UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE RUSSAS

CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

EMANUEL MAXIMILIANO PIRILO LIMA SOUSA

ADOTAÍ: UM APLICATIVO PARA ADOÇÃO DE ANIMAIS COM UTILIZAÇÃO DO ATOMIC DESIGN

RUSSAS

2023

EMANUEL MAXIMILIANO PIRILO LIMA SOUSA

ADOTAÍ: UM APLICATIVO PARA ADOÇÃO DE ANIMAIS COM UTILIZAÇÃO DO ATOMIC DESIGN

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Orientadora: Profª. Drª. Patrícia Freitas Campos de Vasconcelos

RUSSAS

2023

Dados Internacionais de Catalogação na Publicação

Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S696a Sousa, Emanuel Maximiliano Pirilo Lima.
Adotaí: Um aplicativo para adoção de animais com utilização do Atomic Design / Emanuel MaximilianoPirilo Lima Sousa. – 2023.
93 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas,Curso de Engenharia de Software, Russas, 2023.
Orientação: Profa. Dra. Patrícia Freitas Campos de Vasconcelos.

1. Abandono de animais. 2. Atomic Design. 3. App. 4. User Interface. I. Título.

CDD 005.1

EMANUEL MAXIMILIANO PIRILO LIMA SOUSA

ADOTAÍ: UM APLICATIVO PARA ADOÇÃO DE ANIMAIS COM UTILIZAÇÃO DO ATOMIC DESIGN

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Aprovada em: xx/xx/xxxx.

BANCA EXAMINADORA

_____

Profª. Drª. Patrícia Freitas Campos de Vasconcelos (Orientador)
Universidade Federal do Ceará (UFC)

_____

Profª. Drª. Anna Beatriz dos Santos Marques
Universidade Federal do Ceará (UFC)

_____

Prof. Me. Pitágoras Graça Martins
Universidade Federal do Ceará (UFC)

Dedico este trabalho a Deus, a meus pais, Maria Wanderléia de Lima, Luis Lima Sousa e a minha avó, Maria Ilda de Lima.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me proporcionar a capacidade para realizar este trabalho, por me dá força e perseverança e não me deixar desistir na busca dos meus objetivos.

Aos meus pais, Maria Wanderléia de Lima e Luis Lima Sousa pela educação que me foi passada ao longo da vida e por não desacreditarem dos meus sonhos.

A minha avó, Maria Ilda de Lima, por me proporcionar todo o apoio financeiro necessário durante a graduação, e desse modo, possibilitar a minha continuidade no curso.

Aos meus amigos, Ismael Sousa e João Denilson, pelos anos morados juntos durante a graduação e por todo o apoio e ajuda.

A minha orientadora, Dra. Patrícia Freitas Campos de Vasconcelos, por aceitar conduzir meu trabalho, me guiar no rumo da pesquisa e sanar todas as minhas dúvidas.

A todos os professores da Universidade Federal do Ceará – Campus Russas, por todo o ensino de qualidade e conhecimento repassado, tornando possível a minha formação.

“Se Deus permite sonhar, ele permite realizar.”
(Matheus Saloto)

RESUMO

O número de animais de estimação no Brasil tem crescido significativamente, e com isso, também cresce o número de animais abandonados. Diversos fatores explicam os motivos que levam as pessoas a praticarem o abandono, como: a falta de planejamento, mudança de residência e fatores econômicos. Tal prática, ocasiona em problemas ainda maiores tanto para a população como para os animais nas ruas, pois eles estão sujeitos a maus-tratos, acidentes de trânsito fatais e transmissões de doenças. Esta pesquisa objetiva a criação de um *app* para adoção de animais, aproximando animais que precisam ser adotados de futuros tutores. Para isso, objetivou-se a criação da interface do *app* a partir da utilização do *Atomic Design*, como metodologia para criação de interfaces de usuário de maneira padronizada, hierárquica e de fácil reutilização. Além da utilização da engenharia de *software* orientada a reuso, como processo para guiar as atividades da metodologia. Dessa forma, foram identificadas e documentadas as necessidades a partir do estudo de outros trabalhos, prototipado os componentes e as telas do *app*. Além disso, a solução projetada foi validada a partir de um questionário disponibilizado aos usuários junto dos protótipos, com o objetivo de obter *feedbacks* da solução apresentada e verificar se os requisitos coletados foram atendidos. Foi ainda realizado o desenvolvimento do aplicativo utilizando o framework React Native e aplicado um teste de usabilidade para verificar o nível de satisfação dos usuários com o *app*. Através deste trabalho, realizou-se pesquisa bibliográfica, *benchmarking* de soluções desenvolvidas, documentação de requisitos, prototipação, aplicação de questionário ao público-alvo, desenvolvimento do *app* e teste de usabilidade.

Palavras-chave: abandono de animais; atomic design; app; user interface;

ABSTRACT

The number of pets in Brazil has grown significantly, and with that, the number of abandoned animals has also grown. Several factors explain the reasons that lead people to practice abandonment, such as: lack of planning, change of residence and economic factors. This practice causes even greater problems for both the population and the animals on the streets, as they are subject to mistreatment, fatal traffic accidents and disease transmission. This research aims to create an app for animal adoption, bringing animals that need to be adopted closer to future tutors. For this, the objective was to create the app's interface using Atomic Design, as a methodology for creating user interfaces in a standardized, hierarchical and easy-to-reuse way. In addition to the use of reuse-oriented software engineering, as a process to guide the methodology's activities. In this way, the needs were identified and documented based on the study of other works, prototyping the app's components and screens. In addition, the designed solution was validated based on a questionnaire made available to users along with the prototypes, with the aim of obtaining feedback on the presented solution and verifying whether the collected requirements were met. The application was also developed using the React Native framework and a usability test was applied to verify the level of user satisfaction with the app. Through this work, bibliographical research, benchmarking of developed solutions, requirements documentation, prototyping, application of a questionnaire to the target audience, app development and usability testing were carried out.

Keywords: animal abandonment; atomic design; app; user interface;

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ABINPET	Associação Brasileira da Indústria de Produtos para Animais de Estimação
ACV	Animais em Condição de Vulnerabilidade
API	Application Programming Interface
CSS	Cascading Style Sheet
DOM	Document Object Model
ER	Entidade Relacionamento
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IBGE	Instituto Brasileiro de Geografia e Estatística
iOS	iPhone Operating System
JSON	JavaScript Object Notation
IPB	Instituto Pet Brasil
OHA	Open Handset Alliance
ONG	Organização não governamental
REST	Representational State Transfer
RF	Requisito funcional
RN	Regras de negócio
RNF	Requisito não-funcional
UFC	Universidade Federal do Ceará
UI	User Interface
UML	Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

O abandono de cães e gatos é uma prática que vem crescendo a cada ano, isso faz com que se tenha um número de animais cada vez maior nas ruas das cidades brasileiras. Animais em Condição de Vulnerabilidade (ACVs) são animais que vivem sob tutela das famílias classificadas abaixo da linha da pobreza, ou que vivem nas ruas e dependem de cuidados de pessoas ao redor. Entre os anos de 2018 a 2020, o número de ACVs mais que dobrou, subindo de 3,9 milhões para 8,8 milhões, sem incluir os animais que são resgatados por maus-tratos e abandonados. A maioria desses animais abandonados e resgatados por maus-tratos vivem sob tutela de organizações não governamentais (Instituto Pet Brasil, 2022).

O número de animais de estimação também está em crescimento, segundo dados da ABINPET (2022), a população de animais de estimação é cerca de 149,6 milhões, entre cães, gatos, peixes, aves e répteis e pequenos mamíferos. A maioria é de cachorros (58,1 milhões), seguido de felinos (27,1 milhões), peixes (20,8 milhões), aves (41 milhões) e outros (2,53 milhões). Com o aumento dos animais de estimação, também cresce o número de animais abandonados. Essa realidade faz com que se tenha um elevado número de cães e gatos nas ruas das cidades brasileiras, tornando-se um longo desafio à saúde pública, já que estes animais podem causar agressões, poluição ambiental, transmissão de zoonoses e acidentes de trânsito fatais (SCHERER, 2021).

Uma forma de reduzir o número de animais nas ruas é através da prática de adoção, utilizando os meios digitais. Atualmente, a forma mais utilizada para divulgação dos animais são as redes sociais, que possuem grande alcance de pessoas e ajudam a promover a conscientização da população na luta contra o abandono. Checoni (2020) afirma que as redes sociais possuem papel importante no processo de adoção, pois auxiliam as famílias que desejam adotar e não sabem onde procurar. Entretanto, apesar de serem bastante importantes, as redes sociais não possuem tanta eficácia, pois não foram desenvolvidas com essa proposta. Assim, a utilização de uma ferramenta exclusiva para adoção de animais pode ser considerada como solução prática e eficiente, para aproximar o animal com seu futuro tutor.

Tais soluções como forma de aproximar animais a tutores são objetos de estudo de diversos trabalhos, como a pesquisa de Guimarães (2021), que desenvolveu um aplicativo para simplificar o processo de adoção de animais, e como resultado, foram identificados alguns

problemas quanto ao *design* da aplicação, a indisponibilidade para dispositivos iOS e a necessidade de funcionalidades como cadastro de usuário e edição de pets. Já na pesquisa de Rocha e Junior (2021), que também desenvolveu um aplicativo para doação de animais, foram encontradas algumas necessidades como, a falta de mais informações sobre os pets, além de funcionalidades como busca por espécie e geolocalização. Nesse cenário, torna-se evidente que as soluções atuais podem ser melhoradas, se atendidas as necessidades, proporcionando satisfação e benefícios para os usuários.

Desse modo, será aplicado o *Atomic Design* como metodologia para criação de componentes de um aplicativo para adoção de animais, possibilitando mais facilidade na interação entre pets e pessoas que desejam doar ou adotar, contribuindo assim com a redução do número de animais nas ruas e consequentemente, a proliferação de doenças e maus-tratos. A utilização do *Atomic Design* no desenvolvimento de sistemas é essencial, pois introduz uma forma modularizada e independente de criação do produto, o que proporciona uma rápida adaptação e evolução do sistema separada dos seus componentes (SANTIAGO, 2022).

2 OBJETIVOS

2.1 Objetivo geral

Desenvolver um aplicativo para adoção de animais com a utilização do *Atomic Design* como metodologia que auxilia na criação de componentes de interface reutilizáveis.

2.2 Objetivos específicos

- Utilizar os fundamentos do *Atomic Design* na criação de componentes reutilizáveis;
- Utilizar o *React Native* como *framework* no desenvolvimento do aplicativo;
- Validar os protótipos desenvolvidos com os usuários utilizando a técnica de questionário;
- Realizar um teste de usabilidade para verificar a sastisfação dos usuários com a utilização do *app*;

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são descritas as teorias que serão abordadas durante a realização desta pesquisa.

3.2 Abandono de animais

O número de animais abandonados no Brasil vem crescendo a cada ano. De acordo com o Instituto Pet Brasil (2022) num primeiro levantamento realizado em 2018, o número de animais em condições de vulnerabilidade chegou a marca de 3,9 milhões no país. Já em 2020, após um novo levantamento, esse número saltou para 8,8 milhões, o que representa um crescimento de 126%. Animais em Condições de Vulnerabilidade (ACVs) são aqueles que vivem sob tutela de famílias classificadas abaixo da linha da pobreza, ou que vivem nas ruas e dependem dos cuidados de pessoas ao redor. Desses animais, cães representam 69,4% (6,1 milhões), enquanto os gatos correspondem a 30,6% (2,7 milhões). Segundo esses números, 2,1% dos ACVs evoluem para o abandono completo.

As principais causas que contribuem para o abandono dos animais de acordo com Prado et al., (2018), são: a falta de planejamento para ter um animal de estimação, questões financeiras, disponibilidade de tempo e o cuidado envolvido. Segundo Ouriques (2018), os principais motivos de abandonos são: ninhadas esperadas, mudança de casa e fatores econômicos. Já para Santana (2006), a falta de planejamento acarreta nas pessoas a compra de animais simplesmente pelo impulso de consumir, onde muitas vezes o consumo não desperta o vínculo afetivo entre o animal e o homem, fazendo com que as pessoas acabem descartando seus animais de estimação por ficarem desinteressantes após a empolgação inicial.

De acordo com Ouriques (2018), os animais em situação de abandono hoje no Brasil estão sendo considerados problema de saúde pública, onde, cães e gatos abandonados ao relento, em péssimas condições, muitas vezes doentes, transmitem doenças aos humanos. Scherer (2021), reforça esta ideia afirmando que a prática de abandono de animais faz com que se tenha um número elevado de cães e gatos nas ruas das grandes e pequenas cidade brasileiras, tornando-se um longo desafio à saúde pública, já que estes animais podem causar agressões, poluição ambiental, transmissão de zoonoses e acidentes de trânsito fatais.

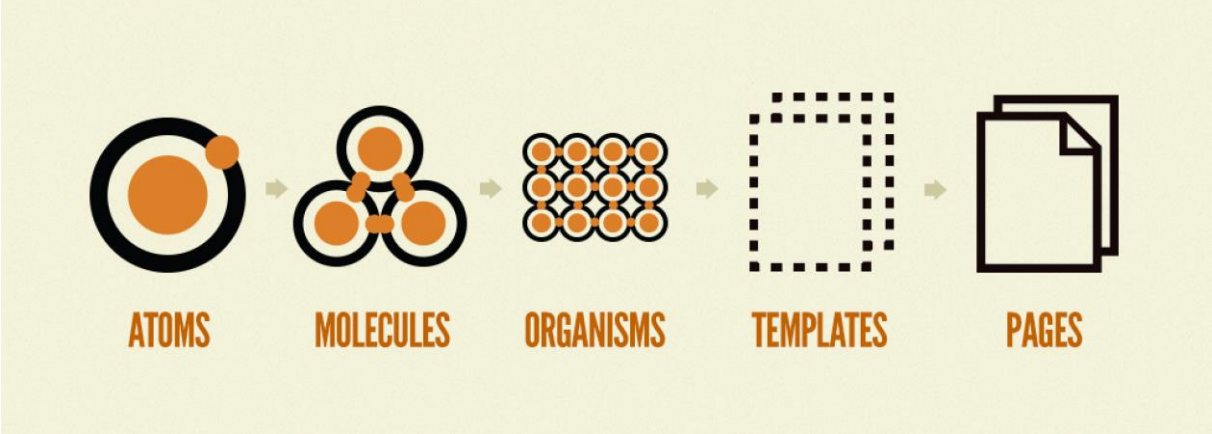
3.3 Atomic Design

O *Atomic Design* é uma metodologia desenvolvida pelo *web designer* Brad Frost em 2013, para criação de componentes de interfaces de maneira hierárquica. Segundo Frost (2016), o *Atomic Design* não é um processo linear, mas sim um modelo mental para nos ajudar a pensar em nossas interfaces de usuário como um todo coeso, e uma coleção de partes ao mesmo tempo.

Esta metodologia é oriunda da química, onde átomos se juntam com outros átomos para criar elementos mais complexos. No mundo natural, os elementos atômicos se combinam para formar moléculas, essas moléculas podem se combinar ainda mais para formar organismos relativamente complexos (FROST, 2016).

Para Bonfitto (2019), a ideia central dessa metodologia é que as interfaces de usuário sejam divididas nas menores partes possíveis, chamadas de átomos, possibilitando um sistema de possibilidades e combinações. A metodologia do *Atomic Design* é composta por 5 etapas: Átomos, Moléculas, Organismos, Templates e Páginas (ver Figura 1).

Figura 1- Etapas do Atomic Design



Fonte: Brad Frost (2016)

As etapas ilustradas na (Figura 1) referem-se aos cinco estágios do *Atomic Design*, iniciando pelos Átomos, que são os blocos de construção que compõem as interfaces, seguido das Moléculas, que são elementos formados a partir de dois ou mais átomos, posteriormente com os Organismos, que são componentes formados por grupos de moléculas e/ou átomos, depois os Templates, que organizam a estrutura de um *layout* utilizando os componentes, e finalmente as páginas, formadas pela junção das etapas anteriores.

3.4 Desenvolvimento de aplicativos móveis

O desenvolvimento de aplicativos móveis tem se tornado uma prática bastante recorrente entre as empresas. Isso ocorre principalmente pela quantidade de acesso que as pessoas têm aos dispositivos nos dias atuais, o que torna um fácil acesso dos produtos ao consumidor pelas empresas. A popularidade dos dispositivos móveis fortaleceu ainda mais essa tendência, proporcionando novas formas de interagir e fechar negócios com o público (OBJECTIVE, 2020).

Segundo pesquisas da *StatCounter Global Stats* (2021), os dois sistemas operacionais para dispositivos móveis mais utilizados no mundo todo são o *Android* e o *iOS.*

Em 2003, Rich Miner, Nick Sears, Chris White e Andy Rubin criaram o *Android* e uma companhia chamada *Android Inc.* Inicialmente, o objetivo era desenvolver um sistema operacional para câmeras fotográficas, entretanto, como não era um mercado promissor decidiu-se investir em celulares (MYMOB, 2020). Posteriormente, o *Android* foi comprado pela *Google* e desenvolvido pela *Open Handset Alliance* (OHA). *Open Handset Alliance* é um grupo de 84 empresas de tecnologia e telefonia móvel que se uniram para acelerar a inovação em dispositivos móveis e oferecer aos consumidores uma experiência móvel mais rica, barata e melhor (OHA, 2021).

Em 2007 foi lançado o iOS, juntamente com o lançamento do primeiro iPhone sob o nome de iPhone OS. Após isso, novas versões do sistema operacional foram disponibilizadas juntamente com o lançamento anual e modelos do iPhone. A partir da versão 4, o sistema deixou de se chamar iPhone OS, sendo abreviado para iOS (JORDÃO, 2021). iOS é um sistema operacional desenvolvido pela Apple, que é executado exclusivamente nos dispositivos móveis da empresa. iOS é uma abreviatura para *iPhone Operation System*, o sistema foi baseado no Sistema Operacional MAC OS X e projetado para atender as necessidades de aparelhos móveis desenvolvidos pela Apple (MENDONÇA, 2011).

Segundo El-Kassas (2017), existem duas principais abordagens no desenvolvimento de aplicações móveis, a nativa e a híbrida.

De acordo com Bidiu (2016), as aplicações nativas são desenvolvidas especificamente para uma plataforma e podem aproveitar ao máximo todos os recursos do dispositivo, elas podem usar a câmera, o GPS, a lista de contato entre outros. Elas também podem incorporar gestos, utilizar o sistema de notificação do dispositivo e funcionar offline.

Já as aplicações híbridas são desenvolvidas utilizando tecnologias web incluindo, HTML, CSS e JavaScript. Para Pinheiro (2020), isso é possível pois as ferramentas que

implementam esse método criam um projeto nativo, instanciam um browser por meio de um componente *WebView* e executam o código da aplicação no *browser*.

Ao contrário das aplicações nativas, os aplicativos híbridos não possuem acesso direto às funcionalidades do dispositivo como, câmera, GPS, lista de contatos, sendo necessário a utilização de algum *framework*[1] que possibilite acesso a esses recursos (MENDES; GARBAZZA; TERRA, 2014).

Apesar da necessidade de um *framework* para ter acesso aos recursos do dispositivo, as aplicações híbridas possuem suas vantagens. Bidiu (2016), sustenta a ideia de que as aplicações híbridas permitem o desenvolvimento multiplataforma, e desse modo, reduzem os custos de desenvolvimento já que o mesmo código pode ser reutilizado em diferentes sistemas operacionais.

[1] Para Johnson e Foote (1998), framework é um conjunto de classes que constitui um projeto abstrato para a solução de uma família de problemas.

4 TRABALHOS RELACIONADOS

Neste capítulo, são apresentados trabalhos que contribuíram para a atual pesquisa quanto ao desenvolvimento de aplicações para adoção de animais, metodologia para prototipação de componentes e utilização do *React Native* como *framework* de desenvolvimento. A busca desses trabalhos foi realizada no Google Acadêmico, que é um repositório público que contém uma variedade de textos da literatura acadêmica.

O trabalho de Guimarães (2021) consiste no desenvolvimento de um aplicativo para simplificar o processo de adoção de animais por parte dos tutores e das ONGs na cidade de Campina Grande-PB. Este trabalho se assemelha na atual pesquisa pelo fato de ser desenvolvida uma aplicação para adoção de animais utilizando o *framework React Native*. A principal diferença, é que Guimarães (2021) cria um protótipo único e exclusivo para a aplicação final, enquanto esta pesquisa utiliza os conceitos do *Atomic Design* para criar os componentes do protótipo de forma hierárquica e que possam ser facilmente reutilizados por toda a aplicação. O objetivo do trabalho foi aproximar das organizações as pessoas que desejam adotar, estimulando a prática de adoção e reduzindo assim o número de animais abandonados nas ruas. A metodologia utilizada consistiu de pesquisas em aplicativos de adoção de animais, com o intuito de encontrar as principais funcionalidades, e o levantamento das tecnologias que seriam utilizadas no aplicativo. Foi ainda utilizado o Scrum como processo iterativo e incremental no desenvolvimento ágil, onde foi dividido em duas sprints. Na primeira sprint, foi realizado o levantamento de requisitos com o público-alvo, e na segunda sprint, foi realizada a configuração do projeto, criação das telas, desenvolvimento e teste de usabilidade. Por fim, o aplicativo foi avaliado por 07 usuários de Campina Grande-PB (com idades entre 20 e 40 anos) para verificar o nível de satisfação dos avaliadores.

A pesquisa desenvolvida por Silva (2017) foi a criação de uma interface para um aplicativo de doação de animais, tendo como motivação uma situação vivenciada pelo autor, que foi a dificuldade de encontrar um lar para uma cadela perdida. Este trabalho se assemelha na atual pesquisa pelo fato de elaborar um aplicativo para doação de animais utilizando uma metodologia de *design*. Entretanto, a diferença entre as duas se dá pelo fato de Silva (2017) utilizar um processo que vai desde a concepção dos requisitos até a prototipação, enquanto esta pesquisa utiliza os fundamentos do *Atomic Design* para estruturar e categorizar os elementos da interface de uma aplicação. O objetivo do trabalho foi desenvolver a interface de uma plataforma centralizada que se dedique exclusivamente a adoção de animais com foco na

usabilidade e simplificação do processo de adoção pelos usuários. A metodologia utilizada foi a dos elementos de experiência de usuário, desenvolvida por Jesse James Garrett, que consiste em projetar experiência de usuário nas interfaces das aplicações utilizando as seguintes etapas: Estratégia, escopo, estrutura, esqueleto e superfície.

O trabalho desenvolvido por Paz (2019) consiste na criação de um *Design System,* que é um conjunto de padrões e práticas organizadas de forma coerente para a criação de elementos e componentes de interface. Este trabalho se assemelha na pesquisa atual pelo fato de se utilizar o *Atomic Design* para organização e categorização dos elementos de interface que servirão de apoio para os desenvolvedores criarem componentes padronizados. Entretanto, a principal diferença, é que Paz (2019) utiliza o *Atomic Design* para a criação de um *Design System* que poderá ser utilizado como apoio ao desenvolvimento da interface de qualquer outro aplicativo, enquanto esta pesquisa utiliza os fundamentos do *Atomic Design* na criação de componentes padronizados para a interface de um aplicativo para doação de animais. O objetivo do trabalho foi criar elementos e componentes padronizados que pudessem ser reutilizados pela equipe e pelos desenvolvedores em qualquer aplicativo. A metodologia utilizada foi o *Atomic Design*, que é uma metodologia que categoriza e padroniza os elementos e componentes da interface nas seguintes etapas: Átomos, moléculas, organismos, templates e páginas.

