



UNIVERSIDADE ESTADUAL DO TOCANTINS  
CÂMPUS DE PALMAS  
CURSO DE SISTEMAS DE INFORMAÇÃO

**ESTUDO E DESENVOLVIMENTO DE UMA APLICAÇÃO PARA O  
AUXÍLIO NA LOCALIZAÇÃO DE CACHORROS PERDIDOS  
UTILIZANDO REDES NEURAIS CONVOLUCIONAIS**

YHAN NUNES LEAL DA SILVA

Palmas - TO

2022



UNIVERSIDADE ESTADUAL DO TOCANTINS  
CÂMPUS DE PALMAS  
CURSO DE SISTEMAS DE INFORMAÇÃO

**ESTUDO E DESENVOLVIMENTO DE UMA APLICAÇÃO PARA O  
AUXÍLIO NA LOCALIZAÇÃO DE CACHORROS PERDIDOS  
UTILIZANDO REDES NEURAIS CONVOLUCIONAIS**

YHAN NUNES LEAL DA SILVA

Projeto apresentado ao Curso de Sistemas de Informação da Universidade Estadual do Tocantins - UNITINS como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

Palmas - TO

2022



## **CURSO DE SISTEMAS DE INFORMAÇÃO**

# **ESTUDO E DESENVOLVIMENTO DE UMA APLICAÇÃO PARA O AUXÍLIO NA LOCALIZAÇÃO DE CACHORROS PERDIDOS UTILIZANDO REDES NEURAIS CONVOLUCIONAIS**

**YHAN NUNES LEAL DA SILVA**

Projeto apresentado ao Curso de Sistemas de Informação da Universidade Estadual do Tocantins - UNITINS como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

---

**Me. Tamirys Virgulino Ribeiro Prado**  
Orientador

---

**Me. Douglas Chagas da Silva**  
Examinador

---

**Me. Silvano Maneck Malfatti**  
Examinador

Palmas - TO

2022

**Dados Internacionais de Catalogação na Publicação  
(CIP) Sistema de Bibliotecas da Universidade Estadual  
do Tocantins**

---

S586e

SILVA, Yhan Nunes Leal da  
ESTUDO E DESENVOLVIMENTO DE UMA  
APLICAÇÃO PARA O AUXÍLIO NA LOCALIZAÇÃO  
DE CACHORROS PERDIDOS UTILIZANDO REDES  
NEURAIS CONVOLUCIONAIS. Yhan Nunes Leal da  
Silva. - Palmas, TO, 2022

Monografia Graduação - Universidade Estadual do  
Tocantins – Câmpus Universitário de Palmas - Curso de  
Sistemas de Informação, 2022.

Orientadora: Tamirys Virgulino Ribeiro Prado

1. inteligência artificial. 2. animais de estimação. 3.  
redes neurais convolucionais. 4. identificação de  
padrões.

**CDD 610.7**

TODOS OS DIREITOS RESERVADOS – A reprodução total ou parcial, de qualquer forma ou por  
qualquer meio deste documento é autorizado desde que citada a fonte. A violação dos direitos do  
autor (Lei nº 9.610/98) é crime estabelecido pelo artigo 184 do Código Penal.

**Elaborado pelo sistema de geração automática de ficha catalográfica da UNITINS com os  
dados fornecidos pelo(a) autor(a).**

**ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO DO CURSO DE SISTEMAS DE INFORMAÇÃO DA FUNDAÇÃO UNIVERSIDADE ESTADUAL DO TOCANTINS - UNITINS**

Aos **13** dias do mês de **Dezembro** de **2022**, reuniu-se na Fundação Universidade Estadual do Tocantins, Câmpus Palmas, Bloco B, às **20:00 horas**, sob a Coordenação do Professora **Tamirys Virgulino Ribeiro Prado**, a banca examinadora de Trabalho de Conclusão de Curso em Sistemas de Informação, composta pelos examinadores Professora **Tamirys Virgulino Ribeiro Prado** (Orientadora), Professor **Douglas Chagas da Silva** e Professor **Silvano Maneck Malfatti**, para avaliação da defesa do trabalho intitulado “**Estudo e Desenvolvimento de uma Aplicação para o Auxílio na Localização de Cachorros Perdidos Utilizando Redes Neurais Convolucionais**” do acadêmico **Yhan Nunes Leal da Silva** como requisito para aprovação na disciplina Trabalho de Conclusão de Curso (TCC). Após exposição do trabalho realizado pelo acadêmico e arguição pelos Examinadores da banca, em conformidade com o disposto no Regulamento de Trabalho de Conclusão de Curso em Sistemas de Informação, a banca atribuiu a pontuação 10.0 .

Sendo, portanto, o Acadêmico: ☒ (X) Aprovado    ☐ ( ) Reprovado

Assinam esta Ata:

Professor Orientador: **Tamirys Virgulino Ribeiro Prado**

Examinador: **Douglas Chagas da Silva**

Examinador: **Silvano Maneck Malfatti**

**Tamirys Virgulino Ribeiro Prado**  
**Presidente da Banca Examinadora**

Coordenação do Curso de Sistemas de Informação



*Este trabalho é dedicado à minha família, e amigos, que nunca me deixou desistir e sempre me apoiou nesta fase tão importante da minha vida.*

# Agradecimentos

Agradeço a Deus, em primeiro lugar, pela força, perseverança e oportunidade de concluir o curso em uma instituição pública e gratuita. Aos meus pais, que acreditaram sempre em mim mesmo em meio aos meus fracassos e foram essenciais na formação do meu caráter. A minha professora orientadora e aos professores que me ajudaram toda a trajetória para entrega deste trabalho. Agradeço aos meus amigos em especial Anne Brandão e Anny Santos por sempre me motivar a não desistir, e aos amigos Júlio César, Davi Silva, Henrique Beckmann, Brayan Mota, Caio César, Richard Bruno, Rafael Freitas, Eduardo Rocha e Gabriel Beckman por me acompanharem neste período de 4 anos de conclusão deste curso, meu mais sincero obrigado e essa vitória não seria possível sem vocês.

*“O sucesso nada mais é que ir de fracasso em fracasso sem perder o entusiasmo”.*

*Winston Churchill*



# Resumo

Com o passar dos anos, nota-se uma crescente aquisição de animais domésticos por parte da população em geral, muitos deles sendo considerados membros da família. Consoante a isto, o aumento da popularidade da internet, permitiu que pessoas em qualquer lugar do mundo, proporcionando um mundo de possibilidade no ramo de relações interpessoais. Contudo, devido ao aumento do número de animais de estimação nos lares das famílias, cresceu também problemas relacionados a perda destes animais de estimação, situação esta que se mostra problemática para os donos de animais, onde apesar de cada vez mais os mesmos estarem imersos nessa nova realidade proporcionada pela internet, muitos donos de pets acabam se deparando com dificuldades em encontrar ferramentas objetivas para o auxílio de problemas relacionados a perda de animais e interação com outros donos, necessidade esta que se torna indispensável para esta parcela da comunidade, tanto por uma questão de saúde do animal como questões relacionadas a tranquilidade ao procurar opções que solucionem estes problemas. Visto a falta de foco claro com relação as principais aplicações de bate-papo virtuais do Brasil em relação a este tema. Esta pesquisa apresenta um estudo para desenvolvimento de um aplicativo de comunicação que auxilie os donos de animais com problemas relacionados à perda de animais de estimação utilizando inteligência artificial no modelo de redes neurais convolucionais e geolocalização dos usuários. Sendo desenvolvido com o framework Flutter em conjunto com as plataformas de serviços em nuvem Firebase e Google Cloud.

**Palavras-chave:** aplicativos de bate-papo, aplicativo de geolocalização, pets, animais de estimação, redes neurais, redes neurais convolucionais, processamento de imagem, inteligência artificial.

# Abstract

Over the years, there has been a growing acquisition of domestic animals by the general population, many of which are considered family members. Accordingly, the rise in popularity of the internet has allowed people anywhere in the world, providing a world of possibilities in the field of interpersonal relationships. However, due to the increase in the number of pets in family homes, problems related to the loss of these pets also grew, a situation that is proving to be problematic for pet owners, where despite the fact that they are increasingly immersed in this new reality provided by the internet, many pet owners end up facing difficulties in finding objective tools to help with problems related to the loss of animals and interaction with other owners, a need that becomes indispensable for this part of the community, both for animal health issues as well as issues related to peace of mind when looking for options to solve these problems. Given the lack of clear focus regarding the main virtual chat applications in Brazil in relation to this topic. This research presents a study for the development of a communication application that helps pet owners with problems related to the loss of pets using artificial intelligence in the model of convolutional neural networks and geolocation of users. Being developed with the Flutter framework in conjunction with the Firebase and Google Cloud cloud service platforms.

**Key-words:** chat applications, geolocation application, pets, neural networks, convolutional neural networks, image processing, artificial intelligence

# Lista de ilustrações

Figura 1 – Flutter x React. . . . .	23
Figura 2 – Rede Neural. . . . .	25
Figura 3 – Rede Neural. . . . .	26
Figura 4 – Rede Neural Convolucional. . . . .	27
Figura 5 – Entradas Rede Neural Convolucional. . . . .	28
Figura 6 – Camada convolução rede neural. . . . .	28
Figura 7 – Camada pooling rede neural. . . . .	29
Figura 8 – Raças do dataset 1. . . . .	37
Figura 9 – Raças do dataset 2. . . . .	38
Figura 10 – Raças do dataset 3. . . . .	39
Figura 11 – Entrada da rede neural. . . . .	40
Figura 12 – Fluxograma Implementação da primeira rede neural. . . . .	41
Figura 13 – Fluxograma Implementação da primeira rede neural. . . . .	42
Figura 14 – Camadas da primeira rede neural via Netron. . . . .	43
Figura 15 – Fluxograma Implementação da segunda rede neural. . . . .	44
Figura 16 – Impressão da configuração da segunda rede neural. . . . .	45
Figura 17 – Camadas da segunda rede neural via Netron. . . . .	46
Figura 18 – Fluxograma Implementação da terceira rede neural. . . . .	47
Figura 19 – Impressão da configuração da terceira rede neural. . . . .	48
Figura 20 – Camadas da segunda rede neural via Netron. . . . .	49
Figura 21 – Fluxograma Implementação da quarta rede neural. . . . .	50
Figura 22 – Impressão da configuração da quarta rede neural. . . . .	51
Figura 23 – Camadas da quarta rede neural via Netron. . . . .	52
Figura 24 – Disposição dos usuários fictícios no mapa de Palmas. . . . .	55
Figura 25 – Aplicação do método Elbow para definição do número de centroids a ser utilizado. . . . .	56
Figura 26 – Disposição dos centroides gerados no mapa com 4 clusters. . . . .	57
Figura 27 – Máquina de estado finito para notificação dos donos de animais. . . . .	61
Figura 28 – Representação do algoritmo da Máquina de estado finito. . . . .	62
Figura 29 – Diagrama de container. . . . .	64
Figura 30 – Short caption . . . . .	65
Figura 31 – Arquitetura do Aplicativo (Clean Architecture). . . . .	66
Figura 32 – UML de Pacotes do Sistema. . . . .	67
Figura 33 – Diagrama de Atividades do Cadastro no Sistema. . . . .	68
Figura 34 – Diagrama de Atividades de Seleção de raça do animal. . . . .	70
Figura 35 – Acurácia da primeira rede neural. . . . .	73

