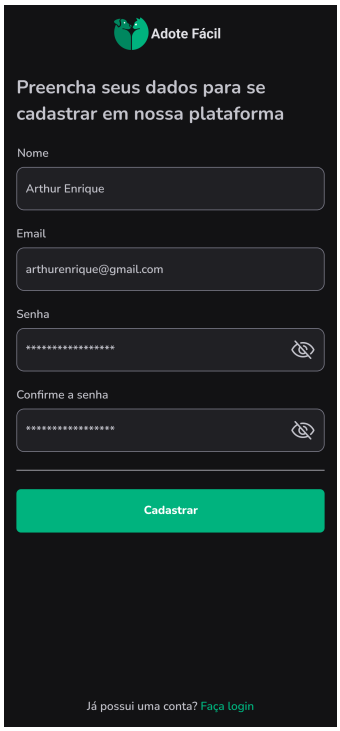
Fonte: Elaborado pelo autor

Figura 42 – Protótipo da tela de *login*



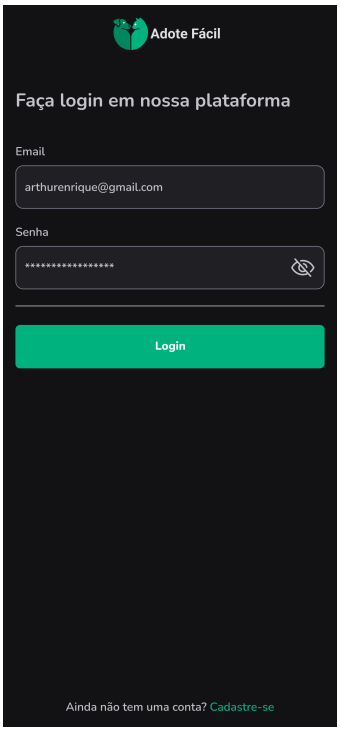
Fonte: Elaborado pelo autor

Figura 43 – Protótipo do cadastro *mobile*



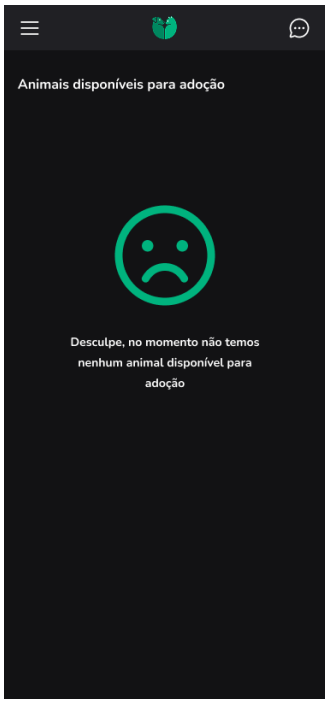
Fonte: Elaborado pelo autor

Figura 44 – Protótipo do *login mobile*



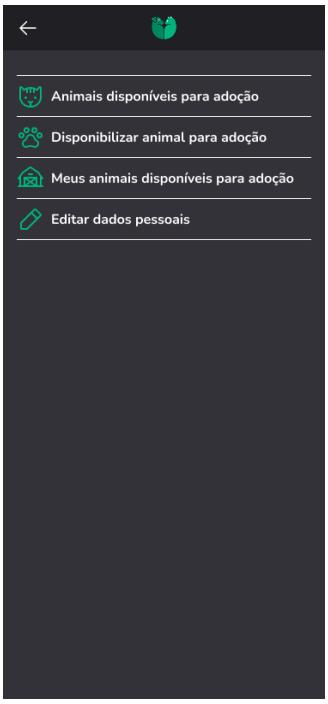
Fonte: Elaborado pelo autor

Figura 45 – Protótipo da tela principal sem animais



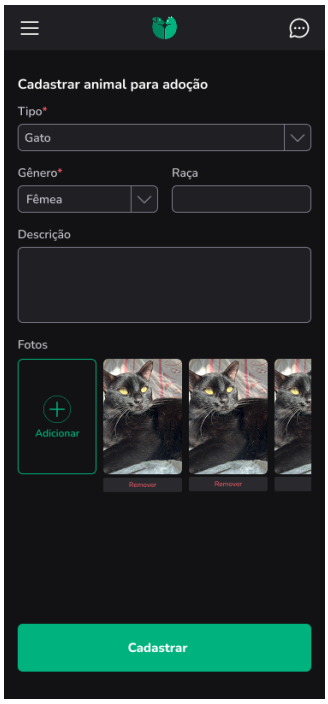
Fonte: Elaborado pelo autor

Figura 46 – Protótipo do menu lateral



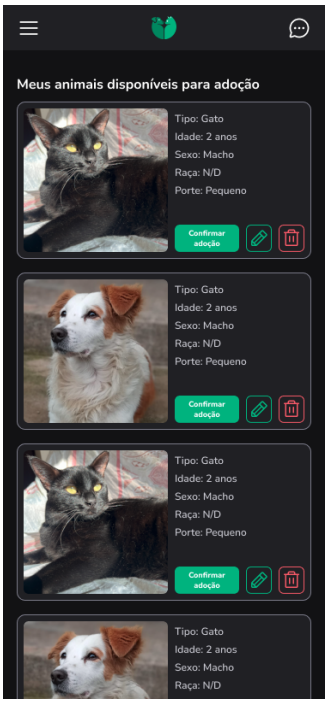
Fonte: Elaborado pelo autor

Figura 47 – Protótipo da tela de disponibilizar animal



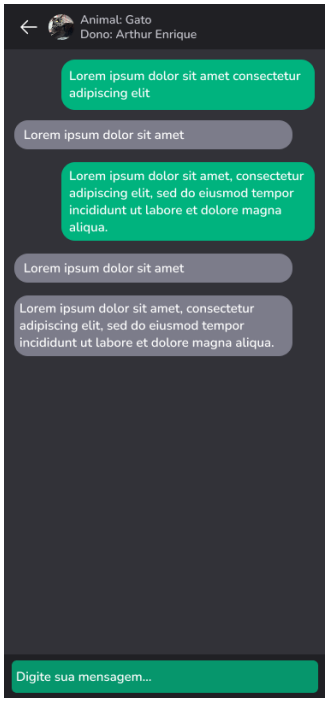
Fonte: Elaborado pelo autor

Figura 48 – Protótipo da tela de meus animais



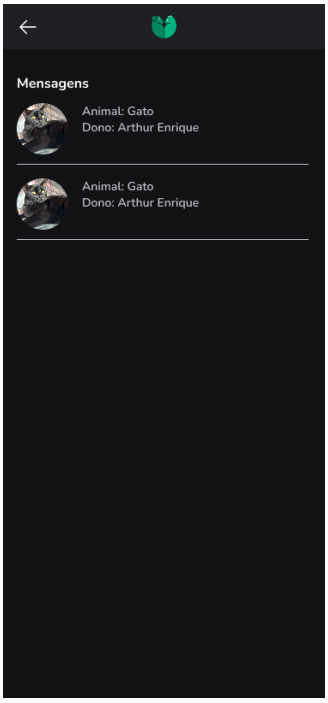
Fonte: Elaborado pelo autor

Figura 49 – Protótipo da tela de *chat*



Fonte: Elaborado pelo autor

Figura 50 – Protótipo da tela de conversas



Fonte: Elaborado pelo autor