Rocha e Junior (2021) desenvolveram um aplicativo para doação de animais com o objetivo de ajudá-los a encontrarem novos lares e tutores. Esse trabalho se assemelha na atual pesquisa pelo fato do autor utilizar o *React Native* como *framework* para o desenvolvimento do aplicativo. Entretanto, a diferença é que Rocha e Junior (2021) não utilizaram uma metodologia para criação dos protótipos e dos componentes da interface, enquanto que esta pesquisa utiliza dos fundamentos do *Atomic Design* para criação categorizada dos elementos que compõem o protótipo e de componentes reutilizáveis pelo aplicativo. A motivação do trabalho foi facilitar e acelerar o processo de adoção, onde o autor relata que a principal forma de divulgação de animais são as redes sociais, que por sua vez, não possuem tanta eficácia já que não são uma plataforma específica para esse fim, e com isto, muitos animais acabam não sendo encontrados. A metodologia utilizada foi conduzida da seguinte forma: revisão da literatura, estudo e escolha das tecnologias, levantamento de requisitos, modelagem da aplicação, desenvolvimento e testes. Por fim, foi realizada uma avaliação por 21 participantes com o objetivo de avaliar se o aplicativo atende as necessidades das ONGs e dos usuários que desejam adotar.

No trabalho de Junior et al., (2019), foi desenvolvido um software para gestão de adoção de animais com o intuito de reduzir a superlotação das ONGs e o número de animais abandonados das ruas, diminuindo assim também, a proliferação de doenças e os maus-tratos.

Este trabalho se assemelha na atual pesquisa pela proposta de desenvolver um software para que os animais possam ser encontrados e adotados. Entretanto, a principal diferença é que Junior et al., (2019) desenvolveu uma aplicação para a web utilizando HTML, CSS e o framework Laravel, enquanto que esta pesquisa propõe o desenvolvimento de um aplicativo móvel utilizando o *framework React Native*. A metodologia utilizada foi o processo da engenharia de software que consistiu das seguintes etapas: levantamento de requisitos, elaboração de diagramas da UML, modelagem e prototipação, além da utilização do Kanban como metodologia ágil na organização das tarefas e dos prazos.

O Quadro 1 abaixo destaca os principais critérios utilizados para comparação dos trabalhos relacionados a este. Como se pode observar, dois trabalhos utilizaram uma metodologia para prototipação, dois utilizaram o *framework React Native*, dois são desenvolvidos para *Android* e *iOS* e apenas um possibilita a busca de animais através de geolocalização.

Quadro 1 - Comparação dos trabalhos relacionados com este trabalho

Trabalho	Utilizou metodologia para prototipação?	Utilizou o framework React Native?	Permite busca através de geolocalização ?	Android & iOS?	Utilizou o Atomic Design?
Guimarães (2021)		X	X	X	
Silva(2017)	X				
Paz(2019)	X				
Rocha e Junior(2021)		X		X	
Junior et al.,(2019)					
Este trabalho	X	X	X	X	X

Fonte: Elaborado pelo autor (2023)

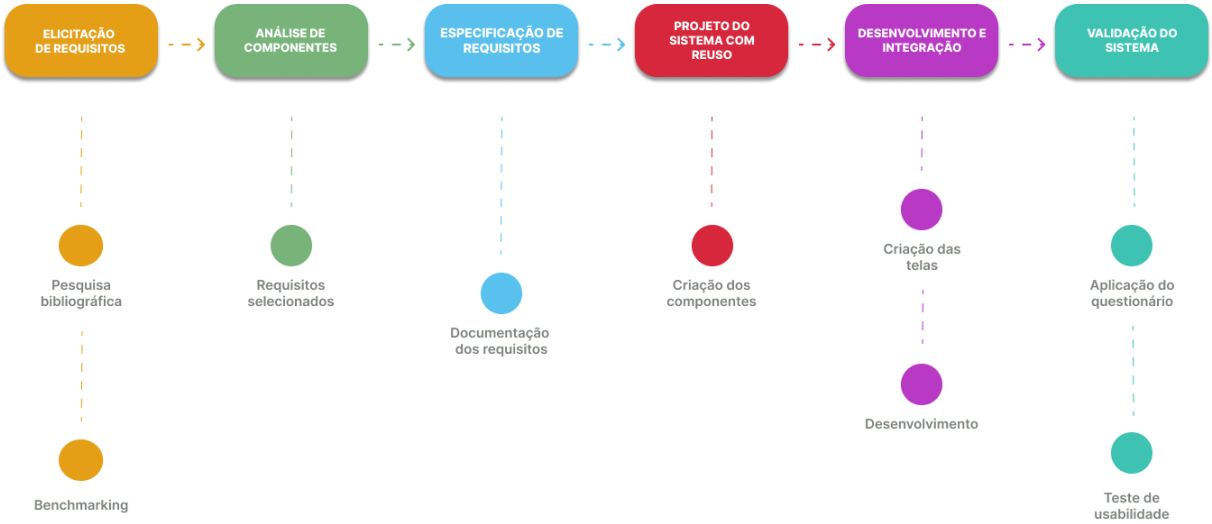
5 PROCEDIMENTOS METODOLÓGICOS

Neste capítulo, são descritas as atividades a serem desempenhadas para o alcance dos objetivos deste trabalho, com o intuito de obter as informações necessárias para a prototipação de componentes reutilizáveis e o desenvolvimento do *app* para adoção de animais. Para isso, foi utilizada a metodologia da engenharia de *software* orientada a reuso, que visa reduzir tempo e custos de desenvolvimento através da criação e utilização de componentes reutilizáveis, consistindo das seguintes etapas: Especificação de requisitos, Análise de componentes, Alteração nos requisitos, Projeto do sistema com reuso, Desenvolvimento e integração e Validação do sistema.

A primeira etapa da engenharia de software orientada a reuso, é a Especificação de requisitos. Esta é a etapa onde os requisitos são documentados, pois eles já foram elicitados. Desse modo, será necessário adaptar duas etapas desta metodologia de modo a se adequarem às necessidades desta pesquisa. A etapa de Especificação de requisitos dará lugar a Elicitação de requisitos, onde será realizado o entendimento das necessidades junto ao público-alvo e a coleta dos requisitos. E por último, a etapa de Alteração nos requisitos será substituída pela Especificação de requisitos, pois com os requisitos já identificados, eles poderão então ser documentados.

A figura abaixo, exemplifica as atividades que foram realizadas nos procedimentos metodológicos, e que foram associadas nesta pesquisa às etapas da engenharia de *software* orientada a reuso. A Pesquisa bibliográfica e o Benchmarking estão inseridos na etapa de Elicitação de requisitos, enquanto o Requisitos selecionados está associado a etapa de Análise de componentes, a Documentação dos requisitos por sua vez está inserida na etapa de Especificação de requisitos, a Criação de componentes faz parte da etapa de Projeto do sistema com reuso, a Criação das telas e o Desenvolvimento fazem parte da etapa de Desenvolvimento e integração, e finalmente, a Aplicação do questionário e o Teste de usabilidade estão inseridos na etapa de Validação do sistema (ver Figura 2).

Figura 2 - Etapas da metodologia



Fonte: Elaborado pelo autor (2023)

Embora a figura acima apresente as etapas da metodologia de forma sequencial, é possível regredir para uma etapa anterior a qualquer momento caso haja necessidade de alteração.

5.2 Elicitação de requisitos

Na etapa de elicitação dos requisitos, busca familiarizar-se com o público-alvo, com o objetivo de entender o contexto e identificar as suas principais necessidades. Para isso, a técnica utilizada nesta etapa foi a pesquisa bibliográfica e o benchmarking.

5.2.1 *Pesquisa bibliográfica*

Para a obtenção do entendimento das necessidades dos usuários, foram realizadas buscas no Repositório Institucional da UFC, onde foram feitas leituras de diversos trabalhos, além de buscas no *Google* Acadêmico, com o objetivo de analisar as pesquisas, a fim de entender as principais necessidades dos usuários e os principais problemas presentes nos trabalhos relacionados. Tais estudos são melhor detalhados no capítulo 4 trabalhos relacionados.

5.2.2 *Benchmarking*

O termo *benchmarking* significa ponto de referência, é uma técnica que analisa produtos, processos ou serviços de terceiros. O intuito é identificar práticas ou estratégias que

obtiveram bons resultados e segui-las, assim como erros que deverão ser evitados. Com base nessa técnica, foram analisados dois trabalhos relacionados, o de Guimarães (2021) e o de Rocha e Junior (2021), com o objetivo de identificar os requisitos, suas limitações e os principais problemas e acertos. Os resultados estão descritos na seção 6.2.

5.3 Análise de componentes

Nesta etapa, são realizadas análises nos projetos já existentes e selecionados os requisitos que serão considerados para o desenvolvimento.

5.3.1 *Requisitos selecionados*

Nesta etapa, foi realizado uma análise dos requisitos identificados na etapa do *benchmarking*, e dessa forma, foram selecionados os requisitos que seriam considerados no desenvolvimento do Adotaí. Os resultados estão descritos na seção 6.3.

5.4 Especificação dos requisitos

Nesta etapa, os requisitos são analisados e documentados com base nas informações obtidas na etapa anterior de análise de componentes. Ainda durante esta fase, os requisitos podem sofrer modificações de modo a se adequarem às necessidades dos usuários.

5.4.1 *Documentação dos requisitos*

Esta etapa contempla a especificação dos requisitos considerados no desenvolvimento do *app* Adotaí. De acordo com Turine e Masiero (1996), o documento de requisitos de software contém todos os requisitos funcionais e de qualidade do software, incluindo as características do produto, seus recursos disponíveis e os critérios de aceitação. Os resultados estão descritos na seção 6.4.

5.5 Projeto do sistema com reuso

A etapa de projeto do sistema com reuso consiste na prototipação de componentes padronizados e que possam ser reutilizáveis. Para isso, foi utilizado o *Atomic Design*, que é uma metodologia para criação e classificação dos componentes desenvolvida por Brad Frost, e que consiste de cinco estágios: Átomos, Moléculas, Organismos, Templates e Páginas.

5.5.1 *Criação dos componentes*

Nesta etapa, foram criados os componentes de interface que estarão presentes na aplicação, utilizando o *Atomic Design* como metodologia. Para isso, foi necessário selecionar os principais elementos visuais que compõem a aplicação dos trabalhos relacionados na etapa de *benchmarking,* como: botões, campo de texto, fontes e cores. A ferramenta utilizada para a prototipação dos componentes foi o *Figma*. Os resultados estão descritos na seção 6.5.

5.6 Desenvolvimento e integração

A etapa de desenvolvimento e integração consiste na integração de todos os componentes já criados até então. Esses componentes unidos, formam as telas que possuem as funcionalidades que atendem as necessidades dos usuários. Essa etapa é muito importante, pois atua como forma de validação dos requisitos. A metodologia utilizada na criação das telas foi o *Atomic Design*.

5.6.1 *Criação das telas*

Nesta etapa, foram prototipadas as telas da aplicação, utilizando como base os componentes criados na etapa anterior e o *Atomic Design* como metodologia. A ferramenta utilizada para criação das telas foi o *Figma*. Os resultados estão descritos na seção 6.6.

5.6.2 *Desenvolvimento*

Nesta etapa, foi desenvolvido o aplicativo a partir das telas prototipadas nas etapas de criação de componentes e criação das telas. O *framework* utilizado para o desenvolvimento do *app* foi o *React Native.* Os resultados estão descritos na seção 6.8.

5.7 Validação do sistema

A última etapa do processo de engenharia de *software* orientado a reuso, é a validação do sistema. Essa etapa, consiste em avaliar junto aos usuários a qualidade dos protótipos, verificando se o que foi projetado corresponde às necessidades dos usuários. A técnica utilizada nesta etapa foi a aplicação do questionário e o teste de usabilidade.

5.7.1 Aplicação do questionário

Nesta etapa, foram avaliados os protótipos desenvolvidos. Para isso, foi disponibilizado para os participantes os protótipos das telas desenvolvidas na etapa anterior, junto de um

questionário no *Google Forms,* com perguntas que possibilitaram verificar se os requisitos foram ou não atendidos. Os resultados estão descritos na seção 6.7.

5.7.2 Teste de usabilidade

Nesta etapa, após o desenvolvimento do aplicativo, foi realizado um teste de usabilidade junto aos usuários com o intuito de avaliar o grau de satisfação com o aplicativo e/ou potenciais problemas. Os resultados estão descritos na seção 6.9.

6 RESULTADOS

Neste capítulo, são apresentados os resultados obtidos a partir das etapas de Elicitação de requisitos, Análise de componentes, Documentação dos requisitos, Projeto do sistema com reuso, Desenvolvimento e integração e Validação do sistema.

6.2 Benchmarking

Nesta etapa, foram selecionados dois trabalhos relacionados com o intuito de extrair os requisitos que compõem uma aplicação de adoação de animais, e dessa forma, poder identificar suas limitações, principais problemas e acertos. Os trabalhos analisados foram os de Guimarães (2021) e o de Rocha e Junior (2021).

6.2.1 *Trabalho de Guimarães*

Guimarães (2021), desenvolveu um *app* chamado AdotaPet CG, para simplificar o processo de adoção de animais na cidade de Campina Grande-PB. O objetivo foi aproximar as ONGs das pessoas que desejam adotar, estimulando a prática de adoção e reduzindo o número de animais abandonados nas ruas.

Como limitação, foi identificado que somente usuários do tipo ONG poderiam cadastrar um animal para adoção, sendo necessário os tutores levarem os pets até uma ONG para que eles possam ser cadastrados, ocasionado possivelmente supoerlotações. Foi identificado ainda como limitação, o fato de não poder buscar as ONGs por determinada região e visualizá-las no mapa, dessa forma, o usuário precisa procurar por todo o mapa a cidade que deseja visualizar tais ONGs.

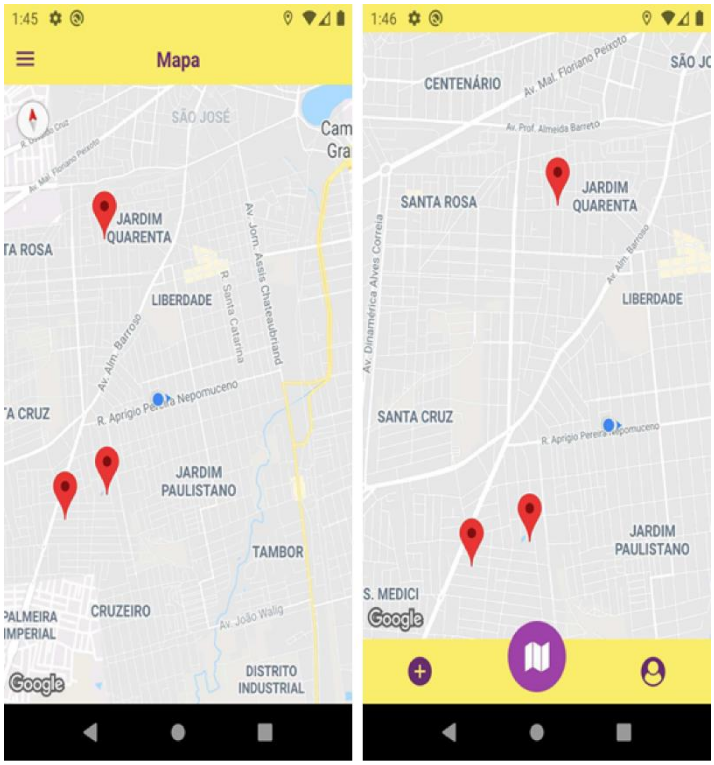
A utilização do mapa no aplicativo foi um grande acerto, pois possibilita identificar as ONGs rapidamente a partir da sua localização geográfica. Por fim, restringir o cadastro de pets somente a ONGs se torna um problema, pois acaba reduzindo a quantidade de animais a serem disponibilizados para adoção.

Os requisitos identificados no aplicativo AdotaPet CG foram:

- Realizar autenticação;
- Cadastrar ONG;
- Atualizar ONG;
- Visualizar ONG;
- Cadastrar pet;

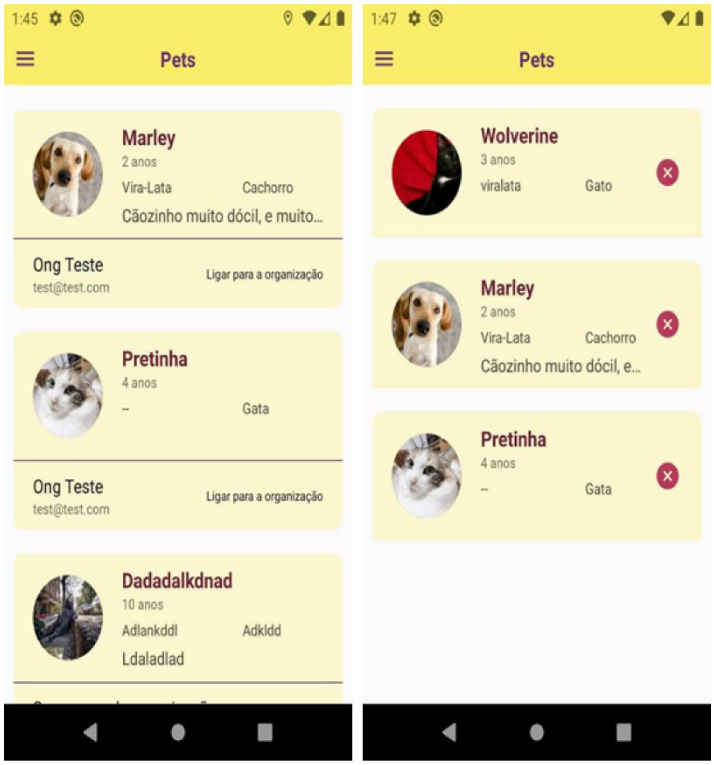
- Visualizar pet;
- Remover pet;

Figura 3 - Telas de mapa do usuário e ONG



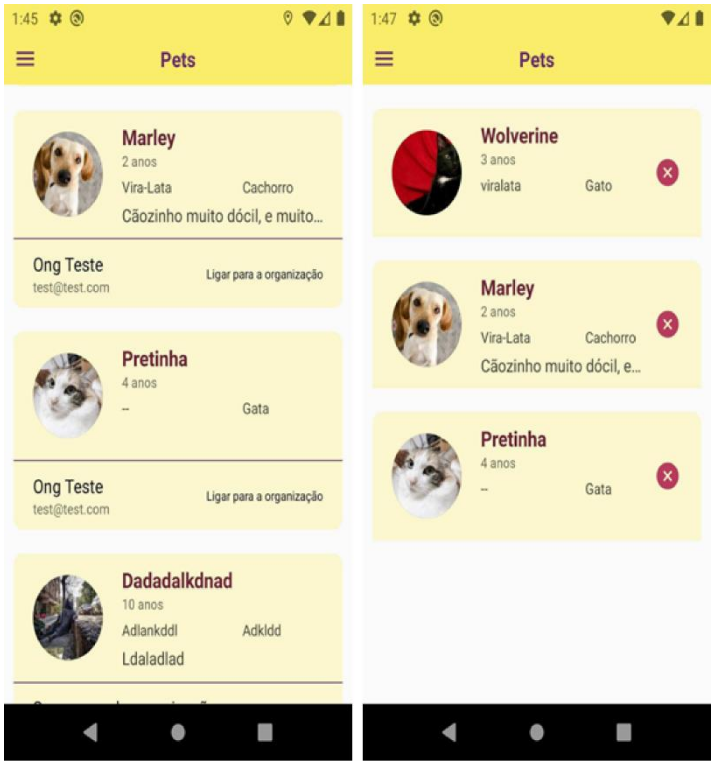
Fonte: Guimarães (2021)

Figura 4 - Tela de listagem de pets do usuário e ONG



Fonte: Guimarães (2021)

Figura 5 - Telas de cadastro e edição da ONG



Fonte: Guimarães (2021)

6.2.2 *Trabalho de Rocha e Junior*

Rocha e Junior (2021), desenvolveram um aplicativo chamado MiAuDote, com o objetivo de acelerar o processo de adoção e ajudar os animais a encontrarem novos lares e tutores. O *app* possui uma interface bonita e cores agradáveis, além de funcionalidades como login e cadastro de usuário, edição de perfil, cadastro e listagem de pets e muitas outras informações referentes ao animal, como raça, porte e idade.

Como limitação, foi identificado que o usuário não consegue realizar uma filtragem por localidade dos animais disponíveis para adoção, dessa forma, haverá possivelmente uma lista extensa de todos os animais disponíveis, dificultando a busca de um pet para adoção.

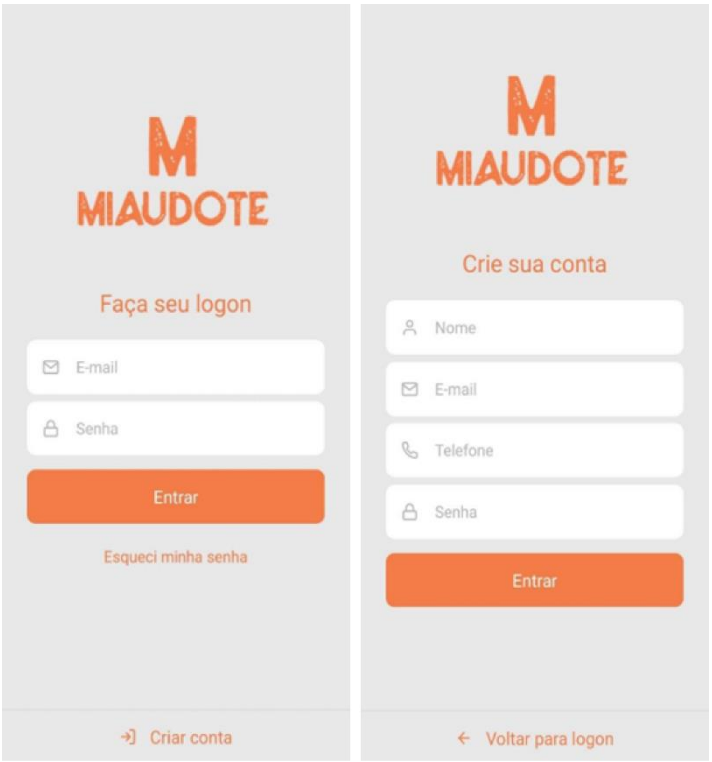
O botão para entrar em contato com o tutor do pet se torna um grande acerto, pois facilita a comunicação do possível adotante com o tutor do pet.

Por fim, restringir a visualização dos pets disponíveis para adoção a somente a cidade do autor acaba se tornando um problema, pois os usuários podem desejar adotar pets de outras cidades.