Figura 36 – Perda da primeira rede neural. . . . .	73
Figura 37 – Acurácia da segunda rede neural. . . . .	74
Figura 38 – Perda da segunda rede neural. . . . .	75
Figura 39 – Acurácia da terceira rede neural. . . . .	76
Figura 40 – Perda da terceira rede neural. . . . .	76
Figura 41 – Acurácia da quarta rede neural. . . . .	77
Figura 42 – Perda da quarta rede neural. . . . .	78
Figura 43 – Imagens com predição correta da quarta rede neural. . . . .	78
Figura 44 – Imagens com predição incorreta da quarta rede neural. . . . .	79
Figura 45 – Fluxograma de separação das imagens em grupos. . . . .	81
Figura 46 – Categoria das raças após agrupamento. . . . .	82
Figura 47 – Matriz de Confusão. . . . .	83
Figura 48 – Telas Iniciais do Sistema. . . . .	85
Figura 49 – Telas de identificação de raça. . . . .	86
Figura 50 – Telas de listagem de animais perdidos. . . . .	87
Figura 51 – Telas de notificação de usuário. . . . .	88
Figura 52 – Telas de Chat e Perfil do usuário. . . . .	89
Figura 53 – Tela de confirmação do animal. . . . .	90
Figura 54 – Tela de conversa entre donos. . . . .	91
Figura 55 – Ícone Sonarqube. . . . .	92
Figura 56 – Tela de conversa entre donos. . . . .	93
Figura 57 – Cabeçalho do formulário. . . . .	102
Figura 58 – Pergunta 1. . . . .	102
Figura 59 – Pergunta 2. . . . .	103
Figura 60 – Pergunta 3. . . . .	103
Figura 61 – Pergunta 4. . . . .	103
Figura 62 – Pergunta 5. . . . .	103
Figura 63 – Pergunta 6. . . . .	104
Figura 64 – Pergunta 7. . . . .	104
Figura 65 – Pergunta 8. . . . .	104
Figura 66 – Pergunta 9. . . . .	104
Figura 67 – Pergunta 10. . . . .	105
Figura 68 – Pergunta 11. . . . .	105
Figura 69 – Pergunta 12. . . . .	105
Figura 70 – Pergunta 13. . . . .	106
Figura 71 – Resposta 1. . . . .	107
Figura 72 – Resposta 2. . . . .	107
Figura 73 – Resposta 3. . . . .	108
Figura 74 – Resposta 4. . . . .	108

Figura 75 – Resposta 5. . . . .	109
Figura 76 – Resposta 6. . . . .	109
Figura 77 – Resposta 7. . . . .	110
Figura 78 – Resposta 8. . . . .	110
Figura 79 – Resposta 9. . . . .	110
Figura 80 – Resposta 10. . . . .	111
Figura 81 – Resposta 11. . . . .	111
Figura 82 – Resposta 12. . . . .	112
Figura 83 – Resposta 13. . . . .	113
Figura 84 – Nuvem de palavras 1. . . . .	114
Figura 85 – Nuvem de palavras 2. . . . .	115
Figura 86 – Requisito Funcional 01 . . . . .	117
Figura 87 – Requisito Funcional 02 . . . . .	118
Figura 88 – Requisito Funcional 03 . . . . .	118
Figura 89 – Requisito Funcional 04 . . . . .	119
Figura 90 – Requisito Funcional 05 . . . . .	119
Figura 91 – Requisito Funcional 06 . . . . .	120
Figura 92 – Requisito Funcional 07 . . . . .	120
Figura 93 – Requisito Funcional 08 . . . . .	120
Figura 94 – Requisito Funcional 09 . . . . .	121
Figura 95 – Requisito Funcional 10 . . . . .	121
Figura 96 – Requisito Funcional 11 . . . . .	121
Figura 97 – Requisito Funcional 12 . . . . .	122
Figura 98 – Requisito Não Funcional 01 . . . . .	123
Figura 99 – Requisito Não Funcional 02 . . . . .	123
Figura 100–Requisito Não Funcional 03 . . . . .	124
Figura 101–Requisito Não Funcional 04 . . . . .	124
Figura 102–Requisito Não Funcional 05 . . . . .	124
Figura 103–Requisito Não Funcional 06 . . . . .	125
Figura 104–Requisito Não Funcional 07 . . . . .	125

# Lista de abreviaturas e siglas

**AOT** - Ahead Of Time.

**API** - Application Programming Interface.

**APP** - Application.

**Pet** - Animais de estimação.

**HTTP** - Hypertext Transfer Protocol.

**JIT** - Just In Time.

**OSI** - Open System Interconnection.

**SGBD** - Sistema Gerenciador de Banco de Dados.

**NoSql** - Não relacional.

**IA** - Inteligência Artificial.

**VM** - Virtual Machine.

**I.A** - Inteligência Artificial.

**CNN** - Rede Neural Convolucional.

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>17</b>
<b>1.1</b>	<b>Justificativa</b>	<b>18</b>
<b>1.2</b>	<b>Objetivos</b>	<b>19</b>
1.2.1	Objetivo Geral	19
1.2.2	Objetivos Específicos	19
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>20</b>
<b>2.1</b>	<b>Trabalhos Relacionados</b>	<b>20</b>
<b>2.2</b>	<b>Aplicativos de Comunicação para donos de pets existentes na Atualidade</b>	<b>20</b>
2.2.1	Pupz	21
2.2.2	Flockr	21
<b>2.3</b>	<b>Dart e Flutter</b>	<b>22</b>
<b>2.4</b>	<b>Firebase</b>	<b>22</b>
<b>2.5</b>	<b>Google Cloud</b>	<b>23</b>
<b>2.6</b>	<b>Docker</b>	<b>24</b>
<b>2.7</b>	<b>Tensorflow</b>	<b>24</b>
2.7.1	Python	24
<b>2.8</b>	<b>Kmeans</b>	<b>25</b>
<b>2.9</b>	<b>Máquina de estados finitos</b>	<b>25</b>
<b>2.10</b>	<b>Rede Neural</b>	<b>26</b>
2.10.1	Rede Neural Convolucional	27
2.10.1.1	Entrada	27
2.10.1.2	Convolução	28
2.10.1.3	Funções de Ativação	29
2.10.1.4	Pooling	29
2.10.2	Inception V3	30
2.10.3	Mobilenet V2	30
<b>3</b>	<b>METODOLOGIA</b>	<b>31</b>
<b>3.1</b>	<b>Materiais</b>	<b>31</b>
3.1.1	Ferramentas utilizadas para a pesquisa	34
<b>3.2</b>	<b>Funcionamento da Aplicação</b>	<b>35</b>
<b>3.3</b>	<b>Banco de imagens utilizado no treinamento</b>	<b>36</b>
<b>3.4</b>	<b>Treinamento das redes neurais</b>	<b>40</b>
3.4.1	Primeira rede neural	41

3.4.2	Segunda rede neural . . . . .	44
3.4.3	Terceira rede neural (MobileNetV2) . . . . .	47
3.4.4	Quarta rede neural (InceptionV3) . . . . .	50
<b>3.5</b>	<b>Agrupamento dos usuários utilizando K-means . . . . .</b>	<b>53</b>
3.5.1	Geração sintética dos dados para agrupamento dos usuários . . . . .	53
3.5.2	Elbow method . . . . .	56
3.5.3	Adição de um usuário novo do aplicativo no seu respectivo grupo . . . . .	58
<b>3.6</b>	<b>Máquina de estados finitos para identificação do animal perdido . . . . .</b>	<b>60</b>
3.6.1	Ações ao identificar de um animal perdido . . . . .	60
<b>3.7</b>	<b>UML (<i>Unified Model Language - Linguagem de Modelagem Unificada</i>) . . . . .</b>	<b>63</b>
3.7.1	Diagrama de Contêineres . . . . .	64
3.7.2	UML de Caso de Uso . . . . .	65
3.7.3	UML de Pacotes . . . . .	66
<b>3.8</b>	<b>Diagrama de Atividades . . . . .</b>	<b>68</b>
3.8.1	Diagrama de Atividades do Login de um usuário . . . . .	68
3.8.2	Diagrama de seleção da raça do animal . . . . .	70
<b>4</b>	<b>RESULTADOS . . . . .</b>	<b>72</b>
<b>4.1</b>	<b>Desempenho redes neurais . . . . .</b>	<b>72</b>
4.1.1	Primeira Rede neural . . . . .	72
4.1.2	Segunda Rede neural . . . . .	74
4.1.3	Terceira Rede neural (MobileNetV2) . . . . .	75
4.1.4	Quarta Rede neural (InceptionV3) . . . . .	77
4.1.4.1	Matriz Confusão . . . . .	79
<b>4.2</b>	<b>Protótipos de Telas Desenvolvidas . . . . .</b>	<b>84</b>
4.2.1	Telas Iniciais do Sistema . . . . .	84
4.2.2	Telas de cadastro do animal de estimação . . . . .	86
4.2.3	Telas de visualização dos animais perdidos . . . . .	87
4.2.4	Telas de notificação dos animais perdidos pelo sistema . . . . .	88
4.2.5	Telas de Chat e Perfil do usuário . . . . .	89
<b>4.3</b>	<b>Validação junto a Lei Geral de Proteção de dados . . . . .</b>	<b>89</b>
<b>4.4</b>	<b>Testes de qualidade de código . . . . .</b>	<b>92</b>
<b>4.5</b>	<b>Ressalvas da aplicação . . . . .</b>	<b>94</b>
<b>5</b>	<b>CONCLUSÃO . . . . .</b>	<b>95</b>
<b>5.1</b>	<b>Trabalhos futuros . . . . .</b>	<b>96</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>97</b>
<b>5.2</b>	<b>Anexo A - Questionário para Validação de uma Aplicação de auxílio na localização de animais perdidos . . . . .</b>	<b>102</b>



5.2.1	Perguntas do formulário . . . . .	102
5.2.2	Respostas do formulário . . . . .	107
5.2.3	Nuvem de palavras . . . . .	114

## **APÊNDICES** **116**

	<b>APÊNDICE A – REQUISITOS DO SISTEMA . . . . .</b>	<b>117</b>
A.0.1	Requisitos Funcionais . . . . .	117
A.0.2	Requisitos Não Funcionais . . . . .	123

# 1 Introdução

A presença de animais domésticos nos lares das famílias brasileiras vem tornando-se cada vez mais frequente de acordo com pesquisa realizada pelo IBGE ([IBGE, 2019a](#)).

Segundo Santana ([SANTANA, 2004](#)) a mudança comportamental gerada pelo relacionamento entre seres humanos e animais domésticos está cada vez mais relevante, tendo em vista que os animais passaram a ser considerados membros da família, e isso muitas vezes é motivo de preocupação e bem-estar no dia a dia dos seus donos.

Com o crescimento de ferramentas de comunicação, pessoas com interesses semelhantes conseguem entrar em contato com mais facilidade. Atualmente as principais ferramentas em circulação no território brasileiro segundo ([LOUREIRO, 2019](#)) são: WhatsApp, Telegram e Messenger. Essas ferramentas não permitem de maneira clara que usuários que não tenham tido contato prévio interajam entre si.

Contexto este, que se estende a donos de animais domésticos, sobretudo ao fato de frequentemente precisam se comunicar ou compartilhar dúvidas com outros donos de animais de modo a auxiliar em casos de desaparecimento de animais. Outra abordagem é proporcionar *redes*<sup>1</sup> de comunicação entre os mesmos a fim de compartilhar notícias ou simplesmente apoio mutuo em relação à localização e saúde de animais desaparecidos.

Apesar de existir algumas aplicações em circulação no mercado que abordam em certo nível soluções para perda de animais de estimação, como, por exemplo, o aplicativo Pupz ([PUPZ, 2022](#)), é perceptível a carência de ferramentas que abordam como foco tanto sistemas de comunicação quanto maneira de identificação de animais de estimação. Muitas vezes recurso como comunicação entre os donos de animais para este propósito, necessitam a inclusão de alguma aplicação externa como Instagram, Twitter ou outras mídias sociais para estabelecer este contato.

Perante ao exposto, este trabalho tem como finalidade o estudo do desenvolvimento de uma aplicação voltada a uma nova abordagem de comunicação pública: um aplicativo de bate-papo voltado a donos de animais para o auxílio na localização de animais perdidos, de modo a proporcionar rápida e fácil comunicação entre donos de animais que eventualmente acabam perdendo seus pets. A aplicação funcionará identificando a raça do animal através de uma foto previamente inserida pelo mesmo, limitando os usuários disponíveis através da localização proveniente de um sistema de agrupamento, realizando filtros para avisar o dono de um animal perdido que um animal semelhante foi identificado na devida localização da foto.

---

<sup>1</sup> redes: transferência eletrônica de informações, comunicação mediada por um computador ou periférico.

## 1.1 Justificativa

Apesar da crescente presença de animais de estimação nos lares das famílias, cresce também o número de animais abandonados ou perdidos ao redor do mundo, relatado que apenas na década passada o território brasileiro contava com 30 milhões de animais de rua em seu território ([JUSBASIL, 2022](#)), número este a aumentar com o passar do tempo.

Outrossim, apesar das facilidades decorrentes do advento da internet, muitos donos de animais encontram dificuldade em encontrar ferramentas e se comunicar entre si quando se deparam com a problemática de animais de estimação perdidos. Problema decorrente desde a falta de ferramentas com o focados para este propósito. Para o vigente trabalho, será abordado soluções para a ausência de ferramentas focadas neste propósito para o contexto de perda de animais de estimação.

Em buscas realizadas para este trabalho, foram encontrados algumas aplicações que proporcionam funcionalidades para o problema de animais de estimação perdidos, porém essas aplicações não englobam a interação de usuários entre si, nem interações baseadas na geolocalização, necessitando de algum contato prévio entre os usuários através de ferramentas de terceiros para estabelecer esta comunicação, dificultando assim para os donos de animais de estimação interajam com usuários que possivelmente localizaram seus animais fujões. Portanto, este trabalho pretende avaliar o estudo de caso de uma aplicação de localização de animais perdidos, além de proporcionar maneiras de comunicação entre o usuário que localizou o animal e possíveis donos do mesmo de maneira dinâmica através da própria ferramenta.

## 1.2 Objetivos

### 1.2.1 Objetivo Geral

Realizar o estudo acadêmico para analisar a viabilidade e desenvolver um aplicativo que permita a localização de animais perdidos, além da comunicação entre donos, baseado no usuário que encontrou o animal e possíveis donos do mesmo, baseados na geolocalização e raça do animal encontrado.

### 1.2.2 Objetivos Específicos

- Aplicar uma rede neural convolucional no reconhecimento de imagem.
- Desenvolver uma arquitetura para o desenvolvimento do aplicativo.
- Desenvolver um algoritmo de geolocalização dos usuários.
- Aplicar um algoritmo de *clusterização*<sup>2</sup> para os usuários.
- Aplicar um estudo de caso sobre a API desenvolvida.

---

<sup>2</sup> clusterização: agrupamento de itens

## 2 Referencial Teórico

O presente capítulo aborda os fundamentos e técnicas necessárias para o desenvolvimento deste projeto de conclusão de curso, comparando aplicativos similares, embasamento teórico e descrevendo as tecnologias necessárias para o desenvolvimento da aplicação.

### 2.1 Trabalhos Relacionados

A partir de diversas pesquisas, foram constatados alguns trabalhos que possuem relação com partes do presente trabalho, desde a identificação de raça dos animais utilizando redes neurais e sistemas de geolocalização e identificação de animais utilizando diferentes técnicas.

Dentre eles, podemos citar o trabalho realizado por alunos da universidade de Stanford (KHOSLA et al., 2011), onde é coletado e separado diversas imagens de raças de cachorro para o treinamento e teste de algoritmos de aprendizado de máquina.

Outro exemplo é o trabalho realizado pela aluna Jéssica Salvador Rodrigues da Rocha (ROCHA, 2019), aonde a partir da informação de localização provida pela plataforma *Facebook*, é realizado o agrupamento dos donos e animais para encontrar possíveis donos para animais abandonados ou auxiliar na busca de animais perdidos.

Jacqueline Rafaela (FISTAROL, 2018), projeto para auxiliar ONGS a encontrar animais de rua e auxiliar os mesmos no processo de adoção, válido ressaltar que a partir do trabalho da Jacqueline foi encontrado outro aplicativo relacionado com a proposta do vigente trabalho. O aplicativo *Pet.me*, utiliza o sistema de geolocalização para identificar animais perdidos e deixa disponível para usuários interessados coletarem para adoção ou entregar a ONGS, além de disponibilizar outros serviços como localização de Pet shop entre outros serviços (HUGO, 2022).

### 2.2 Aplicativos de Comunicação para donos de pets existentes na Atualidade

O mercado brasileiro está repleto de aplicativos que proporcionam interação entre usuários, seja por troca de mensagens ou atuando como um registro diário das atividades realizadas pelas pessoas. Sem dúvidas esses aplicativos mudaram como a sociedade moderna se comunica e interage virtualmente, descartando-se entre os demais: *WhatsApp*<sup>1</sup>,

---

<sup>1</sup> WhatsApp: <https://faq.whatsapp.com/>

*Instagram*<sup>2</sup> e *Telegram*<sup>3</sup>, nos quais são listados como mais utilizados pela população brasileira segundo (LOUREIRO, 2019). Porém, entre os aplicativos listados nenhum deles tem foco específico para donos de animais de estimação, entretanto nas principais lojas de aplicativos destacamos algumas soluções com foco claro na interação entre donos de animais, entre eles o aplicativo Flockr que proporciona aos usuários uma experiência semelhante ao Instagram no quesito plataforma de mídias sociais, porém voltado para uma abordagem de proporcionar dicas a respeito dos cuidados com animais sendo estas previamente cadastradas na plataforma (SILVA, 2021).

### 2.2.1 Pupz

O aplicativo *Pupz* possui a proposta de ser uma aplicação utilitária para donos de animais de estimação, possuindo recursos como comunicação entre donos e veterinários, localização de animais perdidos por fotos e geração de prontuário para clínicas em geral (PUPZ, 2022). Apesar de possuir uma proposta semelhante ao vigente trabalho, o aplicativo Pupz não possui um sistema de identificação muito eficiente, onde para identificar o animal é necessário que o usuário fotografe a cara do animal de maneira próxima, com o intuito de realizar um sistema de *mapeamento facial* e assim identificar o mesmo, porém ao tratar de animais perdidos, é comum se deparar com situações em que o animal está assustado assim dificultando a captura da foto de maneira próxima.

### 2.2.2 Flockr

Voltado para a interação dos donos de animais de estimação, o Flockr se propõe a ser um aplicativo que permite aos donos de animais de estimação retirem dúvidas e recebam dicas de cuidados com os animais, além de permitir publicação de fotos e registros diários dos mesmos, semelhante ao Instagram (SILVA, 2021). Apesar de possuir donos de animais de estimação como foco, o Flockr não possui um sistema baseado na geolocalização do usuário nem uma busca clara com foco na raça do animal em questão, sendo seu funcionamento semelhante ao do Instagram, sendo um aplicativo que não engloba as funcionalidades propostas neste trabalho.

---

<sup>2</sup> Instagram: <https://www.instagram.com>

<sup>3</sup> Telegram: <https://core.telegram.org>

## 2.3 Dart e Flutter

Para o desenvolvimento da aplicação, foi escolhida a linguagem Dart, junto ao seu Framework Flutter, a mesma sendo escolhida por ser uma tecnologia já apresentada no curso de Sistemas de Informação e possuir uma curva de aprendizado menor, facilitando assim o desenvolvimento.

O Dart é AOT(Ahead Of Time) e JIT(Just In Time) trazendo a performance da compilação através da Dart VM (Virtual Machine) e a velocidade da interpretação de ciclos de desenvolvimento sem a necessidade recompilação de todo o código a cada alteração (OFICIAL, 2011).

O Flutter é framework de desenvolvimento de código aberto escrito em Dart criado pela Google, usado para desenvolvimento de aplicações híbridas (Android e iOS), Web e Desktop (OFICIAL, 2017). Além disso, em sua versão mais recente adicionou suporte nativo a plataforma de desenvolvimento Firebase. essencial para o desenvolvimento desse projeto. (MARINHO, 2020).

Diferente de frameworks concorrentes como o React Native, que utiliza *bridges*<sup>4</sup> para acessar os recursos nativos e compilar para diferentes sistemas operacionais. O Flutter utiliza uma abordagem de compilação nativa de código, onde o código feito em Dart é convertido para código nativo *Android*<sup>5</sup>, e *IOS*<sup>6</sup>.

Abaixo pode ser analisado a comparação entre os dois frameworks.

O framework Flutter atende de maneira assertiva o objetivo da aplicação, onde o mesmo é possuir uma aplicação móvel híbrida de alto desempenho e com apenas um código funcionando como um *BaaS*<sup>7</sup> fornecido pelo Firebase

## 2.4 Firebase

O Firebase é uma plataforma de desenvolvimento de aplicações (GOOGLE, 2022b). Ele fornece aos desenvolvedores, ferramentas para facilitar o desenvolvimento de aplicações, sem necessariamente possuir uma Backend dedicado para a aplicação.

O serviço que será utilizado para o desenvolvimento dessa aplicação será o armazenamento de dado do Cloud Firestore junto ao Storage disponível pelo mesmo, além da utilização das Cloud Functions e Machine Learning para processamento da geolocalização e identificação de imagens. (GOOGLE, 2022c)

---

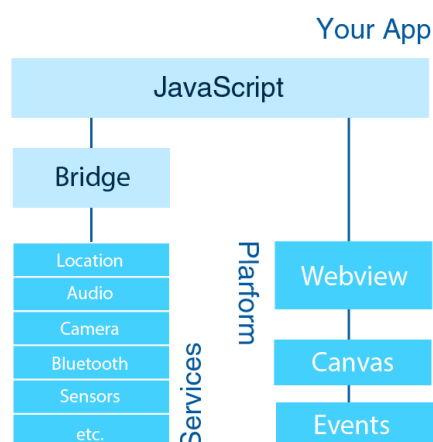
<sup>4</sup> Bridges: biblioteca de terceiros ou recursos adicionais.

<sup>5</sup> Android: atual sistema operacional utilizado pela Google e com o maior número de usuários.

<sup>6</sup> IOS: atual sistema operacional desenvolvido pela Apple.

<sup>7</sup> BaaS: Backend as a Service, conceito de aplicação que apenas provê funcionalidades para o aplicativo mobile ou Web.

### React Native:



### Layout Flutter:

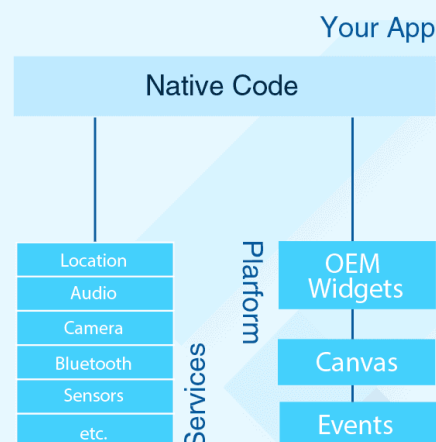


Figura 1 – Flutter x React.

**Fonte:** (RASHEVSKAYA, 2020)

O Firebase atende muito bem o intuito da aplicação, pois retira a necessidade de um backend dedicado para algumas funcionalidades da mesma. Podendo fazer os cálculos de geolocalização e processamento de imagem pelos próprios serviços ofertados pela ferramenta.

## 2.5 Google Cloud

O Google Cloud é um conjunto de ferramentas que possuem como objetivo acelerar o desenvolvimento das aplicações, ofertando recursos nos servidores da própria Google que realizam tarefas como hospedagem de aplicações, acesso a ferramentas do próprio Google como Mapas, Geolocalização, Inteligência artificial entre outros (GOOGLE, 2022e).

Os serviços utilizados para o desenvolvimento da aplicação será a hospedagem de *Containers*<sup>8</sup> através do *Google Cloud Run*<sup>9</sup>, a partir de um Backend Docker para a *API*<sup>10</sup> onde ficará hospedada a rede neural para consumo na aplicação.

O Google Cloud atende muito bem a proposta deste trabalho por possuir a possibilidade de hospedar a aplicação onde está disponibilizada a rede neural sem a necessidade de uma configuração complexa da aplicação, conforme exigido em diferentes sistemas de hospedagem.

<sup>8</sup> Containers: virtualização de nível de sistema operacional para entregar software em pacotes chamados contêineres.

<sup>9</sup> Google Cloud Run: Sistema de hospedagem de contêineres da Google, disponibilizando serviços em nuvem.

<sup>10</sup> API: Interface de programação de aplicações.



## 2.6 Docker

Docker é um conjunto de produtos de plataforma como serviço que usam virtualização de nível de sistema operacional, com o intuito de acelerar o desenvolvimento de aplicações e ajudar o desenvolvedor na hospedagem e generalização de ambientes de desenvolvimento utilizando o sistema de contêineres ([DOCKER, 2022](#)).