Os requisitos identificados no aplicativo MiAuDote foram:

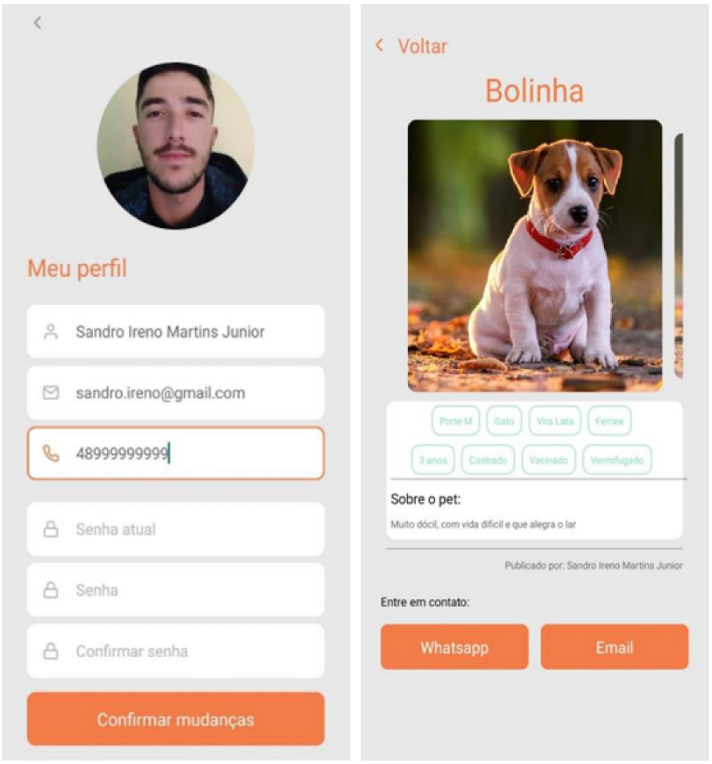
- Realizar autenticação;
- Cadastro de usuário;
- Edição de usuário;
- Cadastro de pets;
- Listagem de pets;
- Listagem dos meus pets;
- Visualização de detalhes dos pets;
- Alerta de denúncias;
- Entrar em contato com o tutor;

Figura 6 - Tela de login e cadastro



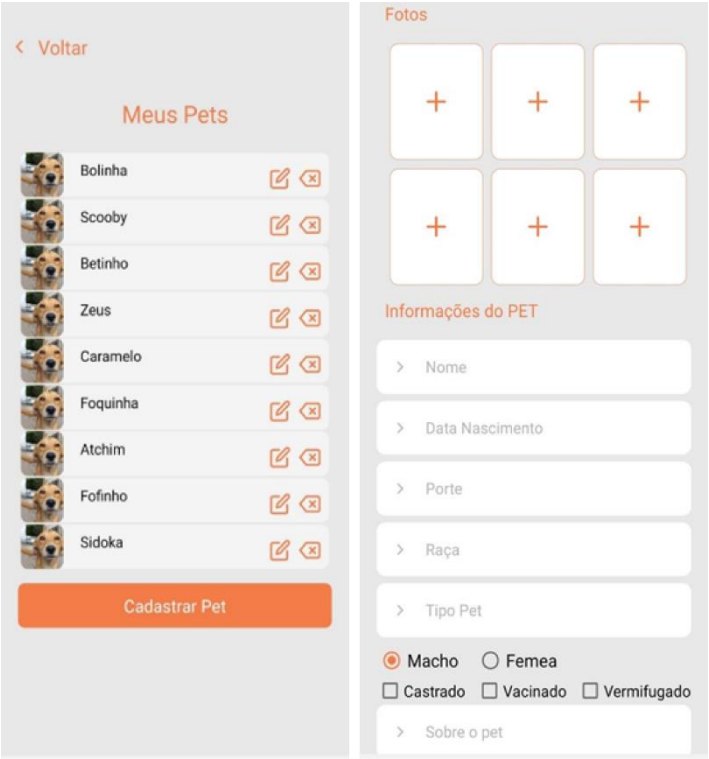
Fonte: Rocha e Junior (2021)

Figura 7 - Tela de edição de perfil e detalhes do pet



Fonte: Rocha e Junior (2021)

Figura 8 - Tela de listagem e cadastro de pets



Fonte: Rocha e Junior (2021)

6.3 Requisitos selecionados

Nesta etapa, diante do *benchmarking* realizado nos trabalhos de Guimarães (2021) e Rocha e Junior (2021), alguns requisitos foram considerados no desenvolvimento do aplicativo Adotaí.

Do aplicativo AdotaPet CG, foram considerados os seguintes requisitos: Realizar autenticação, cadastro e visualização de pets.

Do aplicativo MiAuDote, foram considerados os seguintes requisitos: Realizar autenticação, Cadastro de usuário, Edição de usuário, Cadastro de pets, Listagem de pets, Listagem dos meus pets, Visualização de detalhes dos pets e Contatar tutor.

O quadro abaixo apresenta os requisitos que foram considerados de cada aplicativo (ver Quadro 2).

Quadro 2 - Requisitos selecionados

App	Requisitos							
	Realizar autenticação	**Cadastro de usuário**	**Edição de usuário**	**Cadastro de pet**	**Listagem de pets**	**Listagem dos meus pets**	**Visualização de detalhes dos pets**	**Contatar tutor**
AdotaPet CG	X			X	X			
MiAuDote	X	X	X	X	X	X	X	X

Fonte: Elaborado pelo autor (2023)

6.4 Documentação dos requisitos

Nesta etapa, os requisitos identificados no quadro requisitos selecionados (ver Quadro 2) foram classificados e documentados como: requisitos funcionais, não-funcionais e regras de negócio (ver Quadros de 3 até 13).

Os requisitos funcionais identificados foram: Cadastro de usuário, Edição de perfil, Listagem de pets adotados, Visualização dos detalhes do pet, geolocalização e autenticação do usuário. O RF001, permite que o usuário crie o seu cadastro para poder se autenticar no aplicativo (ver Quadro 3).

Quadro 3 - Requisitos funcional: Cadastro de usuário

Identificador	RF001
Nome	Cadastro de usuário
Descrição	O aplicativo deve permitir o cadastro do usuário.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RF002, permite que o usuário edite os dados do perfil para atualizar alguma informação (ver Quadro 4).

Quadro 4 - Requisito funcional: Edição de perfil

Identificador	RF002
Nome	Edição de perfil
Descrição	O aplicativo deve permitir a edição do perfil do usuário.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RF003, permite que o usuário liste seus pets para saber quais ele cadastrou no aplicativo (ver Quadro 5).

Quadro 5 - Requisito funcional: Listagem dos pets

Identificador	RF003
Nome	Listagem dos pets
Descrição	O aplicativo deve permitir a listagem dos pets do usuário.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RF004, permite que o usuário visualize os detalhes do pet para obter informações do animal e do contato do seu respectivo dono (ver Quadro 6).

Quadro 6 - Requisitos funcional: Visualização dos detalhes do pet

Identificador	RF004
Nome	Listagem dos pets
Descrição	O aplicativo deve permitir a visualização dos detalhes do pet.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RF005, permite que o usuário visualize os pets pelo mapa para encontrar sua localização (ver Quadro 7).

Quadro 7 - Requisito funcional: Geolocalização

Identificador	RF005
Nome	Geolocalização
Descrição	O aplicativo deve permitir geolocalização dos pets.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RF006, permite que o usuário se autentique para utilizar as funcionalidades do aplicativo (ver Quadro 8).

Quadro 8- Requisito funcional: Autenticação

Identificador	RF006
Nome	Autenticação
Descrição	O aplicativo deve permitir autenticação do usuário.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

Os requisitos não-funcionais identificados foram referentes a portabilidade, segurança e integração. O RNF001 é referente a portabilidade, ou seja, o aplicativo deve ser executado nos sistemas operacionais *Android* e *iOS* (ver Quadro 9).

Quadro 9 - Requisito não-funcional: Portabilidade

Identificador	RNF001
Nome	Portabilidade
Descrição	O aplicativo deve ser executado nas plataformas Android e iOS.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RNF002 é referente a segurança, ou seja, o aplicativo deve validar os dados de autenticação do usuário (ver Quadro 10).

Quadro 10 - Requisito não-funcional: Segurança

Identificador	RNF002
Nome	Segurança
Descrição	O aplicativo deve validar as informações de autenticação do usuário.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

O RNF003 é referente a integração, ou seja, o aplicativo deve integrar com whatsapp, e-mail e dados do IBGE (ver Quadro 11).

Quadro 11 - Requisito não-funcional: Integração

Identificador	RNF003
Nome	Integração
Descrição	O aplicativo deve integrar com whatsapp, e-mail e o IBGE.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

As regras de negócio identificadas foram referentes a cadastro e permissão. A RN001 é referente a cadastro, ou seja, o usuário precisará criar um cadastro para poder autenticar na aplicação (ver Quadro 12).

Quadro 12 - Regras de negócio: Cadastro

Identificador	RN001
Nome	Permissão
Descrição	O usuário deverá criar um cadastro para poder autenticar na aplicação.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

A RN002 é referente a permissão, ou seja, o usuário só poderá ter acesso às funcionalidades do aplicativo após estar autenticado (ver Quadro 13).

Quadro 13 - Regras de negócio: Permissão

Identificador	RN002
Nome	Permissão
Descrição	O aplicativo deve permitir acesso às funcionalidades ao usuário somente após a autenticação.
Prioridade	Essencial

Fonte: Elaborado pelo autor (2023)

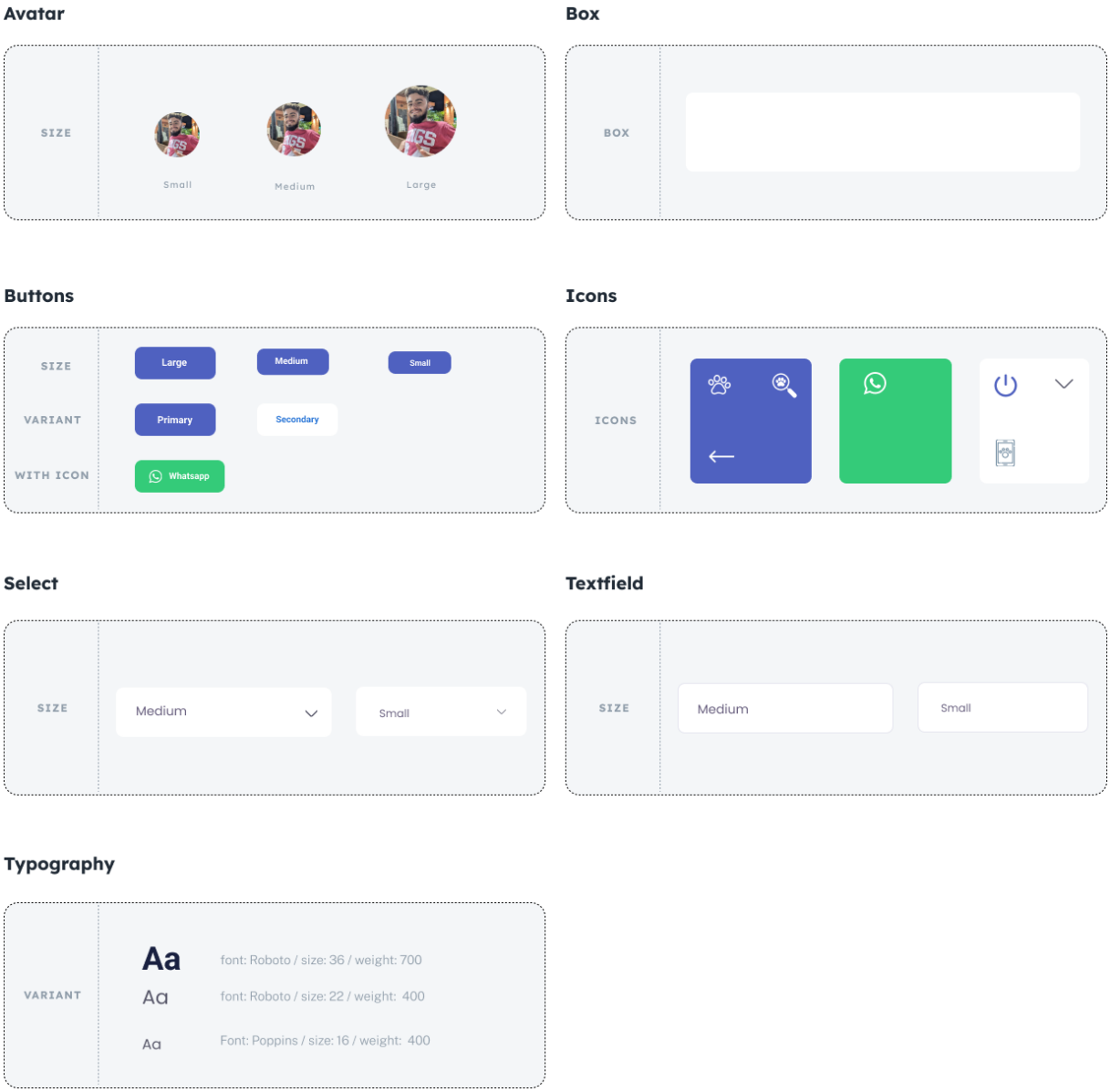
6.5 Criação dos componentes

Nesta etapa, foram prototipados os componentes utilizando a ferramenta *Figma* e o *Atomic Design* como metodologia, classificando-os de forma hierárquica entre: Átomos, Moléculas, Organismos e Templates (ver Figura 9, 10, 11 e 12).

6.5.1 *Átomos*

Os átomos de nossas interfaces servem como os blocos de construção que compõem todas as nossas interfaces de usuário. Esses átomos incluem elementos básicos como campo de texto, botões, labels e outros que não podem ser divididos sem deixar de ser funcionais (FROST, 2016).

Figura 9 – Componentes - Átomos

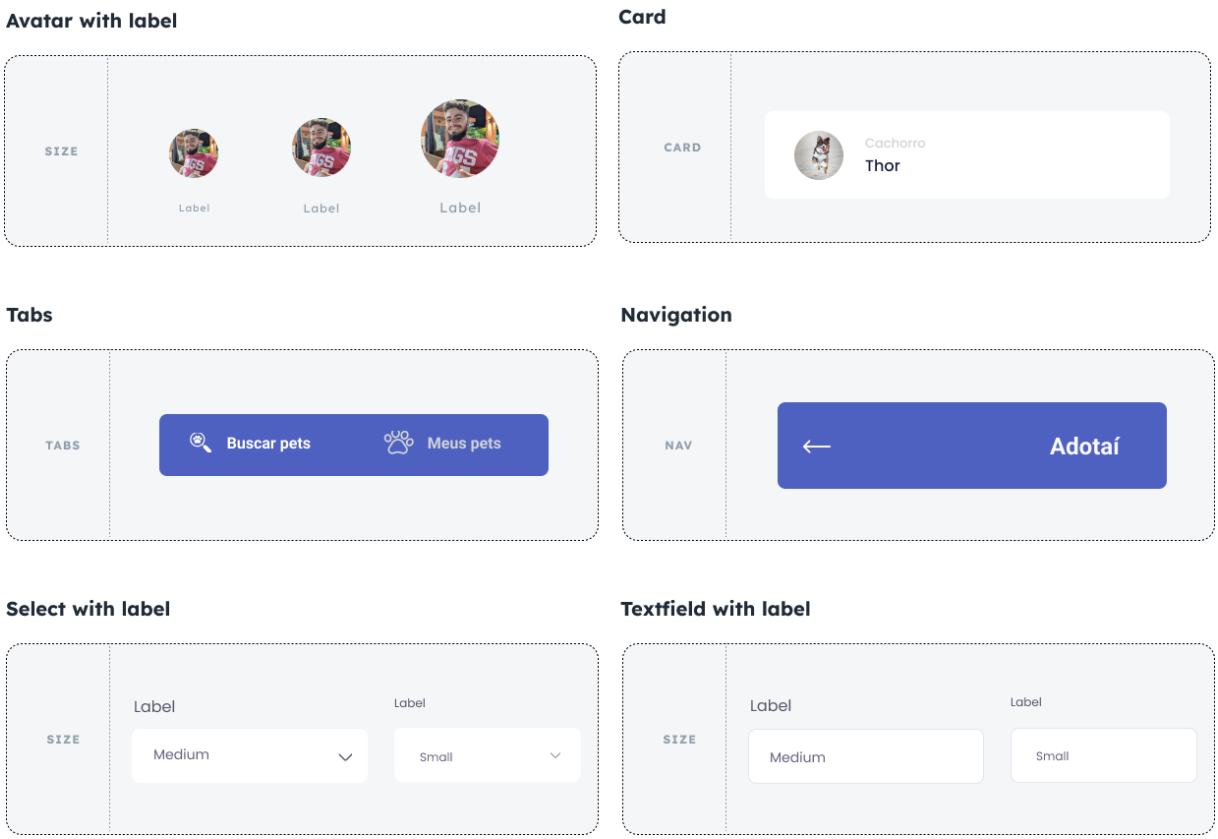


Fonte: Elaborado pelo autor (2023)

6.5.2 *Moléculas*

As moléculas são grupos relativamente simples de elementos da interface do usuário que funcionam juntos como uma unidade. Por exemplo, um label, campo de texto e botão podem se unir para criar uma molécula de formulário de busca (FROST, 2016).

Figura 10 – Componentes - Moléculas

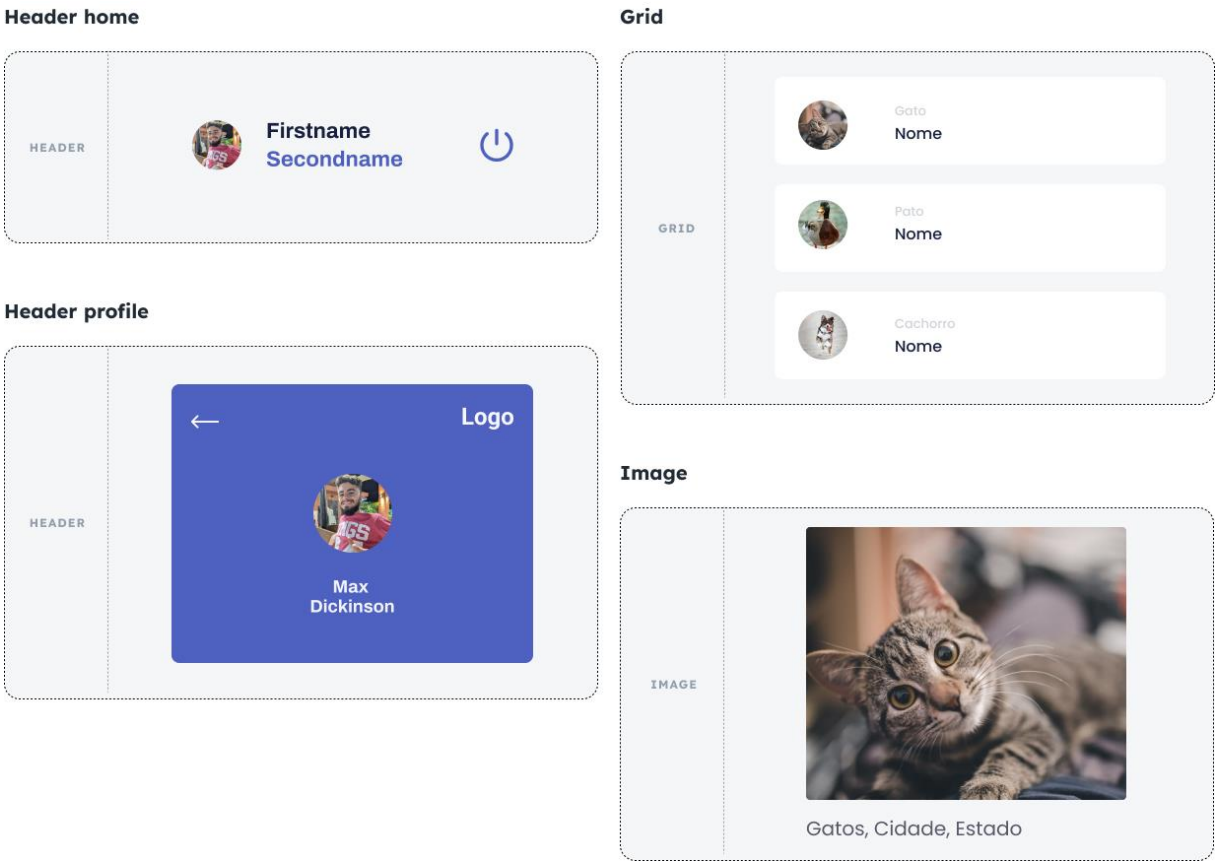


Fonte: Elaborado pelo autor (2023)

6.5.3 *Organismos*

Organismos são componentes de interface de usuário relativamente complexos compostos por grupo de moléculas e/ou átomos e/outros organismos. Por exemplo, um organismos de cabeçalho pode consistir em elementos diferentes, como uma imagem, lista de navegação e um formulário de pesquisa (FROST, 2016).

Figura 11 – Componentes - Organismos

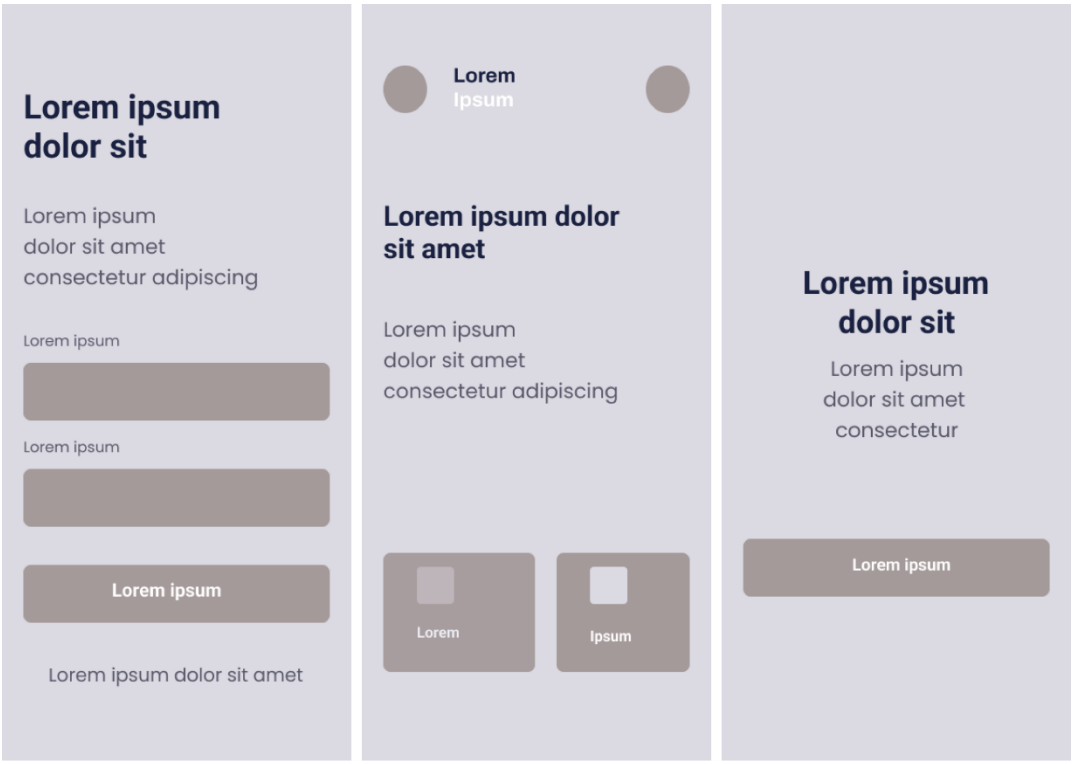


Fonte: Elaborado pelo autor (2023)

6.5.4 *Templates*

Os *templates* são objetos de nível de página que colocam componentes em um *layout* e articulam a estrutura de conteúdo. Por exemplo, podemos pegar o organismo de cabeçalho e aplicá-lo a um *template* de página inicial. Este template de página inicial exibe todos os componentes necessários da página funcionando juntos (FROST, 2016).

Figura 12 - Template de telas do app



Fonte: Elaborado pelo autor (2023)

6.6 Criação das telas

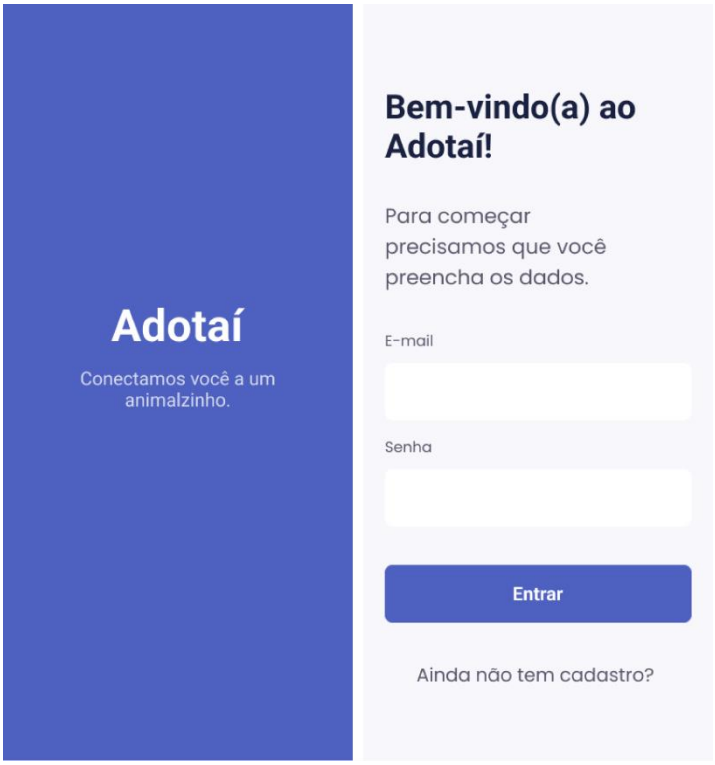
Nesta etapa, foi prototipada a última fase do *Atomic Design*, que são as páginas, para isso, utilizou-se dos componentes desenvolvidos nas etapas anteriores. A ferramenta utilizada foi o *Figma*.

6.6.1 *Páginas*

As páginas são instâncias específicas de modelos que mostram a aparência de uma interface do usuário com um conteúdo representativo real (FROST, 2016).

A figura abaixo apresenta a tela de *splash*, que é a tela exibida para o usuário enquanto o aplicativo está carregando. A segunda tela é a tela de login, onde o usuário pode inserir seus dados e entrar na aplicação. Caso ainda não tenha realizado o seu cadastro, poderá navegar para tela de cadastro (ver Figura 13).

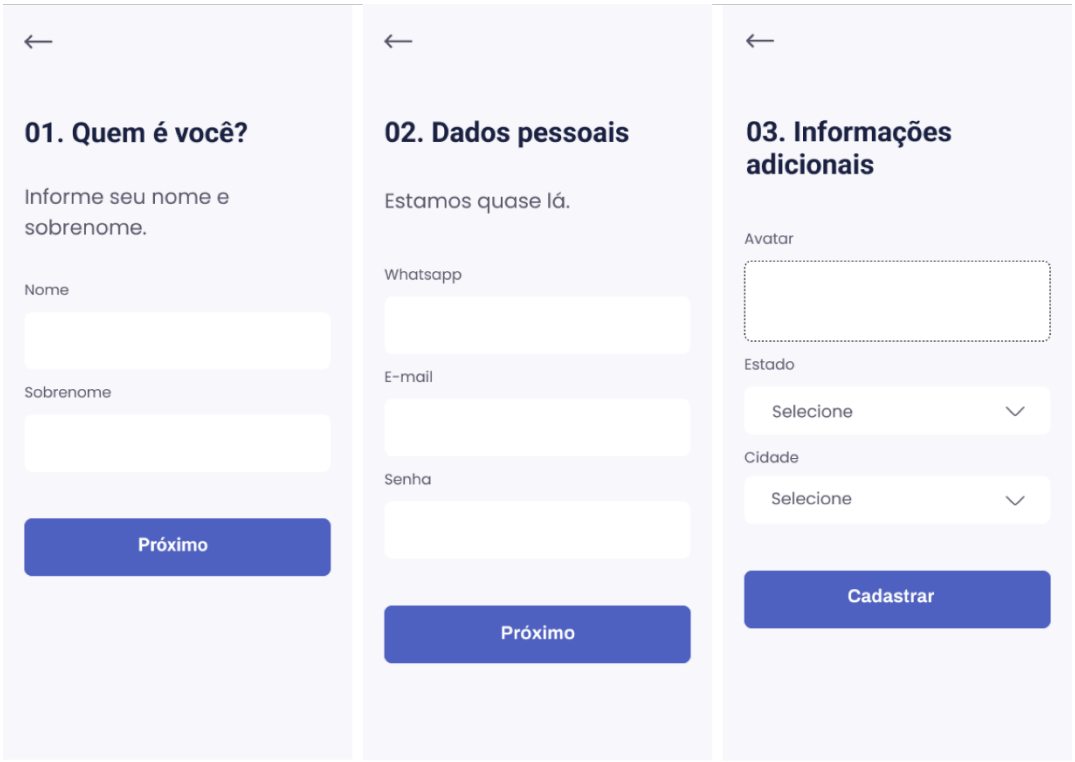
Figura 13 - Páginas – Telas de splash e login



Fonte: Elaborado pelo autor (2023)

A figura abaixo apresenta as telas responsáveis pelo cadastro do usuário. Na primeira tela, o usuário deve informar nome e sobrenome. Na segunda tela, o usuário deve informar o whatsapp, e-mail e senha, e na terceira tela, o usuário pode fazer o upload de uma foto e selecionar o seu estado e a sua cidade (ver Figura 14).

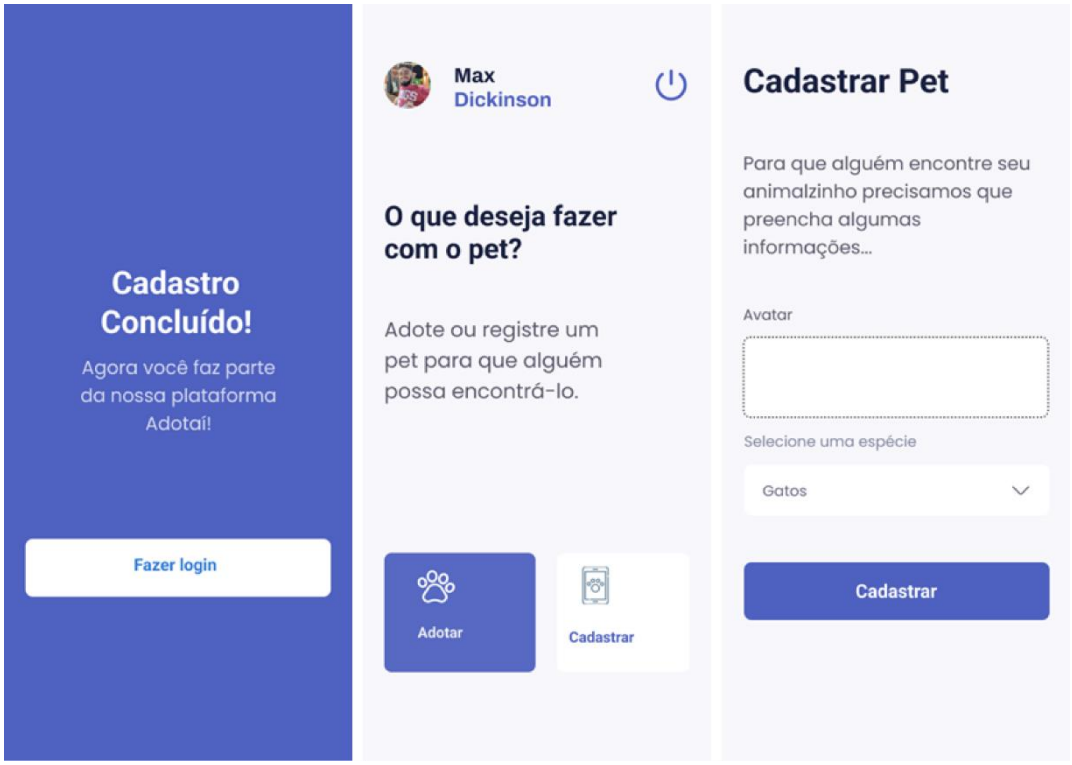
Figura 14 - Páginas - Páginas de cadastro



Fonte: Elaborado pelo autor (2023)

A figura abaixo apresenta a tela de cadastro concluído, que é a tela para onde o usuário é redirecionado quando conclui o seu cadastro. A segunda tela é a tela inicial, esta é a tela pra onde o usuário é redirecionado quando realiza o login, ainda nesta tela, o usuário pode adotar ou cadastrar um pet, editar o seu perfil e fazer logout. A terceira tela é responsável pelo cadastro do pet, nesta tela o usuário irá cadastrar o pet a ser doado (ver Figura 15).

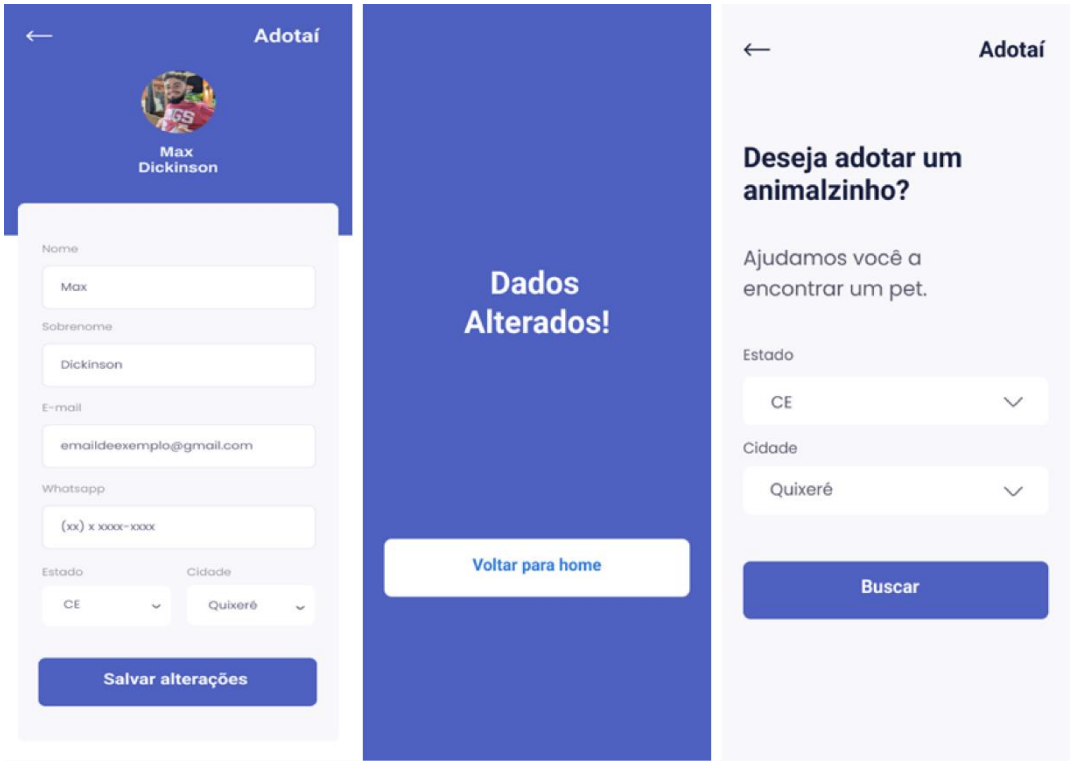
Figura 15 - Páginas – Telas de cadastro concluído, inicial e cadastro de pet



Fonte: Elaborado pelo autor (2023)

A figura abaixo apresenta a tela de edição do usuário, que é a tela onde o usuário poderá atualizar os seus dados. A segunda tela é a de dados alterados, onde o usuário é redirecionado para ela quando atualiza os dados. A terceira tela é a de busca de pets, o usuário seleciona o estado e a cidade que deseja encontrar o pet e faz a busca, em seguida ele é redirecionado para a tela do mapa que terá os pets disponíveis para adoção (ver Figura 16).

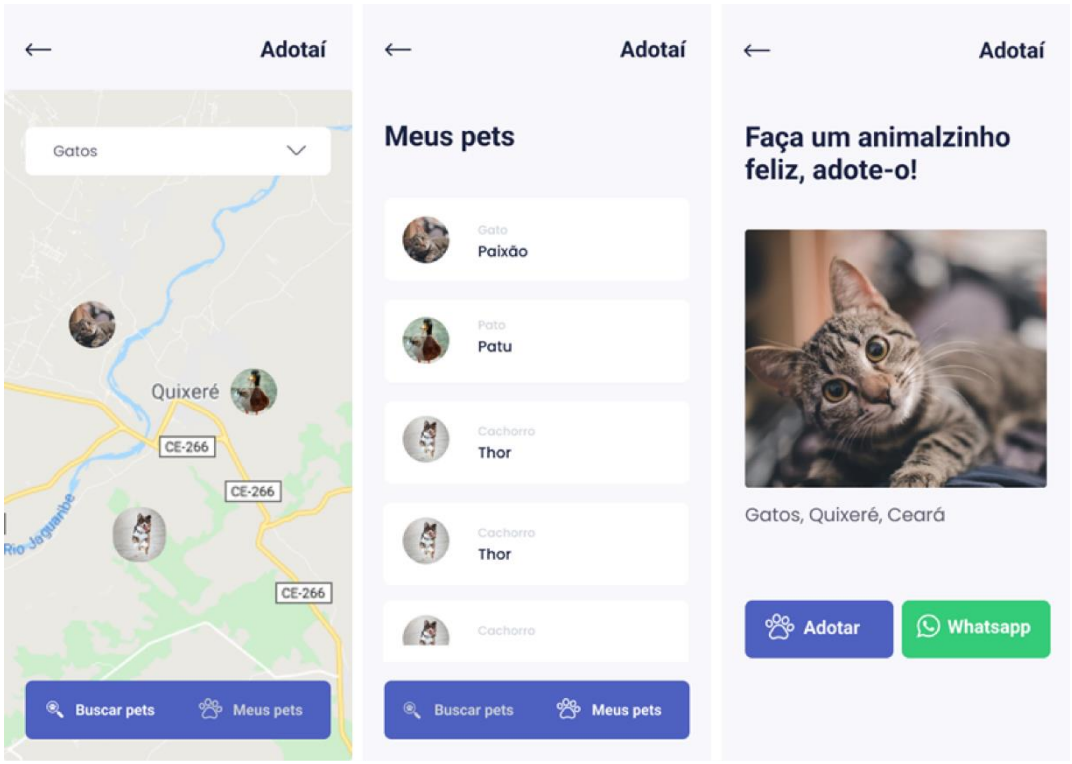
Figura 16 - Páginas - Páginas de edição de usuário, dados alterados e busca de pets



Fonte: Elaborado pelo autor (2023)

A figura abaixo apresenta as telas de mapa, meus pets e detalhes do pet. A primeira tela exibe os animais disponíveis para adoção no mapa, com base na espécie selecionada nesta tela e no estado e cidade selecionados na etapa anterior. A segunda tela exibe a lista de pets do usuário. A terceira tela exibe a imagem do pet que foi selecionado no mapa, ainda nesta tela, o usuário pode solicitar a adoção do pet ou entrar em contato com o seu dono (ver Figura 17).

Figura 17 - Páginas – Telas de localização, listagem e detalhes do pet



Fonte: Elaborado pelo autor (2023)

6.7 Aplicação do questionário

Nesta etapa, foi aplicado um questionário aos usuários com o intuito de validar os protótipos e verificar se o que foi projetado atende às necessidades dos futuros usuários. O formulário foi disponibilizado no *Google Forms* contendo o *link* das telas prototipadas. Ao todo, o questionário ficou aberto por um período de 48h e obteve um total de 18 respostas por pessoas com idade entre 20 e 60 anos. Todas as pessoas preencheram o TCLE (Termo de consentimento livre e esclarecido) por se tratar de pesquisa com seres humanos. O questionário elaborado se encontra no apêndice.

O questionário foi elaborado com 13 perguntas relacionadas a facilidade em utilizar o *app*, *design* do aplicativo e utilidade das funcionalidades propostas. Das 13 perguntas elaboradas, 03 foram perguntas abertas relacionadas ao nome e cidade do participante e sugestões de melhoria do *app*, 10 foram fechadas relacionadas ao *design*, facilidade na utilização e grau de satisfação com o aplicativo.

Para as respostas fechadas, foi utilizada a escala Likert, que é uma escala de resposta muito utilizada em questionários para medir o grau de concordância ou não com uma afirmação.

No questionário aplicado, foi utilizada a escala Likert de 05 pontos com as seguintes alternativas:

- Discordo totalmente
- Discordo
- Indiferente
- Concordo
- Concordo totalmente

O quadro abaixo apresenta as perguntas elaboradas para o questionário, contendo o nome da pergunta e o tipo de cada uma delas, se é uma pergunta fechada ou não (ver Quadro 14).

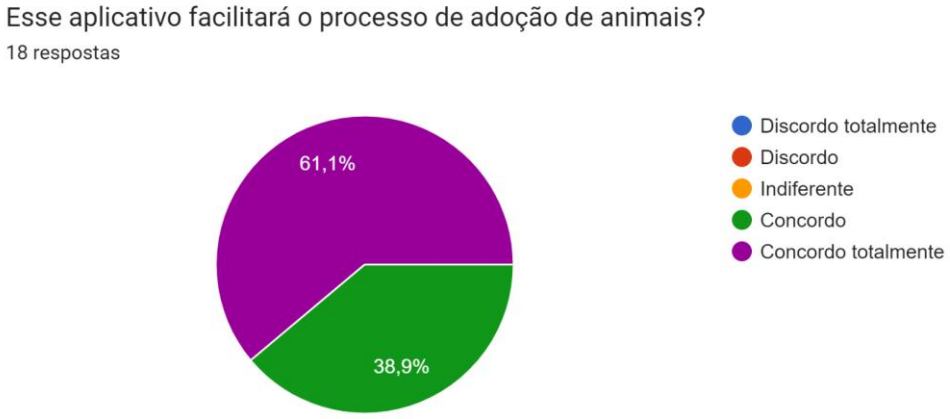
Quadro 14 - Questionário - Perguntas

Número	Pergunta	Tipo de pergunta
01	Qual seu nome?	Aberta
02	Qual sua faixa etária?	Fechada
03	Em qual cidade você mora?	Aberta
04	Esse aplicativo facilitará o processo de adoção de animais?	Fechada
05	O processo de cadastro de pets no aplicativo é fácil de ser realizado?	Fechada
06	A visualização dos pets no mapa é uma funcionalidade útil?	Fechada
07	As cores do aplicativo proporcionam uma sensação agradável?	Fechada
08	Os textos do aplicativo descrevem bem o objetivo de cada tela, deixando claro o que será realizado?	Fechada
09	Os ícones do aplicativo condizem com a informação apresentada?	Fechada
10	Os botões do aplicativo realizam as ações conforme estão descritos?	Fechada
11	Que outras funcionalidades você sugere para o aplicativo? Fique à vontade para sugeri-las.	Aberta
12	Você utilizaria esse app para auxiliar no processo de doação/adoção?	Fechada
13	Com base no aplicativo apresentado, qual seu nível de satisfação para os critérios abaixo:	Fechada

Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam os resultados referentes às respostas do questionário. Em relação a facilidade que o aplicativo proporcionará, 61,1% responderam com Concordo totalmente, enquanto que 38,9% responderam com Concordo (ver Figura 25).

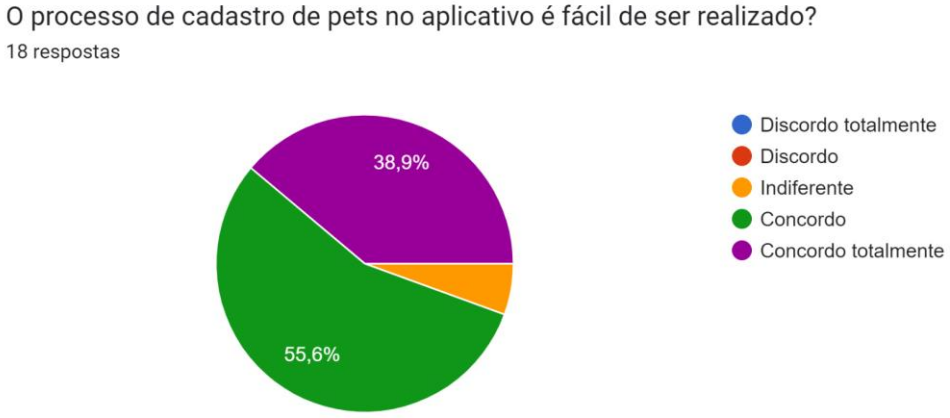
Figura 18 - Questionário - Resposta 04



Fonte: Elaborado pelo autor (2023)

Em relação a facilidade no processo de cadastro dos pets, 55,6% responderam com Concordo, 38.9% responderam com Concordo totalmente e 5,6% responderam com Indiferente (ver Figura 26).

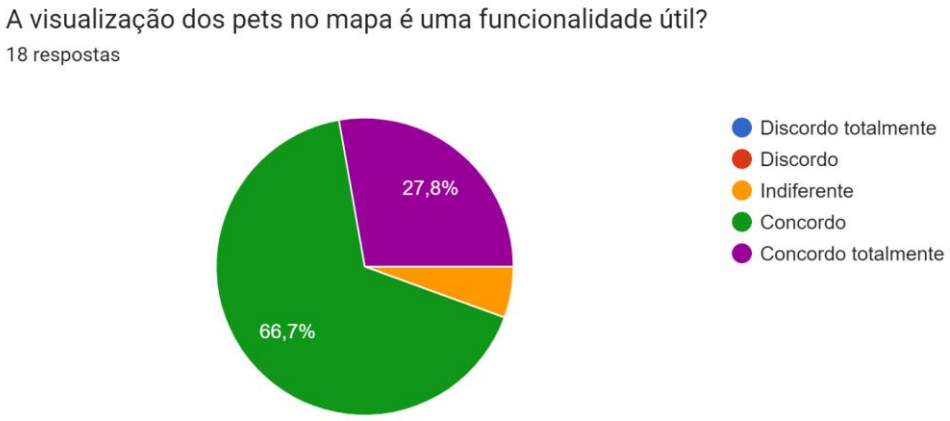
Figura 19 - Questionário - Resposta 05



Fonte: Elaborado pelo autor (2023)

Em relação à utilidade da visualização dos pets no mapa, 66,7% responderam com Concordo, 27,8% responderam com Concordo totalmente e 5,6% responderam com Indiferente (ver Figura 27).

Figura 20 - Questionário - Resposta 06



Fonte: Elaborado pelo autor (2023)

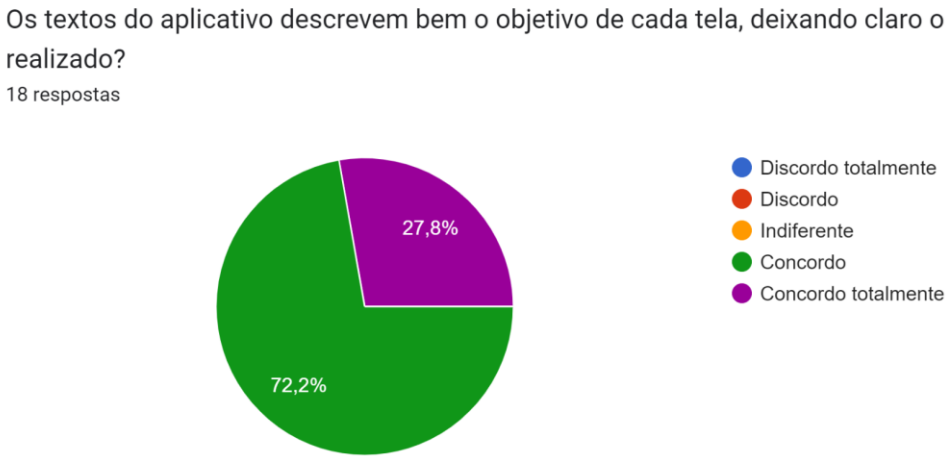
Em relação a sensação agradável que as cores do aplicativo proporcionam, 72,2% responderam com Concordo, 22,2% responderam com Concordo totalmente e 5,6% responderam com Indiferente (ver Figura 28).

Figura 21 - Questionário - Resposta 07

Fonte: Elaborado pelo autor (2023)

Em relação a clareza em que os textos descrevem o que será realizado, 72,2% responderam com Concordo, enquanto 27,8% responderam com Concordo totalmente (ver Figura 29).