A utilização do Docker atende muito bem o trabalho, por permitir a utilização de uma API para consumo da rede neural de maneira a disponibilização da mesma através da integração com o Google Cloud.

## 2.7 Tensorflow

O TensorFlow é uma plataforma de código aberto com foco em desenvolvimento de soluções *Machine Learning*<sup>11</sup>.

Desenvolvida pela Google em 2015, hoje é o principal conjunto de bibliotecas disponível no mercado para trabalhos com rede neural e demais ferramentas relacionadas com inteligência artificial. ([TENSORFLOW, 2022](#))

Sua utilização é bastante semelhante à programação utilizando a linguagem Python, inclusive possuindo bibliotecas para a linguagem, possuindo a maior gama de utilizadores e resolução de problemas vinculadas a mesma. Sendo esta a abordagem utilizada para o desenvolvimento deste trabalho e treinamento da rede neural para identificação da raça do animal de estimação.

### 2.7.1 Python

O Python é uma linguagem de programação open source de alto nível desenvolvida por Guido van Rossum no ano de 1991, possuindo uma curva de aprendizado rápida, é uma ótima escolha para iniciantes além de ser utilizada em matéria da matriz curricular do curso de Sistema de Informação, entre seu funcionamento podemos destacar a interpretação de script, orientação a objetos, tipagem dinâmica entre outros. ([PYTHON, 2022](#))

Sua crescente utilização no ramo de Machine Learning e a facilidade proporcionada pela linguagem na integração das bibliotecas do TensorFlow faz do python uma escolha viável para realização do vigente trabalho.

---

<sup>11</sup> Machine Learning: Aprendizado de Máquina.

## 2.8 Kmeans

O algoritmo Kmeans é um dos algoritmos de agrupamento mais populares atualmente. Sua proposta é basicamente, a partir de um grupo de dados, os mesmos são agrupados a partir de características em comum, alocados em grupos a partir da proximidade dos dados entre si baseados na característica escolhida. Processo este sendo realizado calculando a média dos pontos membros e repetindo o processo de realocação e atualização até que os critérios de convergência seja satisfatório ao usuário.([SAMMUT, 2010](#))

Para a realização do trabalho será utilizado o algoritmo Kmeans para agrupamentos dos usuários através do critério de geolocalização, com o intuito de realizar o sistema de notificação e identificação de um grupo de usuários próximos a um animal identificado.

## 2.9 Máquina de estados finitos

Máquinas de estado finito são usadas para representar aplicações ou circuito lógico. Os conceitos são pensados como máquinas abstratas que devem conter um número finito de estados, sendo um estado chamado por vez, chamado de estado atual. Um estado armazena informações de entrada e saída, e para realizar uma transição entre os mesmos é necessária uma condição específica para realizar determinada mudança de estado chamadas dentro de um estado particular.([KERSCHBAUMER, 2021](#))

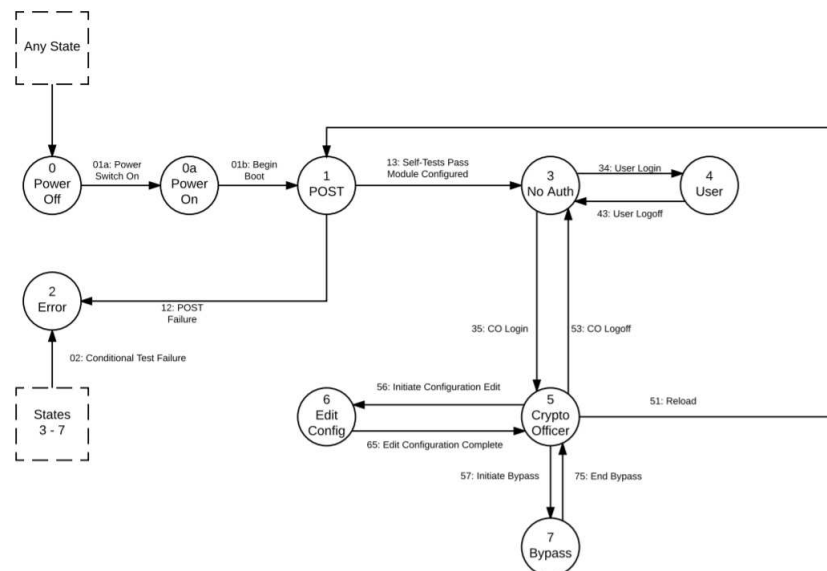


Figura 2 – Rede Neural.

**Fonte:** Oracle.

Para a realização do trabalho será utilizado uma máquina de estados finitos com o intuito de notificar e iniciar o sistema de Chat entre os usuários, enviando notificações e disponibilizando ao usuário opções de conversa entre o sistema e outros usuários.

## 2.10 Rede Neural

Redes neurais são um subconjunto no ramo de algoritmos de *Deep Learning*<sup>12</sup>, onde conseguem simular o comportamento do cérebro dos animais, com o intuito de realizar processos, encontrar padrões entre outras aplicações. (EDUCATION, 2022)

Seu funcionamento se baseia na simulação entre a conexão de neurônios, atribuídos com o nome de "nós" conectando-se entre si através de um sistema de peso e limite com o intuito de realizar operações matemáticas entre as camadas de redes, agrupando e classificando dados com o intuito de alcançar determinado objetivo, como demonstrado na imagem abaixo.

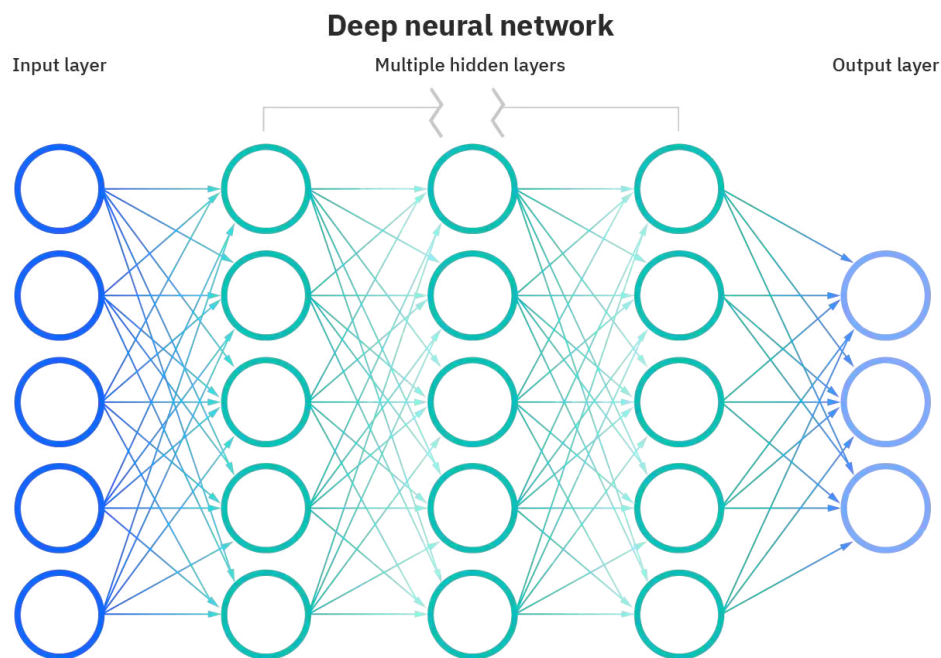


Figura 3 – Rede Neural.

Fonte: Ibm.

<sup>12</sup> Deep Learning: Aprendizado profundo.

### 2.10.1 Rede Neural Convolucional

Uma Rede Neural Convolucional (ou Convolutional Neural Network - CNN) é uma variação ao das redes de Perceptrons de Múltiplas Camadas, utilizada no processamento de dados visuais. Uma Rede Neural Convolucional é um algoritmo de aprendizado profundo capaz de a partir da entrada de uma imagem atribuir aspectos de diferença e importância das mesmas com diferentes propósitos.

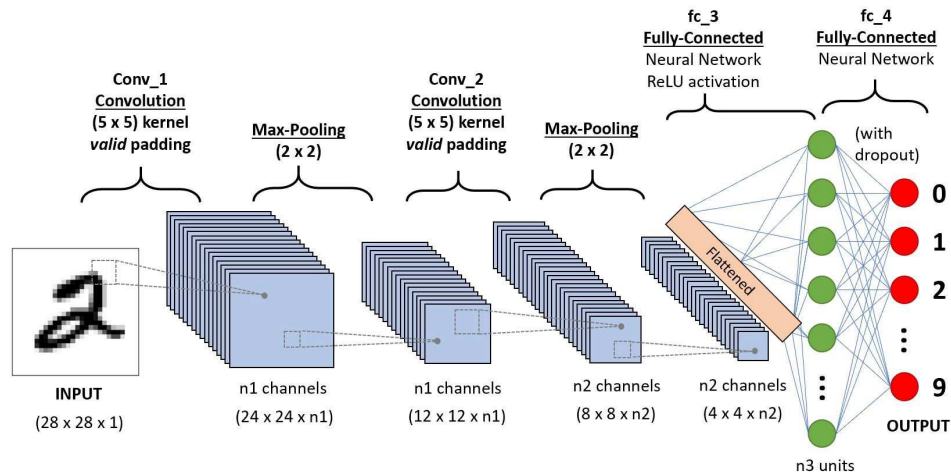


Figura 4 – Rede Neural Convolucional.

Fonte: (KENJI, 2019)

#### 2.10.1.1 Entrada

As entradas das redes neurais convolucionais, são matrizes tridimensionais, sendo duas dimensões obtidas a partir das dimensões das imagens, dado este que é normalmente redimensionado com o intuito de aumentar a eficiência na hora da classificação, e o terceiro dado sendo a profundidade de cores da imagem normalmente utilizado aspectos RGB para definição desta profundidade.

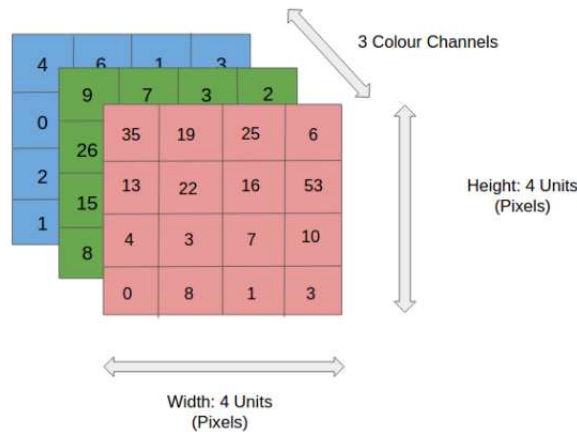


Figura 5 – Entradas Rede Neural Convolutacional.

Fonte: Deep Learning Book (DEEPLARNINGBOOK, 2022)

### 2.10.1.2 Convolução

A camada de convolução é responsável por aprender características representativas das entradas. A profundidade de saída da convolução é igual ao número de filtros aplicados para identificação, quanto mais profundas as camadas convolucionais, mais detalhados os traços identificados no mapa de ativação. O filtro, também chamado de kernel, consiste em pesos inicializados aleatoriamente que são atualizados a cada nova entrada durante o processo de *backpropagation*. A pequena área da entrada na qual o filtro é aplicado é chamada de campo receptivo.

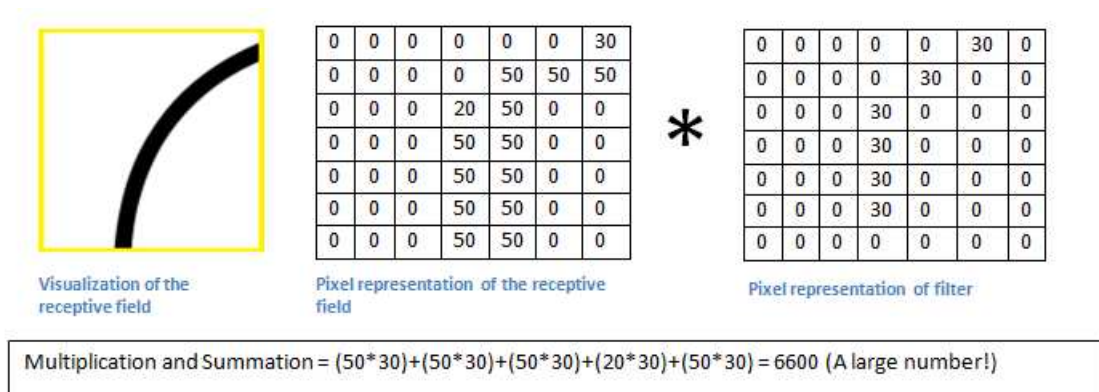


Figura 6 – Camada convolução rede neural.