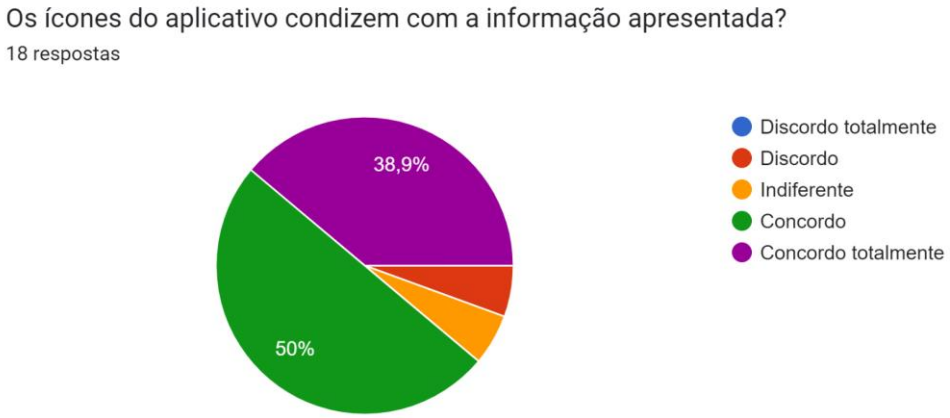
Figura 22 - Questionário - Resposta 08



Fonte: Elaborado pelo autor (2023)

Em relação a consistência dos ícones com a informação apresentada, 50% responderam com Concordo, 38,9% responderam com Concordo totalmente, 5,6% responderam com Discordo e 5,6 com Indiferente (ver Figura 30).

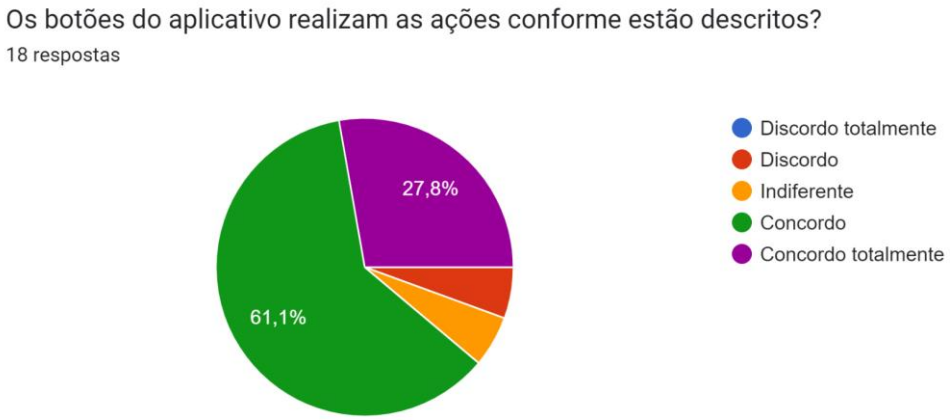
Figura 23 - Questionário - Resposta 09



Fonte: Elaborado pelo autor (2023)

Em relação a consistência dos textos nos botões com as ações realizadas, 61,6% responderam com Concordo, 27,8% responderam com Concordo totalmente, 5,6% responderam com Discordo e 5,6% responderam com Indiferente (ver Figura 31).

Figura 24 - Questionário - Resposta 10



Fonte: Elaborado pelo autor (2023)

Em relação a utilização do app, 94,4% responderam com Sim enquanto 5,6% responderam com Não (ver Figura 32).

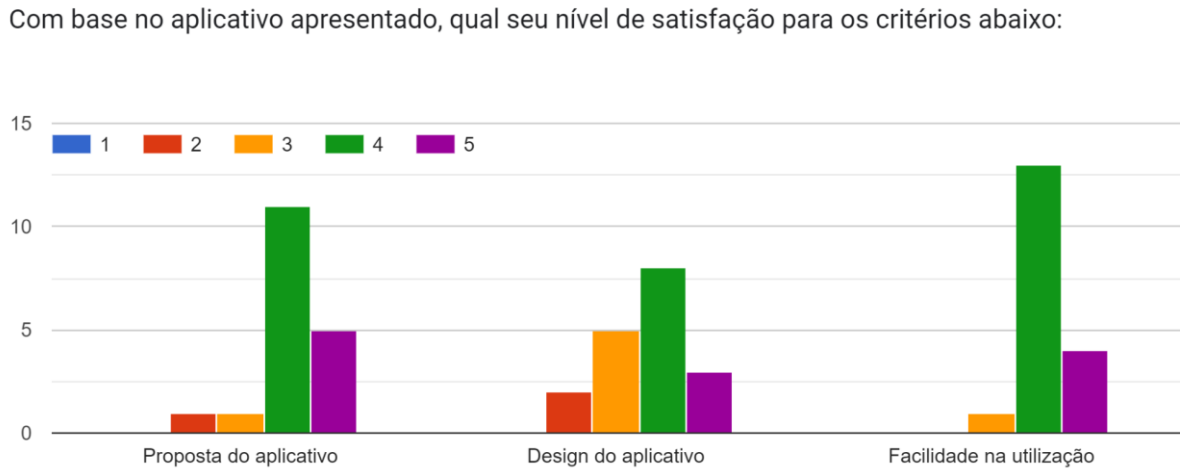
Figura 25 - Questionário - Resposta 12

Fonte: Elaborado pelo autor (2023)

Em relação a satisfação do usuário, numa escala de 1 a 5 e no critério de Proposta do aplicativo, 11 pessoas avaliaram em 4, 5 pessoas avaliaram em 5, uma pessoa avaliou em 2 e uma pessoa avaliou em 3. Já no critério Design do aplicativo, 8 pessoas avaliaram em 4, 3

pessoas avaliaram em 5, 5 pessoas avaliaram em 3 e duas pessoas avaliaram em 2. E finalmente, no critério Facilidade na utilização, 13 pessoas avaliaram em 4, 4 pessoas avaliaram em 5, e uma pessoa avaliou em 3 (ver Figura 33).

Figura 26 - Questionário - Resposta 13



Fonte: Elaborado pelo autor (2023)

O quadro abaixo (ver Quadro 15) apresenta a lista de sugestões dos participantes referente à pergunta 11 (ver Quadro 14).

Quadro 15 - Questionário - Lista de sugestões

Número	Resposta
01	Mais informações sobre o pet cadastrado, estado de saude, idade, fêmea ou macho, raça, dócil ou não, vacinado ou não.
02	Acho que no cadastro do pet, poderia incluir mais informações sobre o pet.
03	Agendar um tipo de pet e ao aparecer notificar ao interessado
04	Poderia ter mais informações sobre os animais, por exemplo, idade.
05	Reputação do adotante.Já tivemos problemas com adotantes que depois de adotar abandonaram os animais mas rua novamente. Deveria haver reputação de todos como com que frequência adota, normalmente irresponsáveis gostam de criar mas não cuidam como se deve.
06	Poderia por nomes e CPF, e chat pra local e encontro, pois não acho seguro encontrar um desconhecido...
07	Mudar a cor do app, colocar um termo de responsabilidade para os turores, deixar a interface mais "animalesca".

08	Alguma forma de especificação no mapa se já foi adotado ou não, mais informações sobre o pet: sexo, idade etc, e também algum tipo de recompensa no App a cada pet adotado ou registrado, assim as pessoas irão fazer algo útil e de ajuda e meio que ainda ser recompensadas , não financeiramente, mas algum tipo de classificação no App ou medalhas, mas como é só um protótipo, fica as ideias!
09	Talvez um sistema de avaliação onde os usuários próximos à pessoa que adotou poderiam informar se o animal está sendo bem cuidado.
10	Uma sessão para pets desaparecidos, onde o dono poderia cadastrar seu pet e colocar informações úteis de forma similar ao cadastro de pets para adoção, e quem encontrasse poderia informar o dono. Mas isso poderia até ser um outro aplicativo.

Fonte: Elaborado pelo autor (2023)

Diante da análise da aplicação do questionário, percebeu-se que o protótipo atende as necessidades, mas que necessita de melhorias a serem realizadas quanto a mais informações sobre o pet como sexo, idade, raça, e se o pet já foi adotado ou não.

6.8 Desenvolvimento

Nesta seção, são apresentadas as etapas realizadas no desenvolvimento do aplicativo, onde foi desenvolvido o *back-end* e o *front-end*. A camada do *back-end* foi desenvolvida utilizando o *framework Node.js* e o *front-end* utilizando o *React Native*. Foi ainda projetado um diagrama de caso de uso para exmplificar o funcionamento da aplicação representando as ações do usuário no sistema e um diagrama de entidade relacionamento. Por fim, será apresentado as telas da aplicação desenvolvida.

6.8.1.1 Histórico de desenvolvimento

Inicialmente, foi desenvolvido o Diagrama de Caso de Uso a partir dos requisitos funcionais obtidos na etapa de Documentação dos Requisitos. Com os requisitos definidos, foi possível então visualizar as funcionalidades que seriam utilizadas pelo usuário. Em seguida, foi desenvolvido o Diagrama de Entidade Relacionamento, etapa essencial para o desenvolvimento do back-end, pois descreve os atributos que as entidades terão e como estarão relacionadas entre si.

Após isso, foi iniciado o desenvolvimento do *back-end* utilizando o *Node.js*, onde o mesmo foi escolhido por ser um ambiente de execução JavaScript no lado do servidor, e desse modo, seria possível reaproveitar pelo autor todo o conhecimento na linguagem JavaScript, reduzindo bastante o tempo de desenvolvimento. Foi ainda utilizado o padrão *Clean*

Architecture no *back-end*, visando a sepração das responsabilidades em camadas, e permitindo isolar as regras de negócio de qualquer outro *framework*.

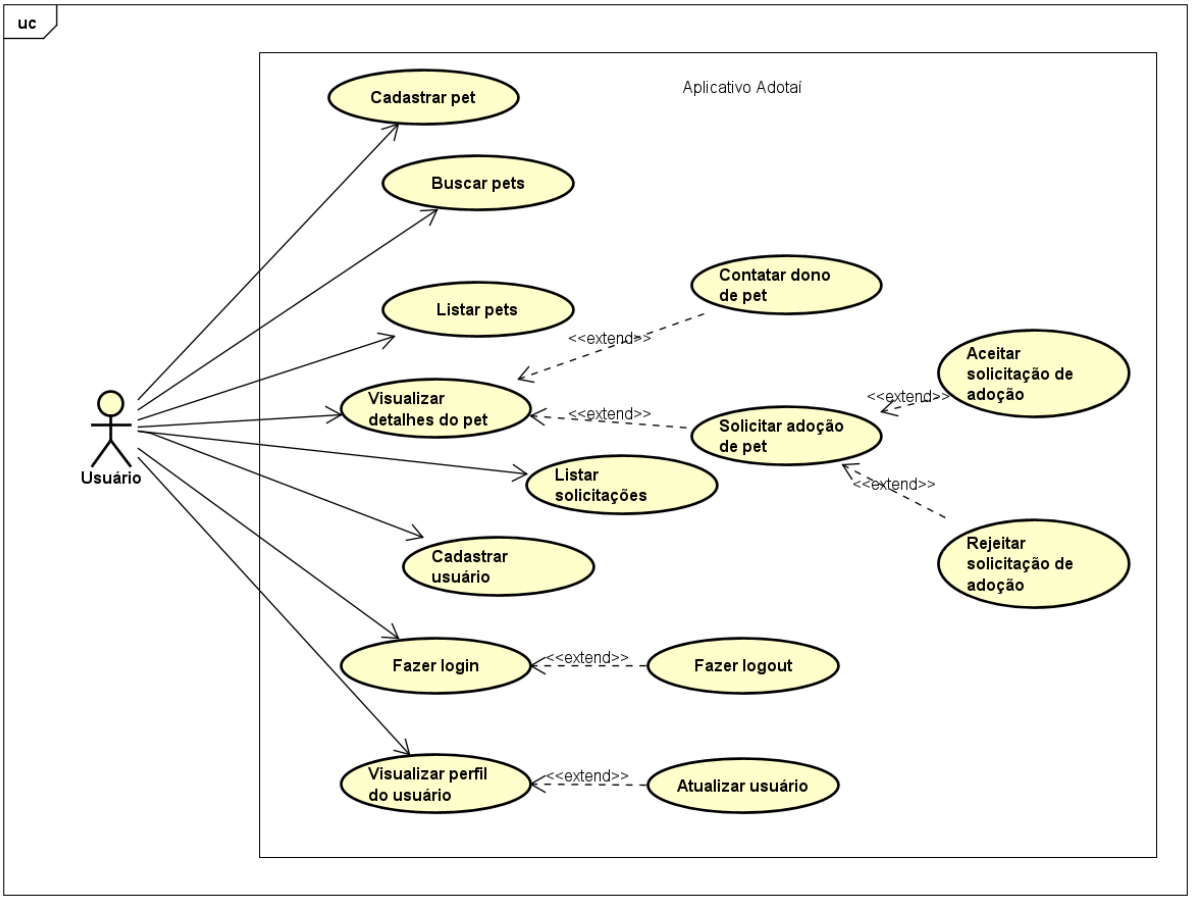
Por fim, foi realizado o desenvolvimento do aplicativo utilizando o *React Native* e a ferramenta *Expo* para auxiliar na execução do ambiente de desenvolvimento, onde inicialmente foram criados os componentes seguindo o *Atomic Design*, em seguida, foram criadas as páginas e por último, ocorreu a integração entre *back-end* e *front-end*.

6.8.1.2 Diagrama de Caso de Uso

Casos de uso são abstrações de pequenas histórias narrativas envolvendo a iteração entre um ou mais usuários (chamados de atores) e o sistema. A idéia é que estes casos de uso representem, por meio dessas pequenas histórias, as funcionalidades de um sistema. Imagina-se um conjunto de atores necessários a operar o sistema e passa-se a descrever o fluxo dos acontecimentos, onde o ator executa uma ação, e o sistema responde de alguma maneira a essa ação, até que alguma funcionalidade tenha sido contemplada (GUDWIN, 2010).

A figura abaixo apresenta o diagrama de caso de uso da aplicação composta do ator usuário e as funcionalidades do sistema, onde o usuário consegue se cadastrar, cadastrar um pet, visualizar os detalhes do pet podendo contatar o dono ou solicitar a adoção, entre outras funcionalidades exibidas abaixo (ver Figura 27).

Figura 27 - Diagrama de caso de uso



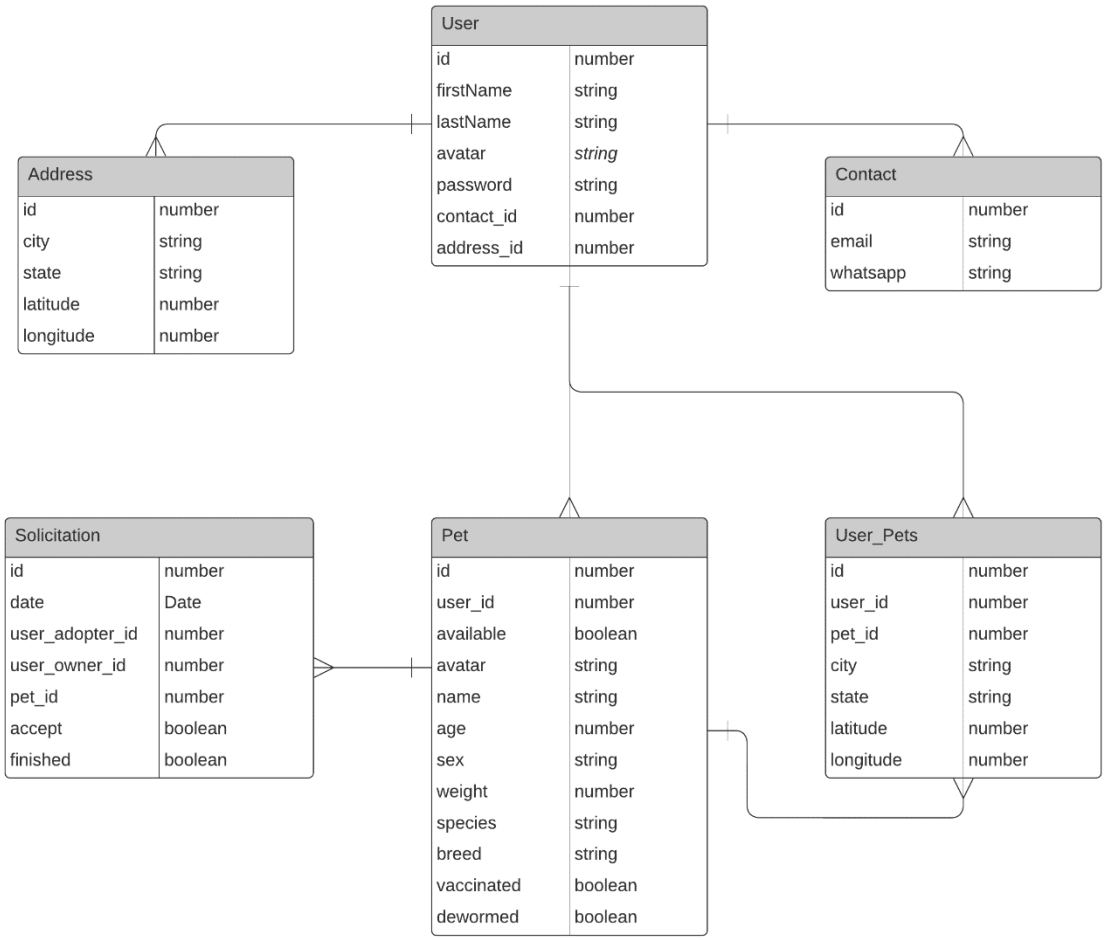
Fonte: Elaborado pelo autor (2023)

6.8.1.3 Diagrama de Entidade Relacionamento

Um diagrama entidade relacionamento (ER) é um tipo de fluxograma que ilustra como “entidades”, pessoas, objetos ou conceitos, se relacionam entre si dentro de um sistema (NOGUEIRA, 1988).

Como forma de estruturar o banco de dados, foi criado um diagrama de entidade relacionamento com as entidades que irão compor o sistema e os dados que serão armazenados. A entidade *User* possui um relacionamento com a entidade *Address*, *Contact e Pet* pois um usuário possui um endereço, informações de contato e uma lista de pets associados a ele. A figura abaixo apresenta o Diagrama de Entidade Relacionamento onde contém as entidades *User*, *Address*, *Contact*, *Pet*, *Solicitation* e seus relacionamentos (ver Figura 28).

Figura 28 - Diagrama entidade relacionamento



Fonte: Elaborado pelo autor (2023)

6.8.1.4 Arquitetura da aplicação

A arquitetura da aplicação foi baseada na arquitetura REST. REST é a sigla para "*Representational State Transfer*", e trata-se de um conjunto de restrições de arquitetura. As restrições que determinam uma arquitetura REST são – cliente-servidor, interface uniforme, *stateless*, cache e camadas (NOLETO, 2022).

API REST é uma interface de programação de aplicações (API) que está em conformidade com o estilo de arquitetura REST (GUPTA, 2022). Segundo Barro (2022), uma API REST funciona através da manipulação de recursos e representações. Essas representações

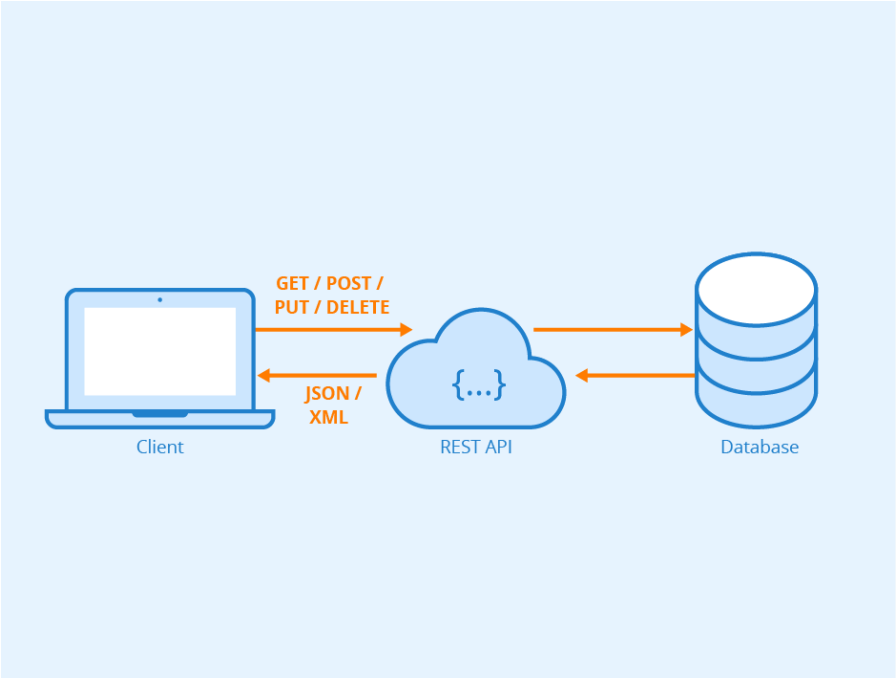
são trocadas entre o cliente e o servidor através de uma interface padronizada e de um protocolo de comunicação específico como o HTTP.

A comunicação utilizando uma API REST ocorre através dos métodos HTTP que entregam ao solicitante a informação representada em diversos formatos. O formato JSON (*Javascript Object Notation*) é o formato mais utilizado porque, apesar de seu nome, é independente de qualquer outra linguagem e pode ser lido por máquinas e humanos (REDHAT, 2020).

O estilo arquitetural REST, enfatiza a escalabilidade das interações entre componentes, a generalização das interfaces, o desenvolvimento independente de componentes e a utilização de componentes intermediários para reduzir a latência das interações, reforçar a segurança e encapsular sistemas legados (FIELDING, 2000).

A figura abaixo exemplifica o funcionamento de uma API REST utilizando os métodos HTTP. O cliente envia uma solicitação para a API usando um método HTTP, a API se comunica com o banco de dados e retorna a informação no formato JSON para o cliente (ver Figura 29).

Figura 29 - Arquitetura REST



Fonte: Seobillity

6.8.1.5 Back-end

O back-end é a parte da aplicação que não vemos, é também referido como *server-side*. O back-end executa as regras de negócio da aplicação, ele é responsável por validar e consistir os dados vindos do front-end (AMARAL e NERIS, 2015). Para Roveda (2021), back-end é toda a parte da programação voltada ao funcionamento interno de um software. Em outras palavras, back-end é tudo aquilo que está por trás da interface de uma aplicação: seus sistemas, banco de dados, segurança, envio e recebimento de informações e armazenamento.

6.8.1.6 Node.js

Node.js é um ambiente de tempo de execução JavaScript baseado no interpretador V8 do Google Chrome, e que permite a execução de código JavaScript fora de um navegador web. Puluceno (2012), afirma que o *Node* possibilita a criação de aplicações escalonáveis e de alta concorrência, pelo fato de possuir uma arquitetura orientada a eventos onde as operações de entrada e saída não bloqueiam o sistema enquanto aguardam por uma resposta. Para a implementação do back-end, foi utilizado o *framework* descrito acima.

6.8.1.7 Clean Architecture

Clean Architecture "Arquitetura Limpa", é um padrão arquitetural de desenvolvimento de software criado em 2012 por Robert C. Martin. Esse padrão propõe a separação das responsabilidades dividindo o código em camadas, de modo que cada camada seja independente de qualquer outra.

De acordo com Martin (2012), um sistema construído utilizando a arquitetura limpa produz sistemas que são:

- **Independente de Frameworks:** A arquitetura não depende da existência da alguma biblioteca de software carregada de recursos. Isso permite que você use tais estruturas como ferramentas, em vez de ter que amontoar seu sistema em suas restrições limitadas.
- **Testável:** As regras de negócio podem ser testadas sem interface do usuário, banco de dados, servidor web ou qualquer outro elemento externo.
- **Independente da UI:** A UI (User Interface) pode mudar facilmente, sem alterar o resto do sistema. Uma UI da Web pode ser substituída por uma UI de console, por exemplo, sem alterar as regras de negócios.

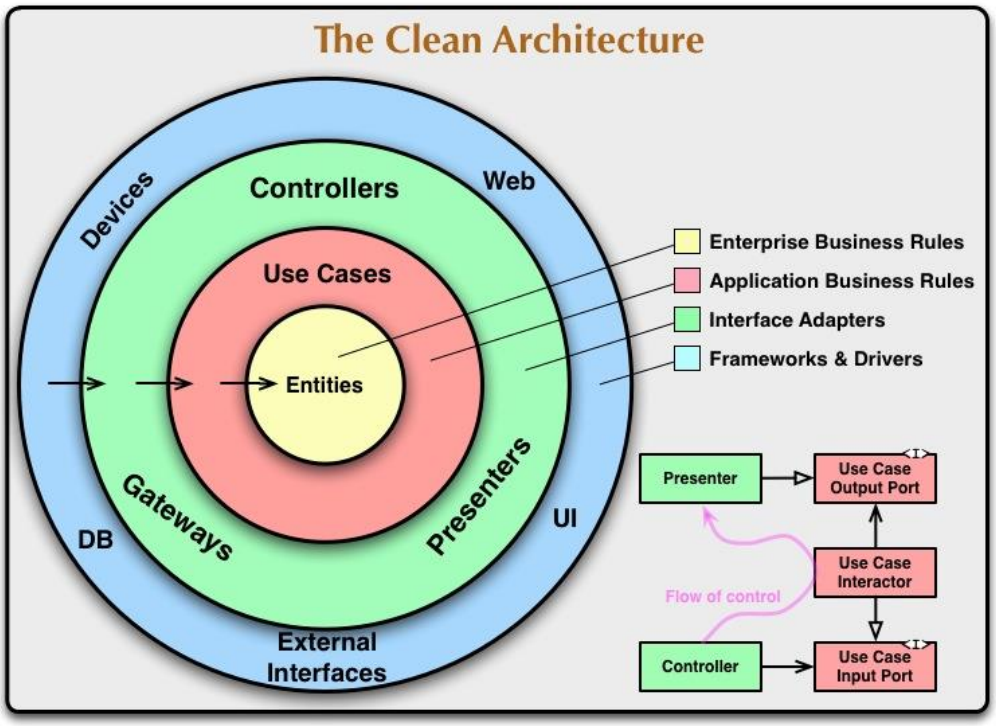
- **Independente de banco de dados:** Você pode trocar Oracle ou SQL Server por Mongo, BigTable, CouchDB ou qualquer outra coisa. Suas regras de negócio não estão vinculadas ao bando de dados.
- **Independente de qualquer agência externa:** Na verdade, suas regras de negócios simplesmente não sabem nada sobre o mundo exterior.

Segundo Martin (2012), a arquitetura limpa pode ser dividida nas seguintes camadas, são elas:

- **Entidades (Entities):** As entidades encapsulam as regras de negócio de toda a empresa. Uma entidade pode ser um objeto com métodos ou pode ser um conjunto de funções e estruturas de dados.
- **Casos de Uso (Use Cases):** O Software nesta camada contém regras de negócios específicas do aplicativo. Ele encapsula e implementa todos os casos de uso do sistema. Esses casos de uso orquestram o fluxo de dados de e para as entidades e direcionam essas entidades para usar suas regras de negócios em toda a empresa para atingir os objetivos do caso de uso.
- **Adaptadores de Interface (Interface Adapters):** O software nesta camada é um conjunto de adaptadores que convertem os dados do formato mais conveniente para os casos de uso e entidades, para o formato mais conveniente para alguma agência externa, como o Banco de Dados ou a Web.
- **Estruturas e Drivers (Frameworks e Drivers):** A camada mais externa geralmente é composta de estruturas e ferramentas como o banco de dados, a estrutura da web, etc.

A figura abaixo ilustra as camadas da arquitetura limpa (ver Figura 30).

Figura 30 - Clean Architecture



Fonte: Martin (2012)

6.8.1.8 Utilização do *Clean Architecture* no back-end

A primeira camada do clean architecture é a Entidade, para exemplificar a utilização da metodologia no back-end, tomei como exemplo a entidade *User*, onde ela é uma classe que contém toda a regra de negócio referente a um usuário, um usuário possui um nome, endereço, informações de contato etc. A figura abaixo ilustra a entidade usuário no back-end (ver Figura 31).

Figura 31- Clean Architecture - Entidade

```
import { IUserDTO } from "../models/UserDTO";

class User {
  firstName: string;
  lastName: string;
  avatar: string;
  email: string;
  whatsapp: string;
  password: string;
  city: string;
  state: string;
  latitude: number;
  longitude: number;

  constructor(user: IUserDTO) {
    this.firstName = user.firstName;
    this.lastName = user.lastName;
    this.avatar = user.avatar;
    this.email = user.email;
    this.whatsapp = user.whatsapp;
    this.password = user.password;
    this.city = user.city;
    this.state = user.state;
    this.latitude = user.latitude;
    this.longitude = user.longitude;
  }
}

export { User };
```

Fonte: Elaborado pelo autor (2023)

A segunda camada é a camada de Aplicação das regras de negócio, para ilustrar, tomei como exemplo o caso de uso referente a criação do usuário, onde a classe *CreateUserUseCase* possui o método *execute* que recebe os dados do usuário a ser criado e verifica se o usuário já existe, caso exista, é lançada uma exceção, caso contrário, o usuário é registrado. A figura abaixo ilustra o caso de uso para criação de usuário (ver Figura 32).

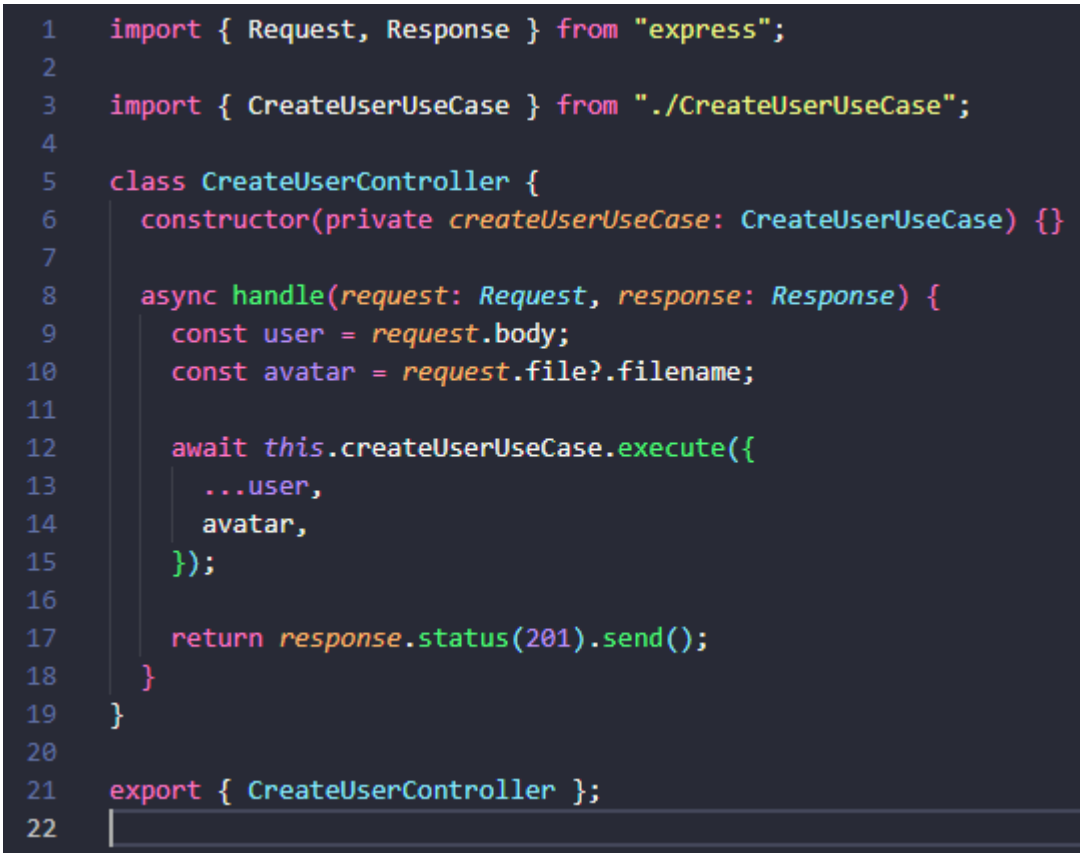
Figura 32 - Clean Architecture - Caso de Uso

```
import { AppError } from "../../errors/AppError";
import { IUserDTO } from "../../models/UserDTO";
import { IUserRepository } from "../../repositories/IUserRepository";

class CreateUserUseCase {
  constructor(private userRepository: IUserRepository) {}

  async execute(user: IUserDTO) {
    const userAlreadyExists = await this.userRepository.findByEmail(user.email);

    if (userAlreadyExists) {
      throw new AppError("User already exists.");
    }

    await this.userRepository.create(user);
  }
}

export { CreateUserUseCase };

```

Fonte: Elaborado pelo autor (2023)

A terceira camada é a camada de Adaptadores de interface, para ilustrar, tomei como exemplo o *controller* responsável pela chamada ao caso de uso de criação do usuário, onde a classe *CreateUserController* possui um método *handle* que recebe os dados do usuário a ser criado e executa o caso de uso responsável pela criação do usuário, em seguida responde com o status 201. A figura abaixo ilustra o controller de criação de usuário (ver Figura 33).

Figura 33 - Clean Architecture - Adaptadores de interface



```
import { Request, Response } from "express";

import { CreateUserUseCase } from "./CreateUserUseCase";

class CreateUserController {
  constructor(private createUserUseCase: CreateUserUseCase) {}

  async handle(request: Request, response: Response) {
    const user = request.body;
    const avatar = request.file?.filename;

    await this.createUserUseCase.execute({
      ...user,
      avatar,
    });

    return response.status(201).send();
  }
}

export { CreateUserController };
```

Fonte: Elaborado pelo autor (2023)

A última camada é a camada de Estruturas e drivers, para ilustrar, tomei como exemplo a classe *UserRepository* que possui toda a responsabilidade da comunicação com o banco de dados. Quando o caso de uso de criação de usuário é executado, o método *create* da classse *UserRepository* é executado, ocorrendo a inserção do usuário no banco de dados. A figura abaixo ilustra o repositório do usuário (ver Figura 34).

Figura 34- Clean Architecture - Estruturas e Drivers

```typescript
import knex from "../../database/connection";
import { IUserDTO } from "../../models/UserDTO";
import { IUserRepository } from "../IUserRepository";

class UserRepository implements IUserRepository {
  async findById(id: number): Promise<IUserDTO | null> {
    const users = await knex<IUserDTO>("user").where({ id }).select("*");

    return users[0] || null;
  }

  async findByEmail(email: string): Promise<IUserDTO | null> {
    const users = await knex<IUserDTO>("user").where({ email }).select("*");

    return users[0] || null;
  }

  async create(user: IUserDTO): Promise<void> {
    await knex<IUserDTO>("user").insert(user);
  }

  async update(user: IUserDTO): Promise<void> {
    await knex<IUserDTO>("user").update(user).where({ id: user.id });
  }
}

export { UserRepository };

```

Fonte: Elaborado pelo autor (2023).

6.8.1.9 Front-end

Para Wales (2020), front-end é a parte com a qual os usuários interagem. Tudo o que se vê quando se está navegando pela internet ou acessando uma aplicação, de fontes e cores a menus suspensos, é tudo uma combinação de HTML, CSS e JavaScript. Segundo Gallinelli (2021), as principais linguagens front-end são HTML, CSS e JavaScript. É com essas linguagens que as páginas web e aplicações ganham vida e permitem uma melhor experiência de usuário. É através do front-end que o usuário interage com a aplicação e insere entradas que o back-end irá processar (ALMEIDA, 2018).

6.8.2.1 React Native

Para o desenvolvimento do app foi utilizado o framework React Native. O React Native é um framework de desenvolvimento de aplicativos móveis criado em 2015 pela equipe do Facebook e que possibilita a criação de aplicativos para Android e iOS utilizando a linguagem JavaScript. O RN é baseado no React, uma biblioteca JavaScript para a construção de interface de usuário.

Diferente do React que manipula a DOM com Virtual DOM, o React Native realiza uma comunicação através de uma *bridge* com chamadas assíncronas para a plataforma móvel, chamando as APIs de ferramentas nativas (BODUCH, 2017). Disponibilizando assim, um conjunto de componentes nativos como *View*, *Text*, *Image*, o acesso a câmera do dispositivo, GPS, entre outros, sem depender da utilização de WebViews (MASIELLO; FRIEDMANN, 2017).

Com o React Native, os desenvolvedores tem a possibilidade de criar componentes customizados que podem ser reutilizados em qualquer tela da aplicação. A figura abaixo apresenta um trecho de um código do aplicativo Adotaí escrito no React Native. O código apresentado abaixo é o componente Box, que recebe um contéudo qualquer como um texto ou uma imagem e exibe esse contéudo dentro do box (ver Figura 35).

Figura 35- Trecho de código - Componente box

```
1  import { BoxStyled } from './styles';
2  import { BoxStyledProps } from './types';
3
4  export default function Box({ children }: BoxStyledProps) {
5      return (
6          <BoxStyled>{children}</BoxStyled>
7      );
8  }
```

Fonte: Elaborado pelo autor (2023)

6.8.2.2 Expo

Para a execução do aplicativo em ambiente de desenvolvimento foi utilizado o Expo. O Expo é uma ferramenta que permite desenvolver, testar e implantar aplicativos Android, iOS e Web com React Native. Essa ferramenta fornece acesso aos recursos do dispositivo como contatos, câmera, GPS etc, sendo possível acessá-los apenas instalando seus respectivos pacotes (EXPO, 2023).

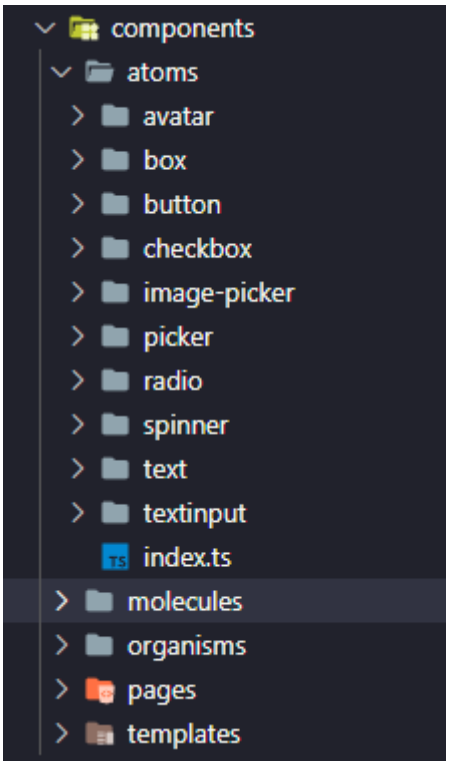
Um recurso interessante do Expo é a possibilidade de executar o aplicativo em tempo real no dispositivo físico do desenvolvedor, isso se dá através do aplicativo *Expo Go* disponível na *Play Store* e *Apple Store*. Com o *Expo Go*, é possível escanear o código *QR Code* gerado na executação do projeto React Native, e desse modo, poder acompanhar visualmente no dispositivo físico cada alteração realizada no código fonte.

6.8.2.3 Utilização do *Atomic Design* no front-end

A metodologia do *Atomic Design* foi utilizada no aplicativo como forma de separar e organizar os componentes por hierarquia, onde foi criado no projeto a pasta *components*, contendo todos os componentes e telas da aplicação. As pastas foram criadas com a mesma nomenclatura da metodologia, sendo elas os: átomos, moléculas, organismos, páginas e templates.

A pasta *atoms* contém os componentes de átomos da aplicação como o avatar, button, box, text, textinput etc (ver Figura 36).

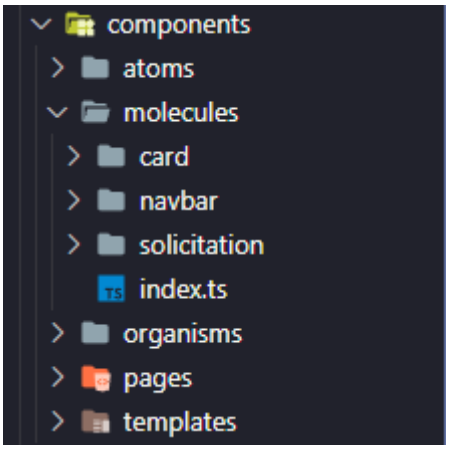
Figura 36 - Organização das pastas - Átomos



Fonte: Elaborado pelo autor (2023)

A pasta *molecules* contém os componentes de moléculas da aplicação como card, navbar e solicitation (ver Figura 37).

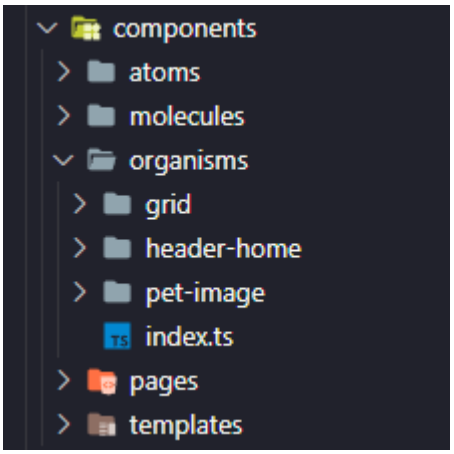
Figura 37 - Organização das pastas - Moléculas



Fonte: Elaborado pelo autor (2023)

A pasta *organisms* contém os componentes de organismos da aplicação como grid, header-home e pet-image (ver Figura 38).

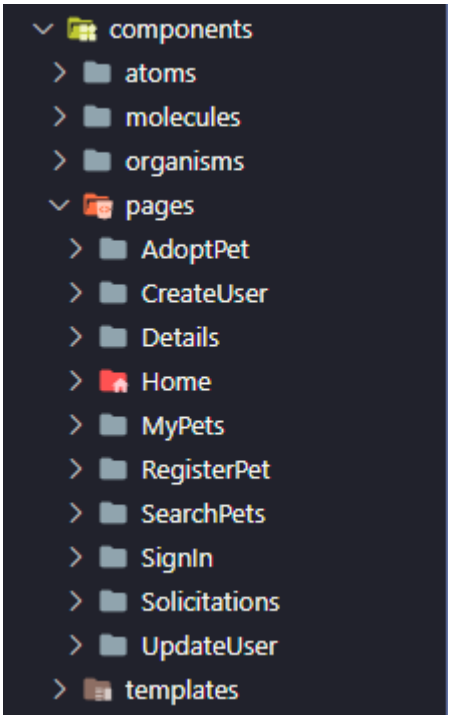
Figura 38- Organização das pastas - Organismos



Fonte: Elaborado pelo autor (2023)

A pasta *pages* contém as telas da aplicação como a tela de cadastro de usuário, tela de login, tela de cadastro de pets, tela de busca de pet etc (ver Figura 39).

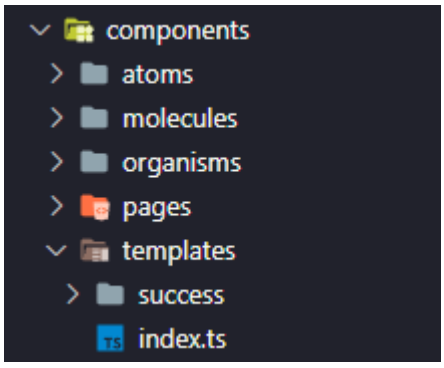
Figura 39 - Organização das pastas - Páginas



Fonte: Elaborado pelo autor (2023)

A pasta *templates* contém telas que são reutilizadas por toda a aplicação, onde na figura abaixo temos a tela de sucesso que é exibida quando usuário conclui alguma operação com êxito (ver Figura 40).

Figura 40 - Organização das pastas - Templates

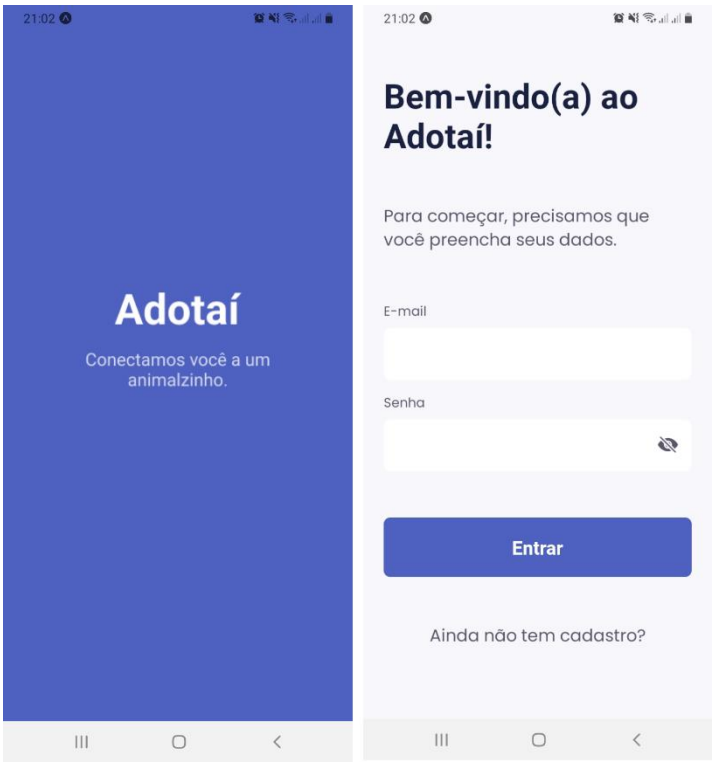


Fonte: Elaborado pelo autor (2023)

6.8.2.4 Apresentação das telas

As figuras abaixo apresentam a tela de *splash* e *login*. A primeira tela (*splash*) é exibida para o usuário quando o aplicativo está carregando. Já a segunda tela (*login*) é responsável pela autenticação do usuário, onde é possível o usuário informar os dados e realizar o login na aplicação caso tenha cadastro, caso não tenha, poderá navegar para a tela de cadastro (ver Figura 41).

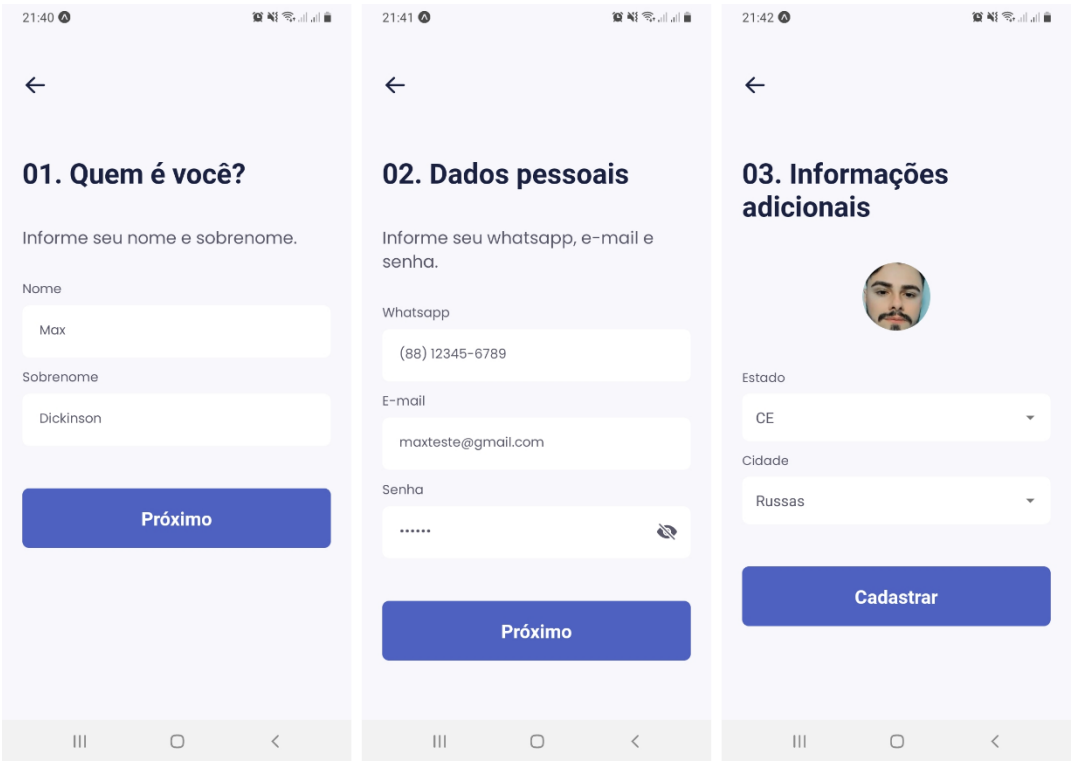
Figura 41- Aplicativo - Tela de splash e login



Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam as telas de cadastro. Na primeira tela o usuário poderá informar seu nome e sobrenome e avançar para a próxima etapa do cadastro. Já na segunda tela, o usuário poderá informar dados de contato como whatsapp, e-mail e uma senha, e assim, poderá avançar para a última etapa. Na última tela, o usuário poderá adicionar sua foto de perfil, estado e cidade e concluir o cadastro (ver Figura 42).