Fonte: (FLORINDO, 2019)

Cada neurônio de um mapa de características é conectado de maneira bidirecional com neurônios vizinhos na camada anterior. Matematicamente, o valor da característica na localização (i, j) referente ao k-ésimo mapa de características da l-ésima camada,  $z_{i,j,k}^l$  é calculado a partir da fórmula:

$$z_{i,j,k}^l = w_k^{lT} x_{i,j}^l + b_k^l.$$

Aonde a partir da fórmula 2.10.1.2 é inferido que o valor de  $x_{i,j}^l$  representa o fragmento da entrada centrada na posição (i, j) referindo-se há l-ésima camada. Além disto, os elementos  $w_k^{lT}$  e  $b_k^l$  são o vetor de pesos e o termo de *bias*<sup>13</sup> do k-ésimo filtro da l-ésima camada, respectivamente.

### 2.10.1.3 Funções de Ativação

As funções de ativação ajudam a introduzir a não linearidade no sistema, permitindo que a rede aprenda qualquer tipo de característica da entrada da rede. Existem diversas funções de ativação, como sigmoid, tanh, softmax, etc., contudo, a função ReLU marcou um avanço, pois acelerou significativamente o tempo de treino e de inferência das redes neurais, tornando-se computacionalmente mais eficiente em comparação com outras funções além de possuir precisão semelhante, tornando-o ideal para redes convolucionais. Esta função redefine todos os valores negativos para zero da saída do nível anterior.

### 2.10.1.4 Pooling

A camada de pooling serve como simplificação das informações da camada anterior, seu objetivo é resumir a informação dos valores após a aplicação da função de ativação em um único valor. Como, por exemplo, em uma situação hipotética a saída da camada anterior após a ativação seja 24x24, a saída do pooling poderá ser 12x12. Para aplicação da camada de pooling, depende da aplicação de métodos para redução de tempo a partir do resultado obtido na camada anterior. O método mais utilizado é o maxpooling, no qual apenas o maior número da unidade é passado para a saída, método essencial principalmente para evitar situações de overfitting.

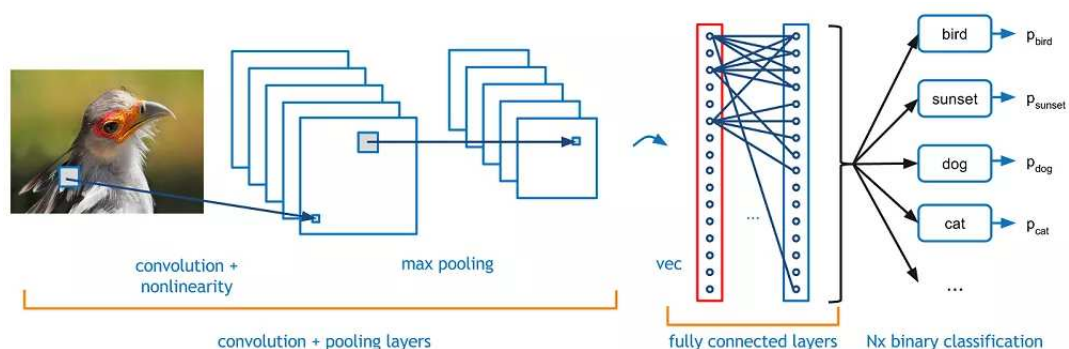


Figura 7 – Camada pooling rede neural.

**Fonte:** (LIMA; RUBIK; MORAIS, 2020)

<sup>13</sup> bias: elemento que serve para aumentar o grau de liberdade dos ajustes dos pesos.

### 2.10.2 Inception V3

O Inception V3 é uma rede neural convolucional, criada com o intuito de auxiliar a identificação de imagens de outras redes neurais. Possuindo uma acurácia em torno de 78% em cima do dataset *Imagenet*<sup>14</sup>, o Inception V3 se tornou o auge de muitas ideias desenvolvidas por vários pesquisadores ao longo dos anos.(GOOGLE, 2022f)

### 2.10.3 Mobilenet V2

Similar ao InceptionV3, o MobileNetV2 também é uma rede neural criada com o propósito de auxiliar a identificação de imagens tanto para rede neurais convolucionais quanto para qualquer algoritmo de visão computacional. Esta rede neural possui 53 camadas de aprendizado profundo baseado no banco de imagens ImageNet disponibilizado pelo Google, utilizado principalmente no propósito de identificação de redes com o propósito de identificação de objetos. (SANDLER et al., 2019)

No vigente trabalho será utilizado uma rede neural convolucional capaz de identificar a raça de um animal de estimação através de um banco de imagens com várias raças a qual a mesma utilizará como conjunto de treinamento. Processo este que será realizado junto a bibliotecas tensorflow com foco na utilização da rede neural convolucional e utilizando a linguagem python.

---

<sup>14</sup> Imagenet: dataset de 1.331.167 imagens providas pelo google para identificação de padrões em redes neurais.

## 3 Metodologia

A metodologia abordada nesse projeto de conclusão de curso é a pesquisa aplicada e bibliográfica.

Onde a pesquisa aplicada refere-se ao desenvolvimento, implantação, e entrega de um aplicativo de identificação de raça de animais domésticos e permita a interação entre os usuários que possuem animais daquela mesma raça e, em simultâneo, estejam próximos entre si.

Marina de Andrade Marconi e Eva Maria Lakatos descrevem a pesquisa bibliográfica, no livro "Fundamentos de Metodologia Científica" como:

"[...] Pesquisa bibliográfica é um tipo específico de produção científica: é feita com base em textos, como livros, artigos científicos, ensaios críticos, dicionários, enciclopédias, jornais, revistas, resenhas, resumos. Hoje, predomina entendimento de que artigos científicos constituem o foco primeiro dos pesquisadores, porque é neles que se pode encontrar conhecimento científico atualizado, de ponta. Entre os livros, distinguem-se os de leitura corrente e os de referência. Os primeiros constituem objeto de leitura reedita, realizada com detida preocupação de tomada de notas, realização de resumos, comentários, discussão, etc. Os livros de referência são livros de consulta, como dicionários, enciclopédias, relatórios de determinadas instituições, como os do Banco Central e do IBGE"(LAKATOS, 2017)

### 3.1 Materiais

Para desenvolvimento da aplicação foram utilizados o seguinte framework e linguagens de programação: Linguagem *Dart* para o desenvolvimento do *front-end* mobile híbrido, possuindo versões para android e ios aplicados junto a utilização do framework *Flutter*, e para o *back-end* foi utilizado a linguagem *Javascript* junto ao framework *Node.js* e por fim para o desenvolvimento da rede neural convolucional foi utilizado a linguagem de programação *Python* junto ao seu framework *Fastapi*.



Sendo para o aplicativo front-end utilizado as seguintes bibliotecas:

- *riverpod* (2.1.1): biblioteca responsável por gerenciamento de *cache*<sup>1</sup> e declaração de *providers*<sup>2</sup>, facilitando a implementação de requisições *http*<sup>3</sup> e implementação de arquiteturas (RIVERPOD, 2022). Biblioteca a qual foi analisado a integração, desenvolvimento e vantagens de utilização, segundo o livro (SINHA, 2021).
- *firebase\_core* (2.1.0): biblioteca que permite a conexão do aplicativo móvel com o painel firebase da Google de maneira simples (FIREBASE, 2022).
- *dio* (4.0.6): biblioteca que permite realizar requisições utilizando o protocolo *HTTP*<sup>4</sup> (DIO, 2022).
- *firebase\_auth* (4.1.0): biblioteca que permite a conexão do aplicativo móvel com o recurso de autenticação do firebase.(AUTH, 2022).
- *cloud\_storage* (4.0.4): biblioteca que permite a conexão do aplicativo móvel com o recurso de armazenamento firestore do firebase.(FIRESTORE, 2022).
- *firebase\_storage* (11.0.3): biblioteca que permite a conexão do aplicativo móvel com o recurso de armazenamento de imagens.(STORAGE, 2022).
- *google\_sign\_in* (5.4.2): biblioteca que permite a autenticação do aplicativo com provedor do Google.(GOOGLE, 2022d).
- *geolocator* (9.0.2): biblioteca que permite obter a localização do dispositivo de maneira simplificada (GEOLOCATOR, 2022).
- *image\_picker* (0.8.6): biblioteca que permite o acesso a câmera e a galeria do dispositivo de maneira rápida.(PICKER, 2022).
- *firebase\_messaging* (2.0.0): biblioteca que permite o disparo de notificações via painel do firebase.(MESSAGING, 2022).

---

<sup>1</sup> Cache: sistema de armazenamento interno de um sistema cujo objetivo é acelerar serviços e processos anteriormente armazenados.

<sup>2</sup> Providers: pacote interno nativo na "componentização" do framework cujo objetivo é facilitar a utilização e gerência do framework.

<sup>3</sup> Http: *Hyper Text Tranference Protocol*, protocolo utilizado para troca de informação entre serviços e aplicações na internet

<sup>4</sup> HTTP: protocolo de transferência de hipertexto.

Para o sistema *back-end* foram utilizados as seguintes bibliotecas:

- *firebase-functions* (3.18.0): biblioteca que permite a conexão do aplicativo móvel com o recurso de autenticação do firebase.([FUNCTIONS, 2022](#)).
- *axios* (0.27.2): biblioteca que permite a realização de requisições HTTP utilizando Javascript.([AXIOS, 2022](#)).
- *express* (4.18.2): biblioteca que proporciona ferramentas para criação de servidores utilizando o protocolo TCP, HTTP ou Socket.([STRONGLOOP/IBM, 2017](#)).
- *firebase-admin* (10.0.2): biblioteca que proporciona a conexão com o painel do firebase.([ADMIN, 2022](#)).
- *fsm* (2.1.3): biblioteca que proporciona a criação de Máquinas de estado finito utilizando Javascript.([FSM, 2022](#)).
- *faker* (7.6.0): biblioteca que proporciona a geração de dados randômicos, sejam estes nome, cidade, CPF entre outros.([FAKER, 2022](#)).
- *k-means* (2.1.3): biblioteca que proporciona a implementação do algoritmo K-means na linguagem Javascript.([KMEANS, 2022](#)).

Por fim, para o treinamento da *rede neural* foram utilizadas as seguintes bibliotecas:

- *fastapi*: framework feito a partir da linguagem python para criação de servidores utilizando o protocolo HTTP.([FASTAPI, 2022](#)).
- *tensorflow*: conjunto de bibliotecas para trabalhar com algoritmos de inteligência artificial utilizando a linguagem Python.([TENSORFLOW, 2022](#)).
- *numpy*: principal biblioteca do mercado para trabalhar com computação científica, proporcionando diferentes ferramentas para trabalhos com números e geometria.([NUMPY, 2022](#)).

Para o armazenamento das imagens e gerenciamento dos dados, será utilizado a plataforma de desenvolvimento de aplicações Firebase ([GOOGLE, 2022b](#)), por ser uma plataforma que possui integração nativa com a nova versão do framework Flutter, e possuir serviços clouds robustos e consolidados no mercado, permitindo um alto fluxo de dados sem prejudicar o desempenho da aplicação, o mesmo se mostrou como uma solução viável para o desenvolvimento deste trabalho. Tendo como base o livro "Beginning Flutter® (A Hands On Guide To App Development) || Adding the Firebase and Firestore Backend"([NAPOLI, 2019](#)), aprofundado a integração entre às duas plataformas, analisando as vantagens e

desvantagens na utilização do mesmo, em comparação a outros serviços cloud e backend próprios.