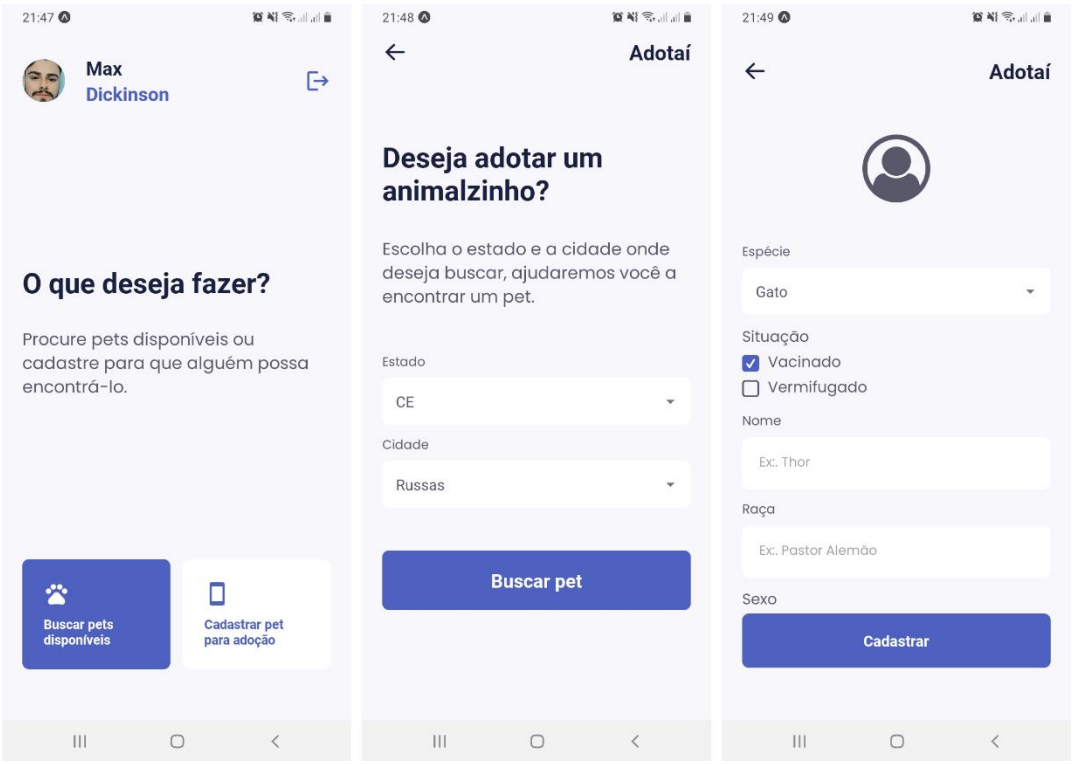
Figura 42- Aplicativo- Telas de cadastro



Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam a tela de home, adoção e cadastro de pet. A primeira tela é a home, essa é a tela para onde o usuário é redirecionado quando faz o login na aplicação. A segunda tela, é onde o usuário irá selecionar o estado e a cidade onde deseja buscar um pet para adoção. A terceira tela é onde o usuário irá cadastrar um pet, informando a sua espécie, situação, nome, raça, sexo, idade e peso (ver Figura 43).

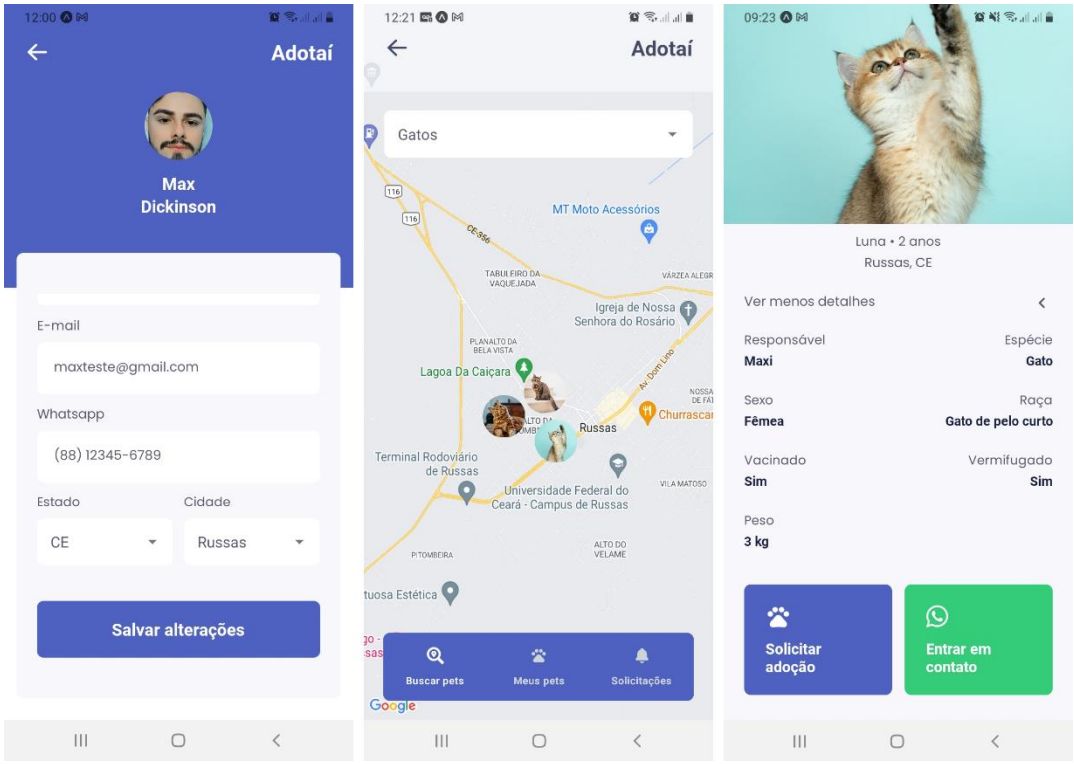
Figura 43- Aplicativo - Tela de home, adoção e cadastro de pet



Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam a tela de edição do usuário, mapa e detalhes do pet. A primeira imagem é a tela onde o usuário poderá atualizar seus dados como nome, sobrenome, e-mail, whatsapp, estado e cidade. A segunda tela é a tela de mapa, essa tela é onde são listados os animais disponíveis para adoção, o usuário seleciona a espécie que deseja buscar e o aplicativo exibe todos os animais daquela espécie com base no estado e cidade selecionado anteriormente. Já a terceira imagem é a tela de detalhes do pet, o usuário acessa ela quando seleciona algum pet no mapa, onde nessa tela, ele pode solicitar adoção ou entrar em contato com o dono do pet (ver Figura 44).

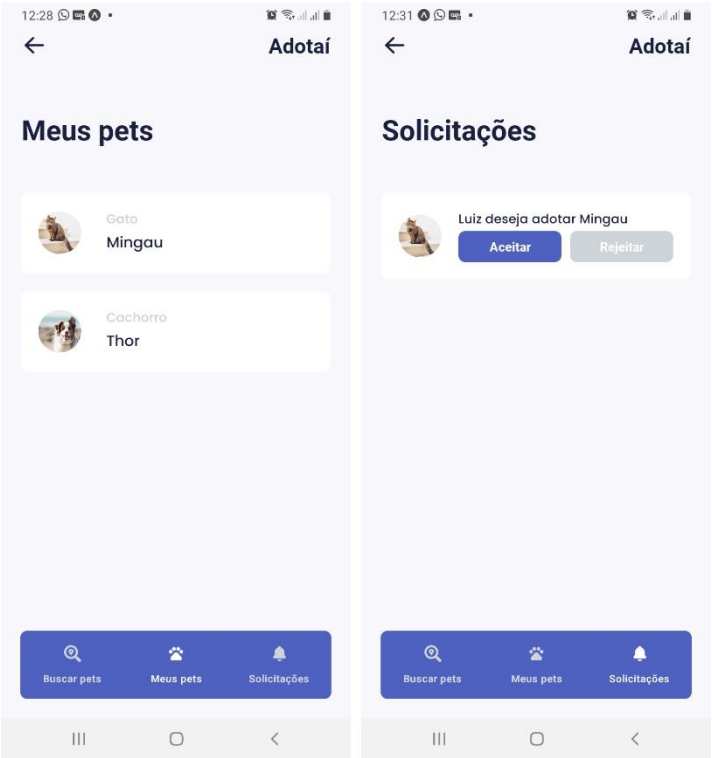
Figura 44 - Aplicativo - Tela de edição do usuário, mapa e detalhes do pet



Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam as telas meus pets e solicitações. A primeira tela é onde lista os animais cadastrados pelo usuário. A segunda tela é onde lista as solicitações de adoção enviadas por outros usuários. Ainda nessa tela o usuário dono do pet pode aceitar ou rejeitar a solicitação, caso aceite, o pet passa a ser de responsabilidade do usuário que solicitou, caso contrário, o pet continua com o seu respectivo tutor (ver Figura 45).

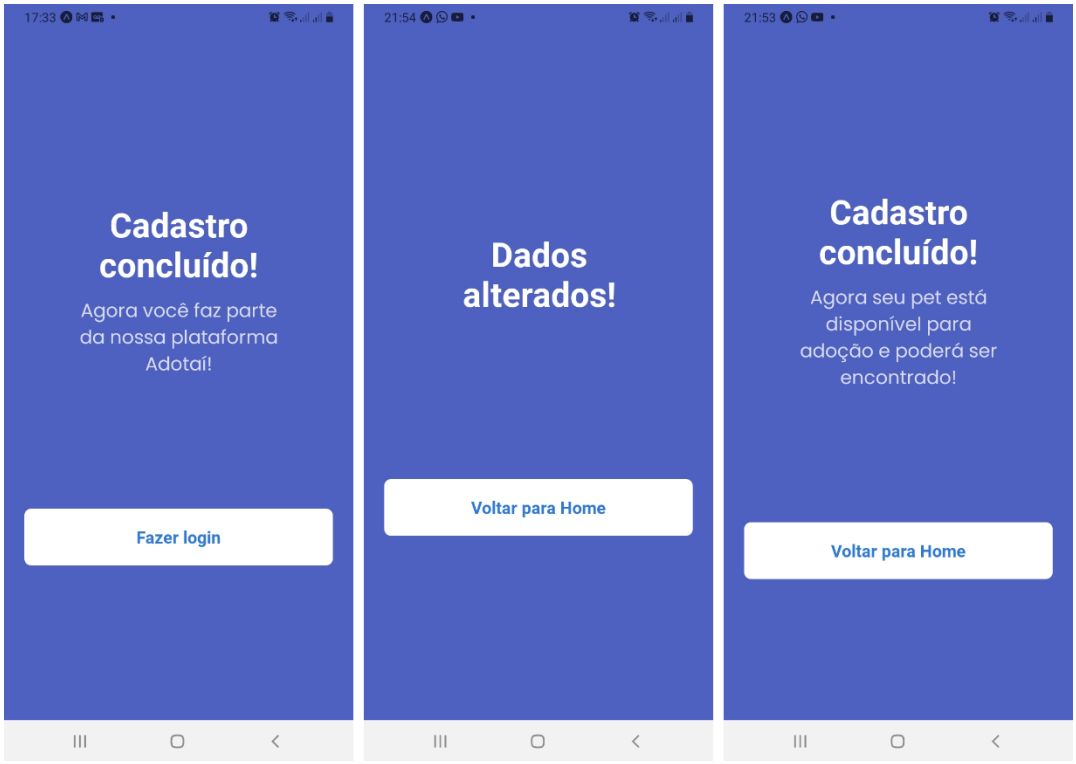
Figura 45- Aplicativo - Tela de meus pets e solicitações



Fonte: Elaborado pelo autor (2023)

As figuras abaixo apresentam as telas de sucesso. A primeira tela é exibida quando o usuário realiza o seu cadastro com sucesso na plataforma. A segunda tela é apresentada após o usuário atualizar os seus dados. Já a terceira tela é exibida quando o usuário cadastra um pet na plataforma (ver Figura 46).

Figura 46 - Aplicativo - Telas de sucesso



Fonte: Elaborado pelo autor (2023)

6.9 Teste de usabilidade

Nesta etapa, foi realizado um teste de usabilidade do aplicativo desenvolvido. Participaram do teste de forma presencial 8 alunos membros do projeto PAE (Projeto de Apoio ao Ensino para à comunidade acadêmica e russana), PROGETE (Programa de Ensino e Troca de Experiências) e uma professora do curso de Engenharia de Software, todos da Universidade Federal do Ceará – Campus Russas.

Inicialmente, houve uma apresentação e explicação de como seria realizado o teste, em seguida, foi selecionado um participante por vez e entregue 3 cenários para que fossem lidos e executados. Cada participante do teste precisou executar todos os cenários, onde cada um dos cenários possuía uma descrição e uma tarefa a ser realizada.

No cenário 1, um usuário estava procurando por aplicativos responsáveis pela doação/adoção de animais e conheceu o Adotaí. No cenário 2, um usuário estava se sentindo muito sozinho(a) ultimamente, e desejava adotar um cãozinho disponível na cidade dele(a). No cenário 3, um usuário estava caminhando pela rua e encontrou um gatinho abandonado, e decidiu cadastrá-lo no aplicativo Adotaí.

Os participantes foram nomeados de P1 a P9, onde no quadro abaixo são descritas as atividades que foram realizadas, sendo pontuadas por 1 as tarefas realizadas sem ajuda externa, 2 as tarefas realizadas com ajuda externa e 3 as tarefas realizadas sem ajuda, mas com dificuldades (ver Quadro 16).

Quadro 16 - Tarefas do teste de usabilidade

Tarefas	**Participantes**									**Médias**		
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**1 = SEM AJUDA**	**2 = COM AJUDA**	**3 = SEM AJUDA COM DIFICULDADES**
1. Realizar o cadastro de usuário.	1	1	1	1	1	1	1	1	1	100%	0%	0%
2. Realizar a busca de um cãozinho e o adotar.	1	1	1	1	1	1	1	1	1	100%	0%	0%
3. Realizar o cadastro do gatinho no aplicativo.	1	1	1	1	3	1	3	3	1	66,66%	0%	33,33%
MÉDIA										88,88%	0%	11%

Fonte: Elaborado pelo autor (2023)

Com base na análise das tarefas realizadas no teste, foi possível observar que 88,88% dos participantes realizaram as tarefas sem ajuda, enquanto que 0% responderam com ajuda, e 11% responderam sem ajuda mas com dificuldades.

Após a realização das tarefas, foi enviado um formulário elaborado no *Google Forms* aos participantes, solicitando que respondessem o questionário SUS – *System Usability Scale.* O formulário possui 10 questões objetivas com as seguintes afirmações:

1. Eu acho que gostaria de usar esse aplicativo com frequência;
2. Eu acho o aplicativo fácil de usar;
3. Eu acho o aplicativo difícil de usar;
4. Eu acho que precisaria da ajuda de uma pessoa com conhecimentos técnicos para usar o aplicativo;
5. Eu consigo cadastrar um usuário no aplicativo;
6. Eu consigo cadastrar um pet no aplicativo;
7. Eu consigo encontrar os pets disponíveis para adoção no aplicativo;
8. Eu achei o aplicativo atrapalhado de usar;
9. Eu me senti confiante ao usar o aplicativo;
10. Eu precisei aprender várias coisas novas antes de conseguir usar o aplicativo;

Para as alternativas das questões, foi utilizado a escala *Likert*, que é uma escala de resposta muito utilizada em questionários para medir o grau de concordância ou não com uma afirmação. No questionário aplicado, foi utilizado a escala *Likert* de 05 pontos com as seguintes alternativas:

- Discordo totalmente
- Discordo
- Indiferente
- Concordo
- Concordo totalmente

Segundo Thomas (2022), para calcular a pontuação das respostas referente as afirmativas que foram avaliadas pelos participantes, deve ser realizado o seguinte cálculo:

- Para as questões ímpares (1,3,5,7 e 9), deve-se subtrair 1 da resposta do participante.
- Para as questões pares (2,4,6,8 e 10), deve-se subtrair o valor da resposta do participante de 5.
- Por fim, realize o somatório dos valores do resultado de cada questão e multiplique por 2,5.

O quadro abaixo apresenta as respostas dos participantes para cada afirmação e a média obtida (ver Quadro 17).

Quadro 17 - Questionário - Teste de usabilidade

Número	Afirmações	P1	P2	P3	P4	P5	P6	P7	P8	P9	AVG
1	Eu acho que gostaria de usar esse aplicativo com frequência.	3	4	3	4	3	5	3	1	5	3,4
2	Eu acho o aplicativo fácil de usar.	4	5	4	4	5	1	4	4	5	4
3	Eu acho o aplicativo difícil de usar.	2	2	4	1	2	2	2	1	1	1,8
4	Eu acho que precisaria da ajuda de uma pessoa com conhecimentos técnicos para usar o aplicativo	1	1	1	1	1	4	1	1	1	1,3
5	Eu consigo cadastrar um usuário no aplicativo.	5	5	5	5	5	4	5	4	5	4,7
6	Eu consigo cadastrar um pet no aplicativo.	5	4	5	5	5	5	5	5	4	4,7
7	Eu consigo encontrar os pets disponíveis para adoção no aplicativo.	4	5	5	5	4	5	5	5	5	4,7
8	Eu achei o aplicativo atrapalhado de usar.	1	4	1	1	1	4	2	2	2	2
9	Eu me senti confiante ao usar o aplicativo.	4	4	4	5	4	5	4	3	4	4,1

10	Eu precisei aprender várias coisas novas antes de conseguir usar o aplicativo.	1	1	4	1	1	1	1	1	2	1,4
	MÉDIA SUS	65	62,5	65	70	62,5	65	65	52,5	65	63,6

Fonte: Elaborado pelo autor (2023)

O questionário SUS é um método que pode ser aplicado ao final de um teste de usabilidade com o intuito de averiguar o nível de usabilidade de um sistema. O método foi criado em 1986 por John Brooke, e pode ser utilizado para avaliar produtos, serviços, hardware, software, websites, aplicações e qualquer outro tipo de interface.

Com base nas respostas dos participantes, o SUS foi calculado e obteve a pontuação média de 63.6 pontos. Para Barboza (2019), uma pontuação entre 60-70 indica que o nível de usabilidade da aplicação é aceitável.

7 CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho buscou soluções para contribuir nos principais problemas apresentados com o abandono de animais nas ruas. Para isso, objetivou-se o desenvolvimento de um aplicativo para doação/adoção de animais utilizando a metodologia do *Atomic Design* para criação dos componentes hierárquicos e reutilizáveis. A elaboração do trabalho ocorreu a partir de uma pesquisa bibliográfica, *benchmarking*, documentação do requisitos, criação dos componentes, criação das telas, aplicação de questionário, desenvolvimento e teste de usabilidade.

A pesquisa bibliográfica possibilitou a descoberta de trabalhos semelhantes na literatura, onde a partir dessa etapa e atavés do benchmarking, foram identificados os principais problemas e funcionalidades num aplicativo de adoção de animais. Essas funcionalidades foram então registradas possibilitando a documentação dos requisitos contendo os seus requisitos funcionais, não-funcionais e as regras de negócio. Foi então a partir dessa documentação, que foram criados com base na metodologia do *Atomic Design* os componentes e as telas da aplicação, telas estas, que proporcionaram a elaboração de um questionário onde foi aplicado ao público-alvo como forma de avaliar os protótipos desenvolvidos.

Em relação aos resultados do questionário de validação do protótipo, a proposta do aplicativo obteve uma média de 88,8%, a facilidade de utilização obteve uma média 94,4% enquanto o design obteve uma média de 61,1%. A partir das telas elaboradas e validadas, possibilitou-se o desenvolvimento do aplicativo utilizando o *framework React Native*. O teste de usabilidade do aplicativo foi conduzido de forma presencial com 8 alunos membros do projeto PAE (Projeto de Apoio ao Ensino para à comunidade acadêmica e russana), PROGETE (Programa de Ensino e Troca de Experiências) e uma professora da Universidade Federal do Ceará – Campus Russas, e como resultado, obteve um nível médio de usabilidade de 63,6 pontos. Nesse cenário, foi possível constatar que o aplicativo atendeu bem as necessidades, mas que ainda necessita de melhorias a serem realizadas no design e na usabilidade.

A principal contribuição desta pesquisa foi propor um aplicativo para adoção de animais, visando a redução do número de animais abandonados nas ruas, proliferação de doenças e acidentes de trânsito, além da utilização de uma metodologia para a criação e organização de componentes reutilizáveis tanto no protótipo, como na arquitetura de pastas do aplicativo. A utilização do *Atomic Design* no *app* propôs um alto ganho de produtividade, pois uma vez criado um componente, ele poderia ser facilmente reutilizado em qualquer lugar da aplicação.

Durante a realização da pesquisa, foi possível constatar que o aplicativo possui uma série de vantagens, mas que necessita de melhorias para atender a todas as necessidades dos usuários. Como trabalho futuro, pretende-se realizar as melhorias identificadas no teste de usabilidade, utilizar *Design Pattern*, e adicionar mais funcionalidades como a criação das telas para recuperação de senha, a utilização de notificações e a possibilidade de criar dentro da plataforma anúncios de animais desaparecidos.

REFERÊNCIAS

ABINPET. **Abinpet**, 2022. Mercado PET BRASIL 2022. Disponível em: https://abinpet.org.br/dados-de-mercado/. Acesso em: 28 Set. 2022.

AMARAL, R. A. do; NERIS, V. P. de A. Análise comparativa entre frameworks de front-end para aplicações web ricas visando reaproveitamento do back-end. **Revista TIS**, São Carlos, v. 4, n. 1, p. 88-96, Abr, 2015.

ALMEIDA, Francisco Emerson Vieira De. **Um comparativo entre frameworks javascript para desenvolvimento de aplicações front-end.** 2018. 43 f. TCC (Graduação) – Curso de Engenharia de Software, Universidade Federal do Ceará, Quixadá, 2018.

BARBOZA, Anderson. **Medium,** 2019. Medindo a usabilidade do seu produto com System Usability Scale (SUS). Disponível em: https://medium.com/design-contaazul/medindo-a-usabilidade-do-seu-produto-com-system-usability-scale-sus-3956612d9229. Acesso em: 27 Jun. 2023.

BARRO, Bruna, B. **Hostinger,** 2022. O que é uma API RESTful e porque isso importa. Disponível em: https://www.hostinger.com.br/tutoriais/api-restful Acesso em: 20 Mar. 2023.

BODUCH, Adam. **React and React Native.** Packt Publishing, isbn: 9781786465658, 2017.

BONFITTO, Isabella Janaína Tonolli Rosa. **Ilustrações em empty states.** 2019. 65 f. TCC (Graduação) - Curso de Tecnologia em Design em Gráfico, Universidade Tecnológica Federal do Paraná. Curitiba, 2019.

BUDIU, Raluca. Mobile: Native Apps, Web Apps, and Hybrid Apps. **Nielsen Norman Group,** 2016. Disponível em: https://www.nngroup.com/articles/mobile-native-apps/. Acesso em: 01 Set. 2022.

EL-KASSAS, Wafaa S. Taxonomy of Cross-Platform Mobile Applications Development Approaches. **Ain Shams Engineering Journal,** Egito, v. 8, n. 2, p. 163-190, Jun, 2017. Disponível em: https://www.sciencedirect.com/science/article/pii/S2090447915001276. Acesso em: 01 Set. 2022.

EXPO. **Reference**, 2023. Disponível em: https://docs.expo.dev/versions/latest/. Acesso em: 21 Jun. 2023.

FIELDING, Roy Thomas. **Architectural Styles and the Design of Network-based Software Architectures.** 2000. 162 f. Tese (Doutorado) – University of California. Irvine, 2000.

FROST, Brad. **Atomic Design Methodology,** 2016. Disponível em: https://atomicdesign.bradfrost.com/chapter-2/. Acesso em: 02 Nov. 2022.

GALLINELLI, Nicholas. **Flatiron School**, 2021. Front end vs Back end development. Disponível em: https://flatironschool.com/blog/front-end-vs-back-end-development/ Acesso em: 10 Mar. 2023.

GUDWIN, Ricardo. R. Diagrama de casos de uso. **Unicamp,** 2010. Disponível em: https://www.dca.fee.unicamp.br/~gudwin/ftp/ea976/UseCases Acesso em: 17 Mai. 2023.

GUIMARÃES, Felipe Pires. **AdoçãoPet CG: Um aplicativo para simplificar o processo de adoção de animais em CG.** 2021. 14 f. TCC (Graduação) - Curso de Ciência da Computação, Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande. Campina Grande, 2021.

GUPTA, Lokesh. **RESTfulAPI,** 2022. What is REST. Disponível em: https://restfulapi.net. Acesso em: 14 Abr. 2023.

INSTITUTO PET BRASIL. **Instituto Pet Brasil**, 2022. Número de animais de estimação em situação de vulnerabilidade mais do que dobra em dois anos, aponta pesquisa do IPB. Disponível em: http://institutopetbrasil.com/fique-por-dentro/numero-de-animais-de-estimacao-em-situacao-de-vulnerabilidade-mais-do-que-dobra-em-dois-anos-aponta-pesquisa-do-ipb/. Acesso em: 02 Set. 2022.

JOHNSON, Ralph E.; FOOTE, Brian; Designing Reusable Classes. **Journal of Object-Oriented Programming**, Urbana–Champaign, v. 1, n. 2, p. 22-35, Jun, 1988. Disponível em: https://www.academia.edu/download/68326912/Designing_Reusable_Classes20210726-5212-mwdg3x.pdf. Acesso em: 09 Out. 2022.

JORDÃO, Henrique Capelato. **MyDog: Um aplicativo mobile para gerenciamento e cuidado de cachorros.** 2021. 63 f. TCC (Graduação) - Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2021.

JUNIOR, Adriano Benatti. **Software para gestão de adoção de animais.** 2019. 98 f. TCC (Graduação) - Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, Faculdade de Tecnologia de Americana, Americana, 2019.

MARTIN, R. C. **The Clean Architecture,** 2012. Disponível em: https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html. Acesso em: 16 Maio. 2023.

MASIELLO, E. FRIEDMANN, J. **Mastering React Native.** Packt Publishing, isbn: 9781785885785, 2017.

MENDES, Mariana Ribeiro; GARBAZZA, Itagildo Edmar; TERRA, Daniela Costa; **Desenvolvimento híbrido versus desenvolvimento nativo de aplicativos móveis.** Bambuí: Semana de Ciência e Tecnologia do IFMG, 2014. p. 1-5. Disponível em: https://www.bambui.ifmg.edu.br/jornada_cientifica/2014/resumos/Info/Desenvolvimento%20híbrido%20versus%20desenvolvimento%20nativo%20de%20apl.pdf. Acesso em: 17 Set. 2022.

MENDONÇA, V. R. L. de; BITTAR T. J; DIAS, M. de. S; **Um estudo dos Sistemas Operacionais Android e iOS para o desenvolvimento de aplicativos.** Goiás: Enacomp, 2011. p.1-8. Disponível em: https://www.enacomp.com.br/2011/anais/trabalhos-aprovados/pdf/enacomp2011_submission_54.pdf. Acesso em: 04 Out. 2022.

MYMOB. **Mymob**, 2020. A história do Android: Conheça a evolução do sistema móvel mais usado no mundo. Disponível em: https://mymob.com.br/blog/historia-do-android.html#:~:text=Foi%20nessa%20época%20que%20uma,era%20o%20seu%20site%20pessoal. Acesso em: 15 Set. 2022.

NOGUEIRA, D. L. **Ferramentas automatizadas para apoio ao projeto estruturado: uma aplicação do diagrama de entidade-relacionamento.** 1988. 336 f. Tese (Doutorado) Ciências em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1988.

NOLETO, Cairo. **Betrybe**, 2022. API REST: O que é como montar uma API sem complicação? Disponível em: https://blog.betrybe.com/desenvolvimento-web/api-rest-tudo-sobre/ Acesso em: 15 Mai. 2023.

OBJECTIVE. **Objective**, 2020. Desenvolvimento de aplicativos: principais etapas e desafios. Disponível em: https://www.objective.com.br/insights/fases-e-desafios-desenvolvimento-de-aplicativos/. Acesso em: 20 Set. 2022.

OPEN HANDSET ALLIANCE. **Open Handset Alliance**, 2021. What would it take to build a better mobile phone? Disponível em: https://www.openhandsetalliance.com. Acesso em: 13 Set. 2022.

OURIQUES, Joana Reckziegel. **Bem estar animal: um abrigo para cães e gatos vítimas de maus-tratos e abandono em Florianópolis.** 2018. 46 f. TCC (Graduação) - Curso de Arquitetura e Urbanismo, Universidade do Sul de Santa Catarina, Florianópolis, 2018.

PAZ, Marina Medeiros da. **Metamorphosis: Um design system reutilizável e adaptável à qualquer projeto mobile.** 2019. 112 f. TCC (Graduação) - Curso de Design, Universidade Federal de Santa Catarina, Florianópolis, 2019.

PINHEIRO, Juliana Silva. **Análise do desenvolvimento de aplicativos mobile nativos e multiplataforma.** 2020. 185 f. TCC (Graduação) - Ciências da Computação, Universidade Federal de Santa Catarina, Florianópolis, 2020.

PRADO, G.F. do; SILVA, A. F; MEDEIROS, F. N. de; REZENDE, G. M. S. de; MARTINS, L. G. S; OLIVEIRA, M. C. de; FERREIRA, N. de. S; VENANCIO, R. D. O. **Não abandone, adote: em defesa da adoção responsável.** Uberlândia: Intercom, 2015. p. 1-8. Disponível em: https://www.researchgate.net/profile/Rafael-Duarte-Oliveira-Venancio/publication/316790067_Nao_abandone_adote_em_defesa_da_adocao_responsavel/links/5911d7beaca27200fe3b5eca/Nao-abandone-adote-em-defesa-da-adocao-responsavel.pdf. Acesso em: 05 Nov. 2022.

PULUCENO, Thiago Vieira. **Estudo de caso sobre uma api rest utilizando a abordagem de programação orientada e eventos com a plataforma node.js.** 2012. 74 f. TCC (Graduação) – Sistemas de Informação, Universidade Federal de Santa Catarina, Florianópolis, 2012.

REDHAT. **Red Hat**, 2020. O que é API REST? Disponível em: https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api Acesso em: 27 Mar. 2023.

ROCHA, Pedro Alencar de Sousa Santos; JUNIOR, Sandro Ireno Martins. **Desenvolvimento de aplicativo para o auxílio de adoção de animais utilizando node e react native.** 2021. 87 f. TCC (Graduação) - Curso de Sistemas da Informação, Universidade do Sul de Santa Catarina, Florianópolis, 2021.

ROVEDA, Ugo. **Kenzie**, 2021. O que é back end, para que serve e como aprender em 2021. Disponível em: https://kenzie.com.br/blog/back-end/ Acesso em: 02 Mai. 2023.

SANTANA, Luciano Rocha; OLIVEIRA, Thiago Pires. Guarda responsável e dignidade dos animais. **Revista brasileira de direito animal**, 2006. Disponível em: https://periodicos.ufba.br/index.php/RBDA/article/view/32362/19167. Acesso em: 25 Out. 2022.

SANTIAGO, Felipe Issa. **Atomic Design aplicado no desenvolvimento de produtos.** 52f. TCC (Graduação) – Curso de Ciência da Computação, Pontifícia Universidade Católica de Goiás, 2022.

SCHERER, A.; CUNHA, C. D. O, P. de. O.; LAUREANO, D. B.; ANDRADE, Emanuely V. R. de. A.; FERREIRA, J. de. F.; BRAGHIROLI, N.; SILVA, S. . da.; MENDONÇA, R. C. A importância da adoção de animais no Brasil. **Pubvet**, São Paulo, v. 15, n. 07, p. 208, Jul, 2021. Disponível em: https://pubvet.com.br/artigo/8058/a-importacircncia-da-adoccedilatildeo-de-animais-no-brasil. Acesso em: 25 Ago. 2022.

SEOBILITY. **Seobility,** 2023. Rest API. Disponível em: https://www.seobility.net/en/wiki/REST_API Acesso em: 30 Jun. 2023.

SILVA, Rafael Lopes Pestana da. **Pet adoto: Projeto de interface de aplicativo para adoção de animais.** 2017. 74 f. TCC (Graduação) - Curso de Comunicação Visual Design, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2017.

STATCOUNTER GLOBAL STATS. **Stat Counter**, 2022. Operating System Market Share Worldwide. Disponível em: https://gs.statcounter.com/os-market-share. Acesso em: 02 Out. 2022.

THOMAS, Nathan. **Usability Geek,** 2022. How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website. Disponível em: https://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/ Acesso em: 25 Jun. 2023.

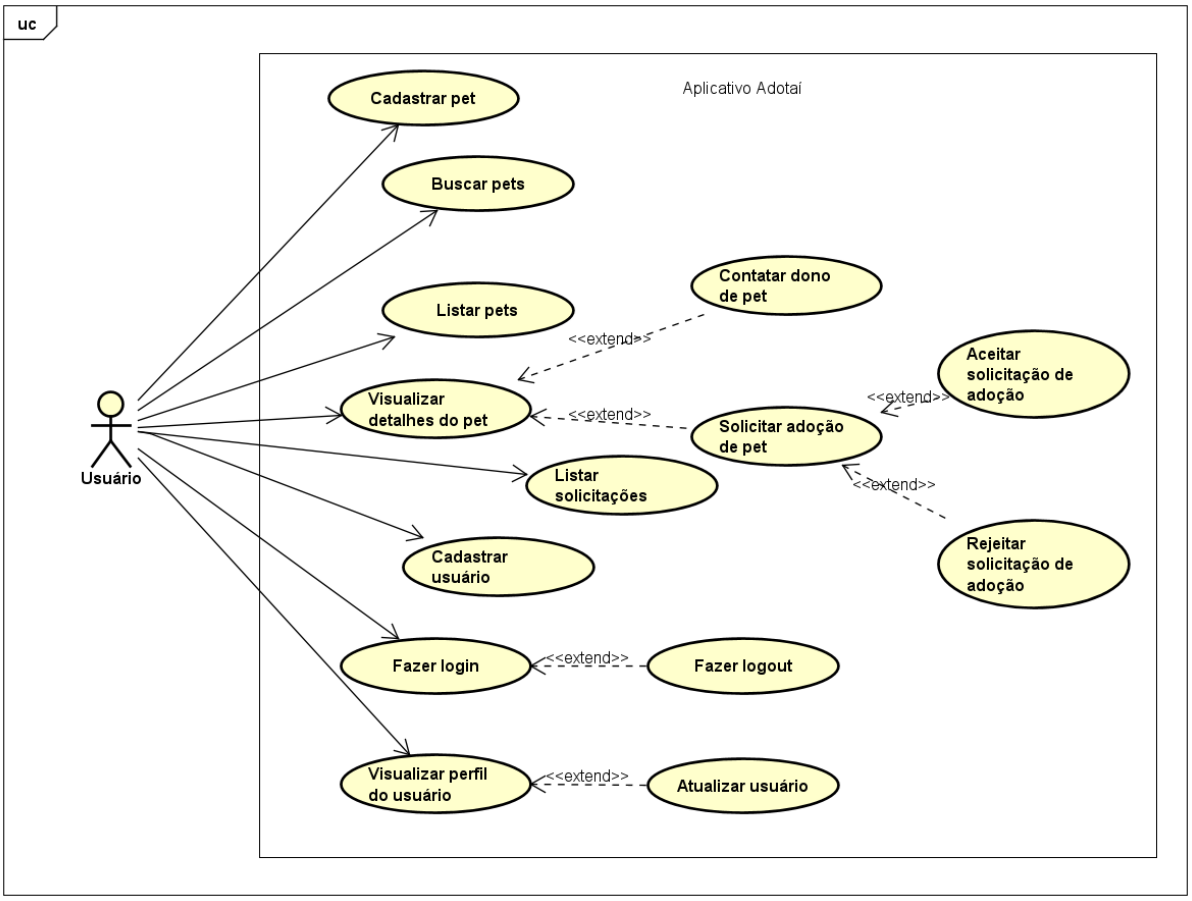
TURINE, Marcelo Augusto Santos; MASIERO, Paulo Cesar. **Especificação de requisitos: uma introdução.** São Carlos: Instituto de Ciências Matemáticas e da Computação, 1996. p. 1-26. Disponível em: http://repositorio.icmc.usp.br/bitstream/handle/RIICMC/6800/Relatório%20Técnico_39_1996.pdf. Acesso em: 04 Nov. 2022.

WALES, Michael. **Udacity**, 2021. 3 Web dev careers decoded: Front-end vs Back-end vs Full Stack. Disponível em: https://www.udacity.com/blog/2020/12/front-end-vs-back-end-vs-full-stack-web-developers.html Acesso em: 07 Mai. 2023.

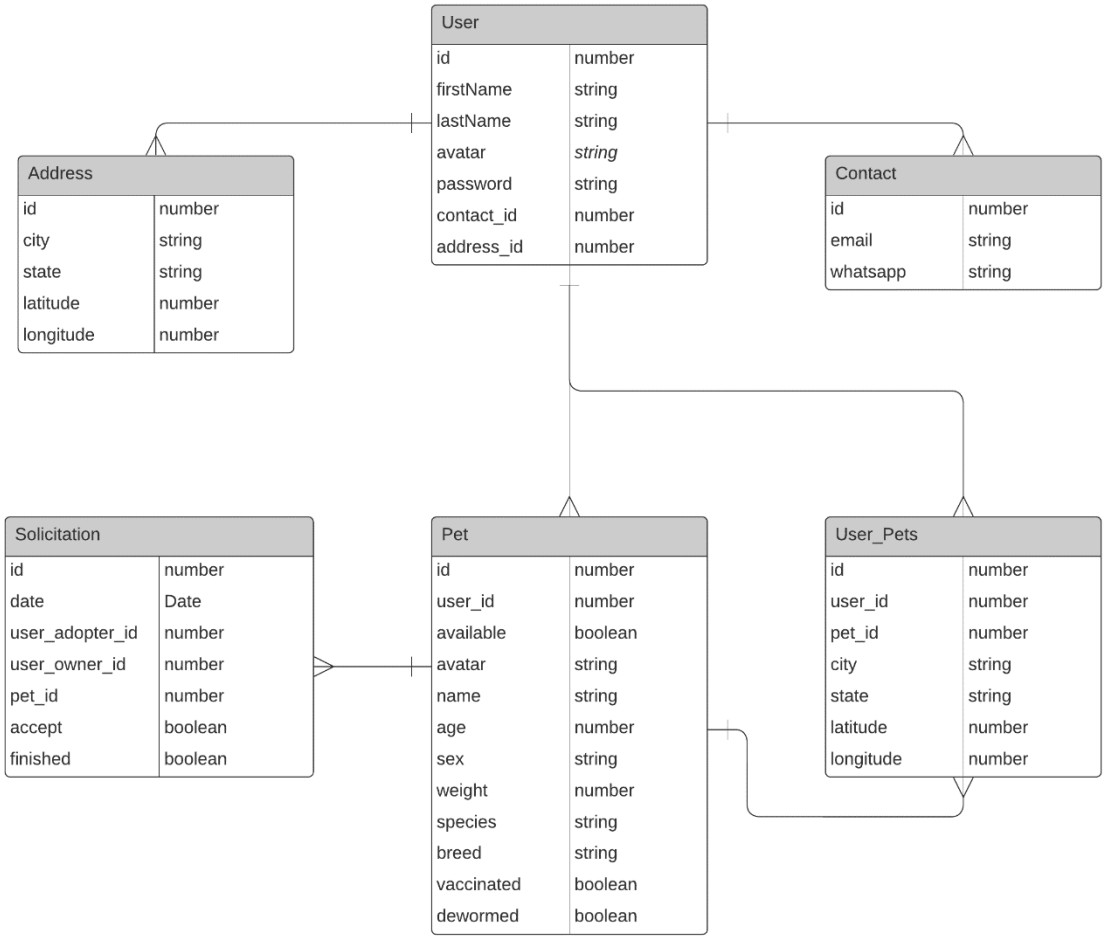
APÊNDICE A – QUESTIONÁRIO DO PROTÓTIPO

Número	Pergunta	Tipo de pergunta
01	Qual seu nome?	Aberta
02	Qual sua faixa etária?	Fechada
03	Em qual cidade você mora?	Aberta
04	Esse aplicativo facilitará o processo de adoção de animais?	Fechada
05	O processo de cadastro de pets no aplicativo é fácil de ser realizado?	Fechada
06	A visualização dos pets no mapa é uma funcionalidade útil?	Fechada
07	As cores do aplicativo proporcionam uma sensação agradável?	Fechada
08	Os textos do aplicativo descrevem bem o objetivo de cada tela, deixando claro o que será realizado?	Fechada
09	Os ícones do aplicativo condizem com a informação apresentada?	Fechada
10	Os botões do aplicativo realizam as ações conforme estão descritos?	Fechada
11	Que outras funcionalidades você sugere para o aplicativo? Fique à vontade para sugeri-las.	Aberta
12	Você utilizaria esse app para auxiliar no processo de doação/adoção?	Fechada
13	Com base no aplicativo apresentado, qual seu nível de satisfação para os critérios abaixo:	Fechada

APÊNDICE B - DIAGRAMA DE CASO DE USO



APÊNDICE C - DIAGRAMA ENTIDADE RELACIONAMENTO



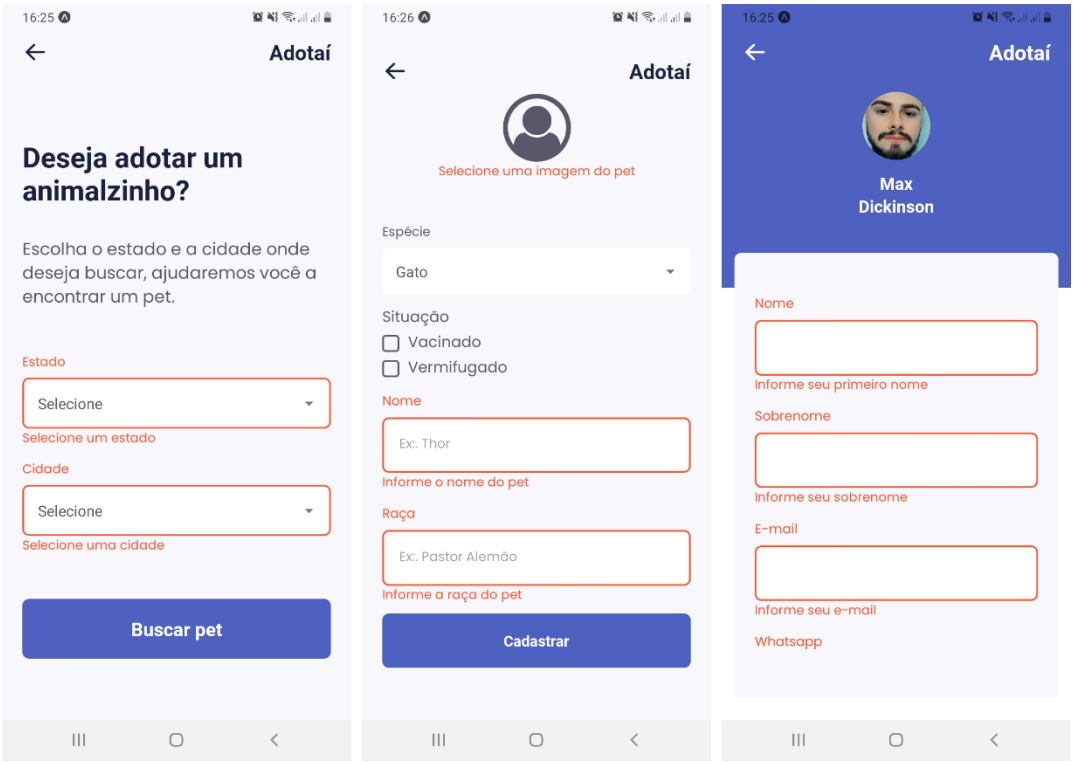
APÊNDICE D – TELAS DO APP COM VALIDAÇÃO

As figuras abaixo apresentam as telas do aplicativo com validações nas entradas de usuário. Caso o usuário não informe um campo obrigatório ou o preencha com uma informação diferente do esperado, uma borda vermelha é exibida no campo de texto ou seleção, seguido de uma mensagem de ajuda (ver Figura 47 e 48).

Figura 47- Aplicativo - Tela de login e cadastro do usuário com validação

Fonte: Elaborado pelo autor (2023)

Figura 48- Aplicativo - Telas de busca, cadastro de pet e edição de usuário



Fonte: Elaborado pelo autor (2023)

APÊNDICE E – QUESTIONÁRIO SUS

Número	Afirmações	P1	P2	P3	P4	P5	P6	P7	P8	P9	AVG
1	Eu acho que gostaria de usar esse aplicativo com frequência.	3	4	3	4	3	5	3	1	5	3,4
2	Eu acho o aplicativo fácil de usar.	4	5	4	4	5	1	4	4	5	4
3	Eu acho o aplicativo difícil de usar.	2	2	4	1	2	2	2	1	1	1,8
4	Eu acho que precisaria da ajuda de uma pessoa com conhecimentos técnicos para usar o aplicativo	1	1	1	1	1	4	1	1	1	1,3
5	Eu consigo cadastrar um usuário no aplicativo.	5	5	5	5	5	4	5	4	5	4,7
6	Eu consigo cadastrar um pet no aplicativo.	5	4	5	5	5	5	5	5	4	4,7
7	Eu consigo encontrar os pets disponíveis para adoção no aplicativo.	4	5	5	5	4	5	5	5	5	4,7
8	Eu achei o aplicativo atrapalhado de usar.	1	4	1	1	1	4	2	2	2	2
9	Eu me senti confiante ao usar o aplicativo.	4	4	4	5	4	5	4	3	4	4,1
10	Eu precisei aprender várias coisas novas antes de conseguir usar o aplicativo.	1	1	4	1	1	1	1	1	2	1,4
	MÉDIA SUS	65	62,5	65	70	62,5	65	65	52,5	65	63,6