Para a identificação da raça do animal utilizando rede neural, será utilizada a biblioteca Tensorflow integrada a linguagem de programação Python para realizar o treinamento da mesma, posteriormente gerando um arquivo *.h5*, a qual será adicionado a uma api construída com o framework Fastapi, com o intuito de realizar o processamento das imagens enviadas via requisição POST para identificação da raça do animal. Tendo a partir do livro "Mobile Deep Learning with TensorFlow Lite, ML Kit and Flutter: Build scalable real-world projects to implement end-to-end neural network on Android and iOS"(SINGH; BHADANI, 2020), o aprofundamento da técnica utilizada e impacto da aplicação de uma rede neural em aplicações Android e IOS.

Para a aplicação do algoritmo de geolocalização do usuário e para realizar a filtragem dos animais de estimação dos usuários, será utilizado o serviço de Cloud Functions ofertado pelo Firebase, junto com a biblioteca de geolocalização, a qual tem o intuito de prover funções via *APIS*<sup>5</sup>, utilizando o protocolo de comunicação http sem a necessidade de um backend próprio para o mesmo (GOOGLE, 2022a). Por ser um serviço ofertado pelo Firebase, é possível interagir com os demais serviços ofertados, assim podendo acessar os dados armazenados pelos usuários para aplicar os algoritmos de geolocalização sobre os mesmos.

### 3.1.1 Ferramentas utilizadas para a pesquisa

Nessa seção será apresentado as ferramentas utilizadas nessa pesquisa:

- Computador utilizado: *Macbook Pro 2020 13,3 processador M1, modelo A2338*.
- *Visual Studio Code 1.17.1* para codificação do aplicativo utilizando o framework flutter.
- *Postman* para testar as requisições HTTP ofertadas pelas clouds functions.
- *PyCharm 2022.2.4* para codificação e treinamento da rede neural para identificação das raças dos animais.
- *Firebase* para armazenagem dos dados e disponibilizar as clouds functions para consulta dos algoritmos.
- *Online Visual Paradigm* para criação dos diagramas relacionados ao aplicativo.
- *Figma* para criação de protótipos de telas do aplicativo.

---

<sup>5</sup> API: *Application Programming Interface*, são conjuntos de padrões e rotinas utilizadas para comunicação entre sistemas.

- *Whimsical* para criação dos fluxogramas da implementação da rede neural.
- *SonarQube* para testes de qualidade do código e revisão da aplicação.
- *Netron* para visualização das camadas e parâmetros da rede neural.

## 3.2 Funcionamento da Aplicação

Com o intuito de solucionar a problemática apresentada na justificativa, foi tomada a decisão da criação de um aplicativo móvel que proporciona aos donos de animais de estimação, devido ao fato que 96% da população brasileira utiliza dispositivos móveis para acesso à internet e comunicação, segundo o site do IBGE em 2019 ([IBGE, 2019b](#)). O aplicativo desenvolvido terá o propósito de ser uma ferramenta que auxilie os mesmos em uma eventual perda de seu animal, ofertando recursos como localização do mesmo e comunicação entre os donos dos animais.

Para utilizar o aplicativo, o usuário poderá ou não se registrar, pois o aplicativo permitirá que usuário sem conta mande uma foto de um animal perdido, e após a identificação, será alertado ao dono do animal via notificação e uma mensagem do próprio sistema alertando aonde foi encontrado o animal e se o animal encontrado pertence ao dono notificado. Porém, para acessar funcionalidades respectivas a comunicação e recepção de notificações o usuário precisará estar autenticado, para poder se comunicar com outros donos, cadastrar seus animais e receber as mensagens caso sejam encontrados em situações de desaparecimento.

Para desenvolver as funcionalidades de localização e identificação do animal no aplicativo, será necessário o usuário ter acesso à internet e permitir que o aplicativo tenha acesso à localização do dispositivo, para cadastro do sistema. Após o Registro, o usuário deverá realizar o cadastro de um animal de estimação, onde no momento do envio da foto do animal para o cadastro, o próprio sistema identificará a raça, que será a base para filtros e identificações caso o animal venha a se perder e receba as notificações e mensagens do próprio sistema. É válido ressaltar, que o sistema de disparos de notificação e envio de mensagens, é ofertado pelo próprio Firebase, facilitando assim o desenvolvimento esta aplicação ([GOOGLE, 2022b](#)).

Além do funcionamento principal do aplicativo descrito acima, o usuário poderá impedir que o aplicativo mostre seu perfil para outros usuários, desta forma ele apenas receberá mensagens do próprio sistema, porém não poderá ser contatado por outros usuários para trocas de informação. As demais funcionalidades essenciais ao aplicativo, foram desenvolvidas utilizando recursos ofertados pelo próprio Firebase, como disparo de notificação pelo *Firebase messaging* ([MESSAGING, 2022](#)), mensagens entre os usuários

através do *Firebase Firestore* ([FIRESTORE, 2022](#)), e armazenamento de arquivos ofertado pelo *Firebase Storage* ([STORAGE, 2022](#)).

Para as funcionalidades de identificação do animal, foi treinada uma rede neural convolucional, que possui o objetivo de identificar a raça do animal para fins de cadastro e posteriormente aviso aos respectivos donos, e para os testes de filtragem e notificação dos donos de animais, foram gerados dados fictícios com o intuito de aplicar um algoritmo de agrupamento(K-Means), e facilitar a identificação de um possível dono para um animal perdido em determinada localização, funcionalidades estas melhores descritas no decorrer deste trabalho.

### 3.3 Banco de imagens utilizado no treinamento

Para realizar o treinamento e identificações dos animais a partir da foto tirada pelo aplicativo, foi utilizado um *dataset* disponível no site Kaggle ([KAGGLE, 2022](#)). Sobre o dataset em questão, a coleta e separação das imagens foi realizada por alunos da universidade de Stanford ([KHOSLA et al., 2011](#)), contando com 20581 (vinte mil quinhentos e oitenta e uma) imagens e 120 raças separadas conforme a presença destes cachorros ao redor do mundo, ao realizar o download é possível se deparar com um arquivo *csv*<sup>6</sup>.

---

<sup>6</sup> csv: Comma-separated values

Devido à quantidade extensa de dados, a visualização foi dividida em 3 partes, abaixo é visível o primeiro grupo de raças.

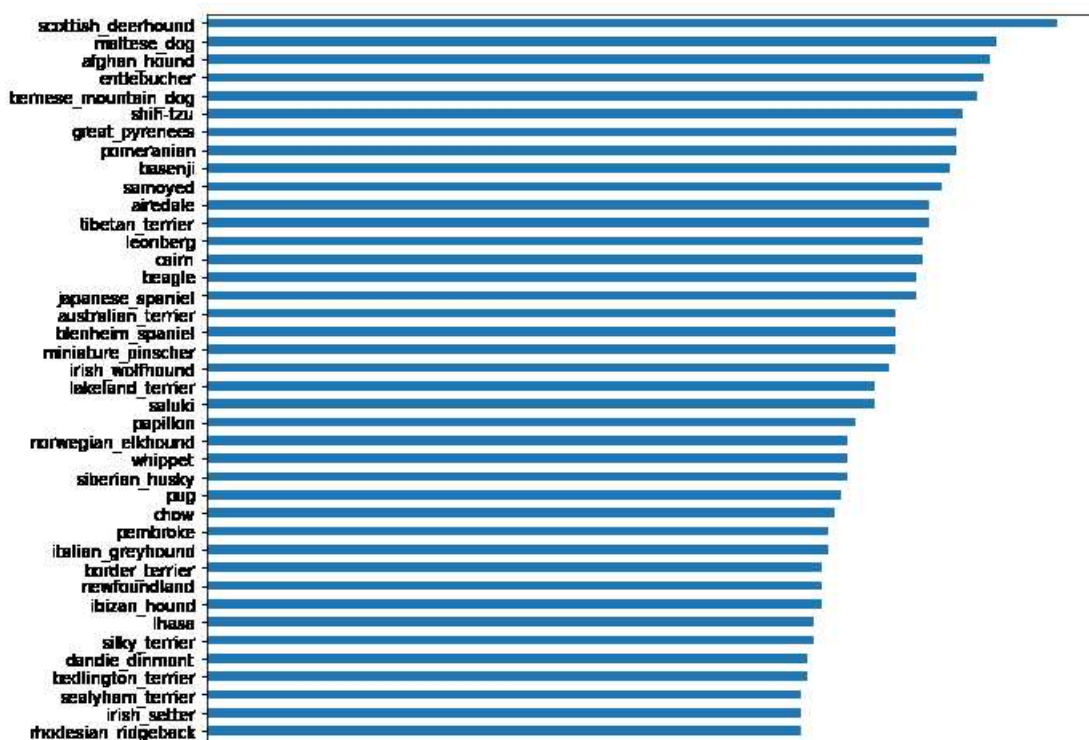


Figura 8 – Raças do dataset 1.

Fonte: Autoria Própria (2022).

Segundo grupo de raças.

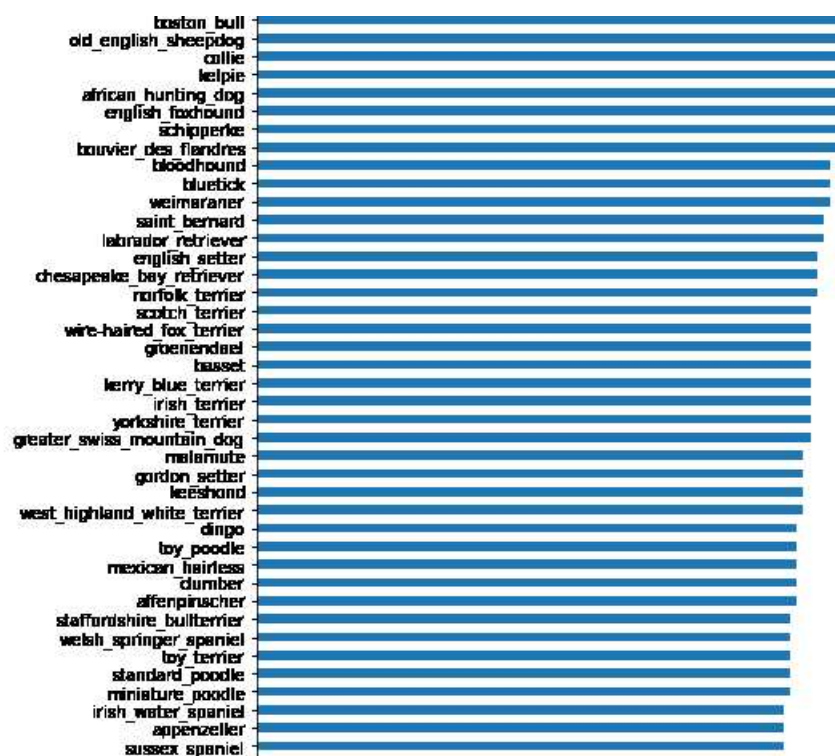


Figura 9 – Raças do dataset 2.

Fonte: Autoria Própria (2022).

Terceiro grupo de raças.

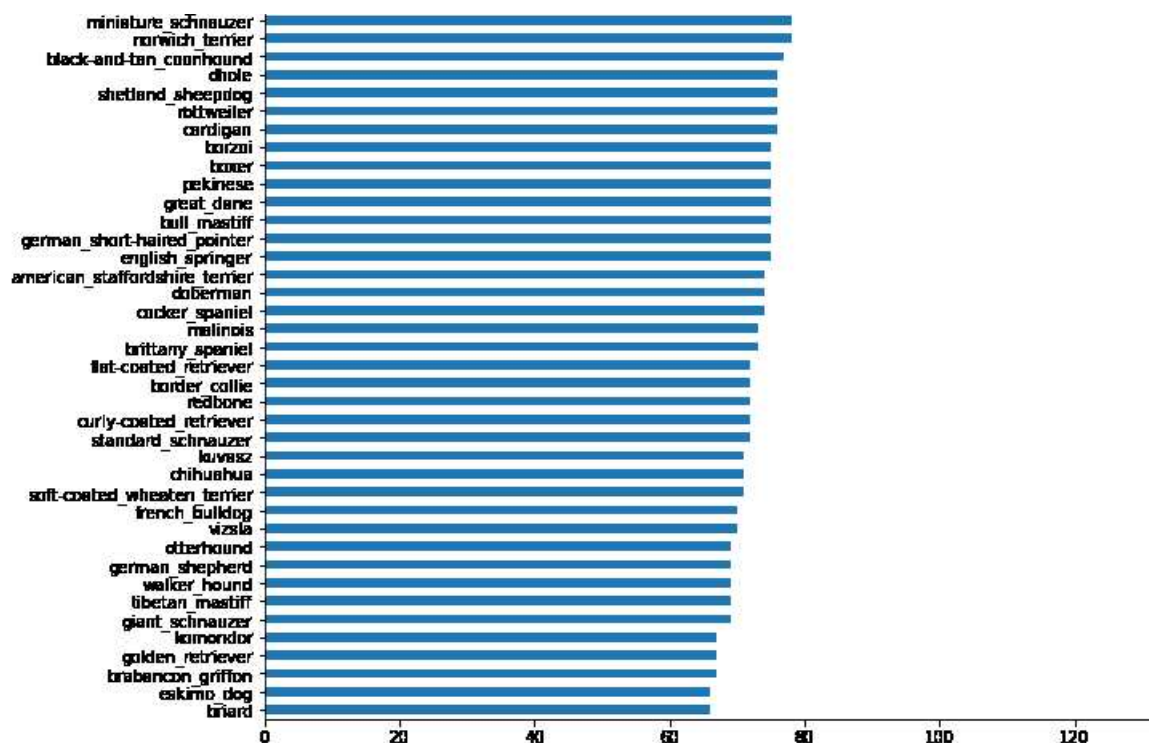


Figura 10 – Raças do dataset 3.

Fonte: Autoria Própria (2022).

Conforme demonstrado acima, é possível ver a distribuição das raças a partir do nome e a quantidade de imagens para cada uma das raças apresentadas. Sendo a maior quantidade disponível para a raça *scottish\_deerhounded*, e a menor para a raça *briard*.



### 3.4 Treinamento das redes neurais

Com o intuito de maximizar o desempenho ofertado na solução da problemática envolvendo animais perdidos, foram treinadas 4 (quatro) redes neurais utilizando diferentes camadas e técnicas para validação e identificação das imagens disponibilizadas no banco de imagens, de modo a encontrar a melhor solução possível para a problemática.

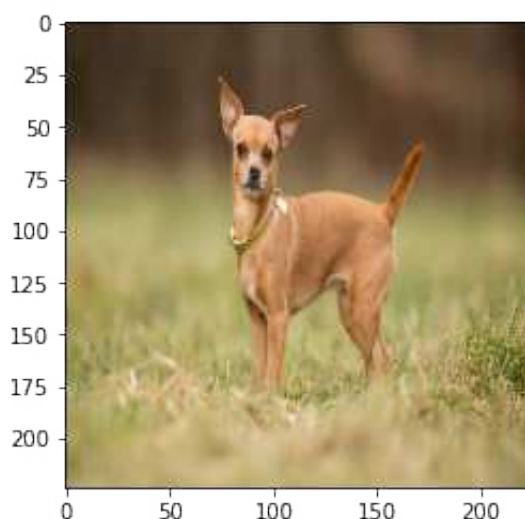


Figura 11 – Entrada da rede neural.

**Fonte:** Autoria Própria (2022).

Todas as redes neurais treinadas receberam o mesmo input, sendo este uma imagem de proporção (224x224), e sua profundidade sendo o valor respectivo na matriz RGB da mesma, como demonstrado na figura acima.

Redes Neurais	Camadas Ocultas	Porcentagem Validação	Porcentagem Teste
1	33.453	25%	25%
2	33.453	25%	25%
3	33.451	25%	25%
4	33.451	25%	25%

Para treinamento das redes neurais foram utilizados 25% para teste e validação, além de 50% do dataset dedicado apenas para o treinamento.

### 3.4.1 Primeira rede neural

A primeira rede neural descrita abaixo, foi desenvolvida com o intuito de maximizar o número de camadas de convolução, aumentando cada vez mais o número de parâmetros provenientes da camada anterior, sem realizar o reajuste de pesos, com o intuito de maximizar a obtenção de detalhes, reajustando o input da camada anterior conforme a imagem é passada por cada neurônio.

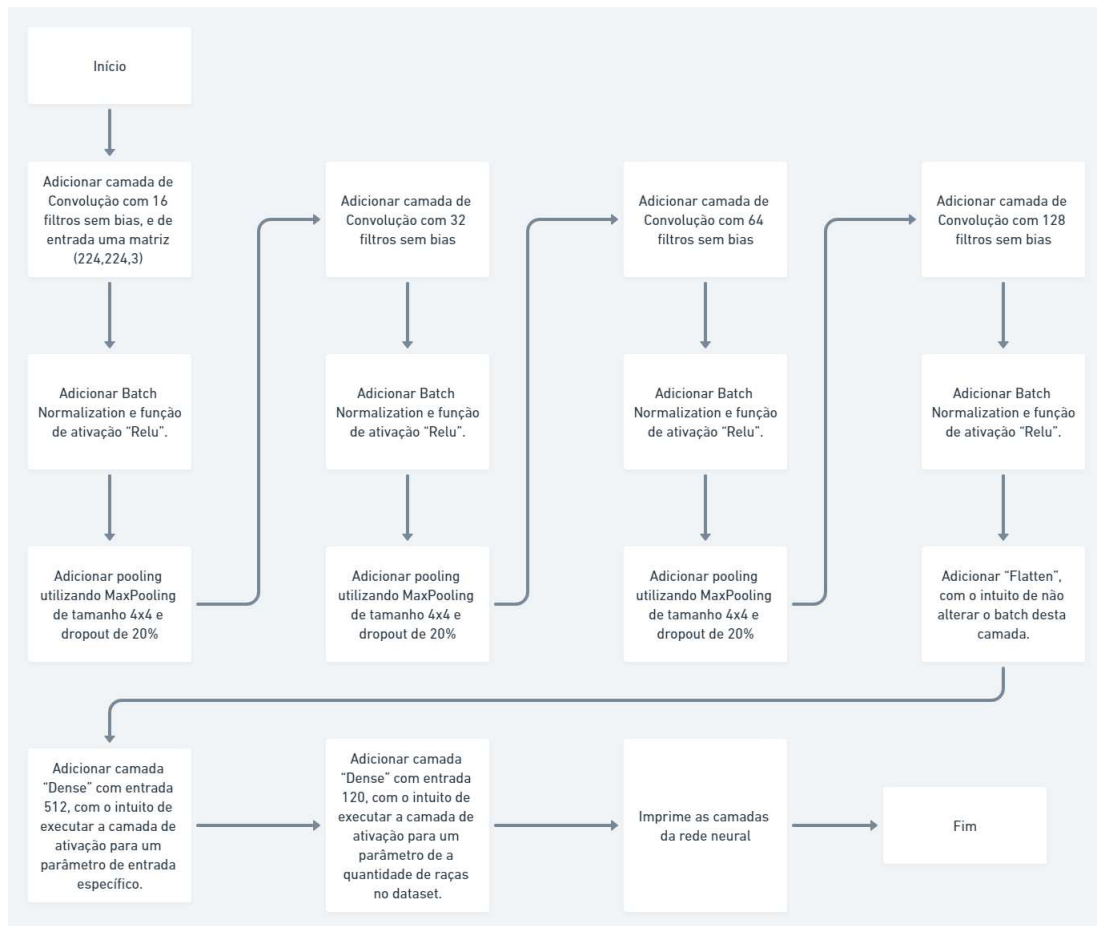


Figura 12 – Fluxograma Implementação da primeira rede neural.

**Fonte:** Autoria Própria (2022).

Tendo como impressão a seguinte configuração mostrado na figura abaixo.

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 224, 224, 16)	432
batch_normalization (Batch Normalization)	(None, 224, 224, 16)	48
activation (Activation)	(None, 224, 224, 16)	0
max_pooling2d (MaxPooling2D)	(None, 56, 56, 16)	0
dropout (Dropout)	(None, 56, 56, 16)	0
conv2d_1 (Conv2D)	(None, 56, 56, 32)	4608
batch_normalization_1 (Batch Normalization)	(None, 56, 56, 32)	96
activation_1 (Activation)	(None, 56, 56, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
dropout_1 (Dropout)	(None, 14, 14, 32)	0
conv2d_2 (Conv2D)	(None, 14, 14, 64)	18432
batch_normalization_2 (Batch Normalization)	(None, 14, 14, 64)	192
activation_2 (Activation)	(None, 14, 14, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 64)	0
dropout_2 (Dropout)	(None, 4, 4, 64)	0
conv2d_3 (Conv2D)	(None, 4, 4, 128)	73728
batch_normalization_3 (Batch Normalization)	(None, 4, 4, 128)	384
activation_3 (Activation)	(None, 4, 4, 128)	0
flatten (Flatten)	(None, 2048)	0
dropout_3 (Dropout)	(None, 2048)	0
dense (Dense)	(None, 512)	1049088
dense_1 (Dense)	(None, 120)	61560

```

Total params: 1,208,568
Trainable params: 1,208,088

```

Figura 13 – Fluxograma Implementação da primeira rede neural.

**Fonte:** Autoria Própria (2022).

Como pode ser visto na Figura 14, para cada camada de convolução a entrada é aumentada conforme o parâmetro da camada anterior, ativando a função de ativação "Relu", para cada camada de convolução adicionada.

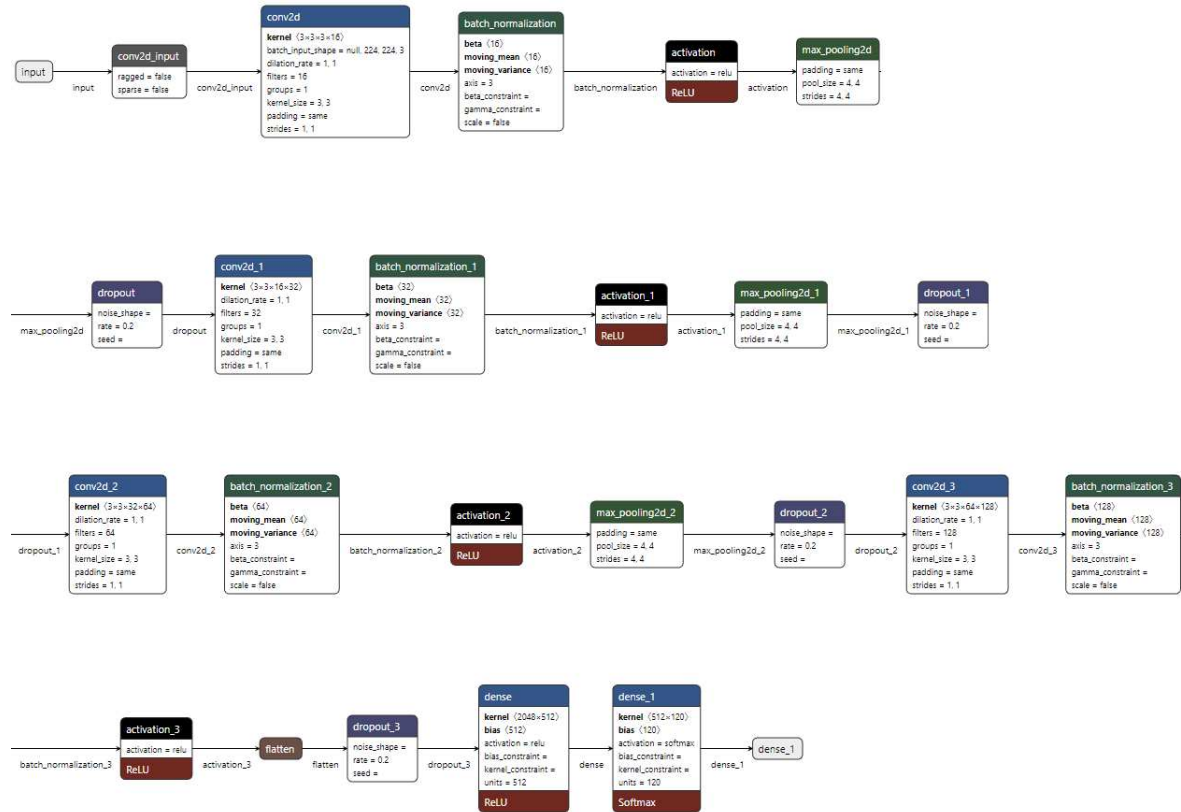


Figura 14 – Camadas da primeira rede neural via Netron.

**Fonte:** Autoria Própria (2022).

Por fim, para manter a linearidade da rede neural, foi adicionado uma camada "Flatten" com o intuito de manter o resultado da função de ativação anterior, repassado para a camada mais Densa, com um parâmetro fixo de 512x512 de entrada, maximizando a quantidade de píxeis e saída da rede neural maximizando ao máximo a eficiência.

### 3.4.2 Segunda rede neural

A segunda rede neural demonstrada abaixo, foi desenvolvida utilizando a estratégia de reajuste de pesos, porém, não foi incrementado a entrada de cada camada de convolução nem utilizado um pooling maior, devido ao risco de *overfitting*<sup>7</sup> para esta rede e minimizar o tempo de treinamento posteriormente relatado no capítulo de resultados. Contudo, a preservação das camadas anteriores e maximização da quantidade de píxeis analisados utilizando uma camada mais densa ainda foi mantido.

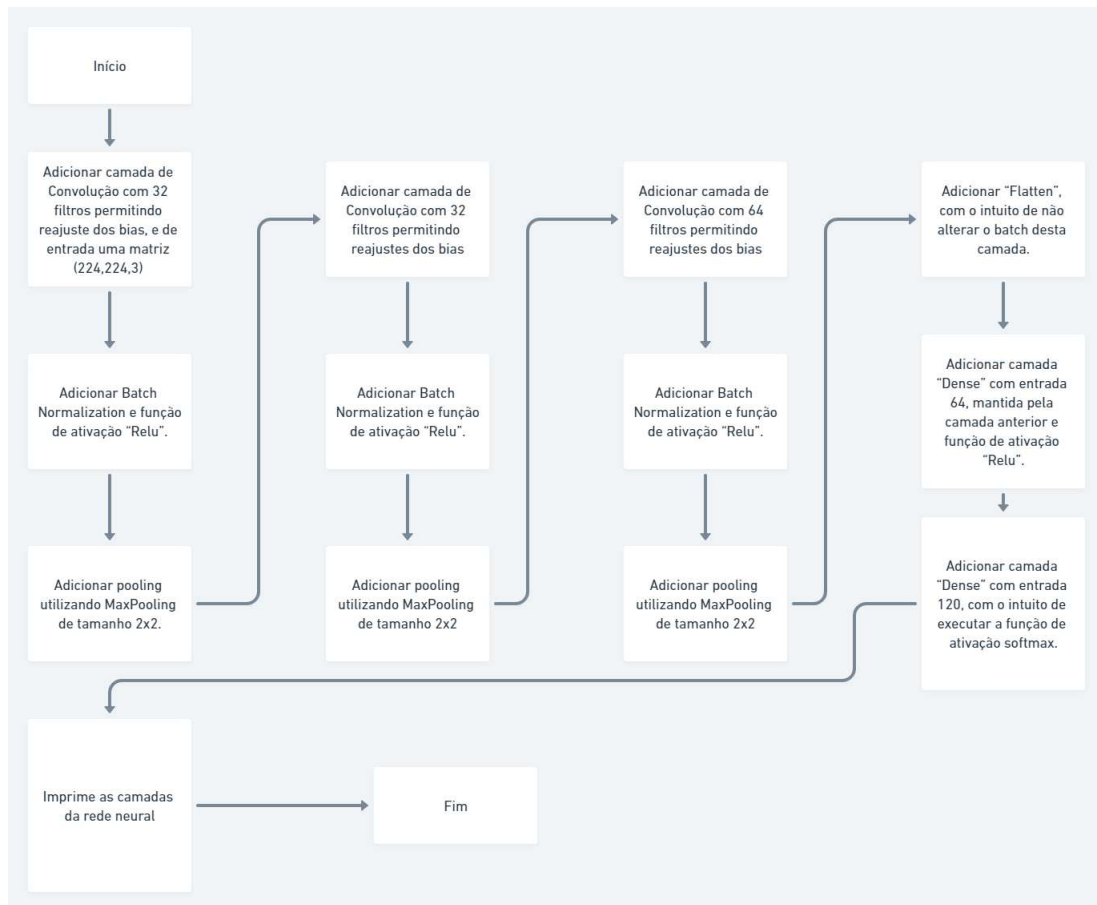


Figura 15 – Fluxograma Implementação da segunda rede neural.

**Fonte:** Autoria Própria (2022).

<sup>7</sup> overfitting: quando um modelo estatístico se ajusta muito bem ao conjunto de dados anteriormente observado, mas se mostra ineficaz para prever novos resultados.

Tendo sua configuração demonstrada na figura abaixo.

Model: "sequential\_1"

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 222, 222, 32)	896
activation_4 (Activation)	(None, 222, 222, 32)	0
max_pooling2d_3 (MaxPooling 2D)	(None, 111, 111, 32)	0
conv2d_5 (Conv2D)	(None, 109, 109, 32)	9248
activation_5 (Activation)	(None, 109, 109, 32)	0
max_pooling2d_4 (MaxPooling 2D)	(None, 54, 54, 32)	0
conv2d_6 (Conv2D)	(None, 52, 52, 64)	18496
activation_6 (Activation)	(None, 52, 52, 64)	0
max_pooling2d_5 (MaxPooling 2D)	(None, 26, 26, 64)	0
flatten_1 (Flatten)	(None, 43264)	0
dense_2 (Dense)	(None, 64)	2768960
activation_7 (Activation)	(None, 64)	0
dropout_4 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 120)	7800
activation_8 (Activation)	(None, 120)	0
Total params: 2,805,400		
Trainable params: 2,805,400		
Non-trainable params: 0		

Figura 16 – Impressão da configuração da segunda rede neural.

**Fonte:** Autoria Própria (2022).

Como pode ser visto na Figura 17, é mantida a entrada da terceira camada de convolução para a camada "Dense", utilizando a função de ativação "SoftMax".

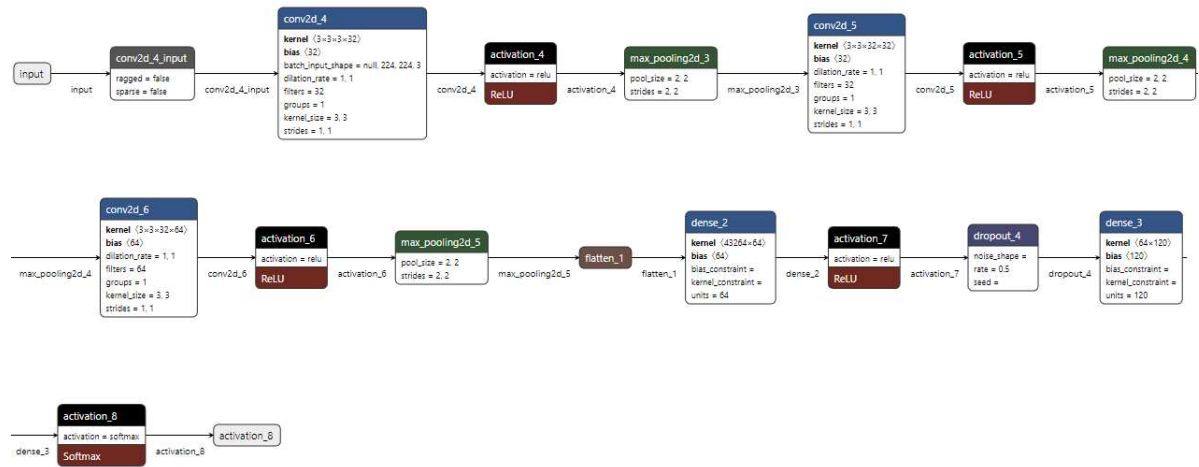


Figura 17 – Camadas da segunda rede neural via Netron.

**Fonte:** Autoria Própria (2022).

Tal ação é tomada no intuito de aumentar a eficiência da rede treinada através da função de ativação, além de criar a matriz de probabilidade a partir da média obtida em cada entrada da matriz

### 3.4.3 Terceira rede neural (MobileNetV2)

A terceira rede neural demonstrada abaixo, foi desenvolvida utilizando o auxílio da rede neural *MobileNetV2*, com o intuito de melhorar o desempenho em comparação ao resultado das redes neurais anteriores, sendo este demonstrado no capítulo de resultados. Foi constatado que ao utilizar os pesos e parâmetros desta rede neural, foi perceptível um aumento significativo na eficiência da rede neural, se tornando até o momento a rede com o melhor desempenho para aplicação da solução à problemática.

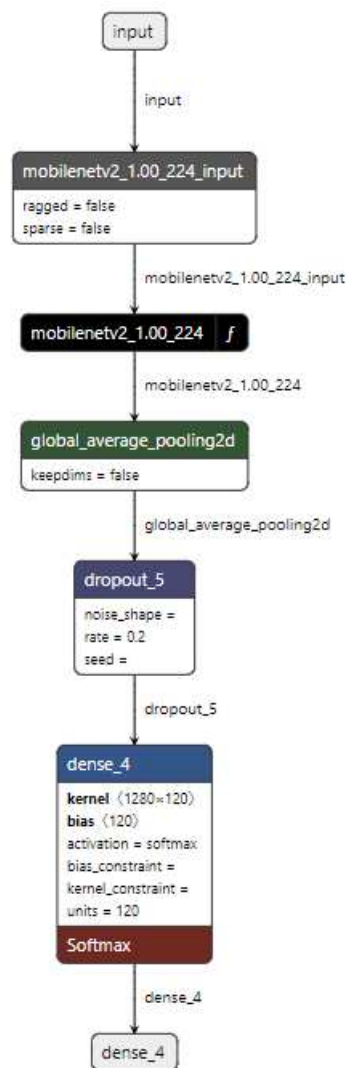


Figura 18 – Fluxograma Implementação da terceira rede neural.

**Fonte:** Autoria Própria (2022).



Tendo sua configuração demonstrada na figura abaixo.

```
Model: "sequential_2"
```

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, None, None, 1280)	2257984
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dropout_5 (Dropout)	(None, 1280)	0
dense_4 (Dense)	(None, 120)	153720

```
=====  
Total params: 2,411,704  
Trainable params: 153,720  
Non-trainable params: 2,257,984  
=====
```

Figura 19 – Impressão da configuração da terceira rede neural.

**Fonte:** Autoria Própria (2022).

Como demonstrado abaixo, o primeiro parâmetro para a rede é o *MobileNetV2*, executada na sequência a sua entrada no contexto geral da rede, com o intuito de obter os resultados e auxiliar no treinamento da rede neural que busca identificar as raças do cachorro.

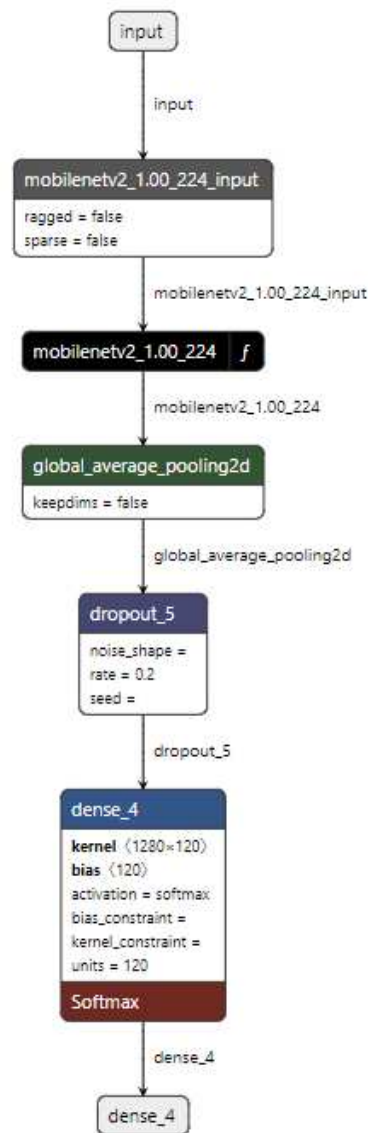


Figura 20 – Camadas da segunda rede neural via Netron.

**Fonte:** Autoria Própria (2022).

Apesar da quantidade de inferior de camadas na rede neural, é nítido que a utilização deste modelo aumentou muito a eficiência obtida na rede.

#### 3.4.4 Quarta rede neural (InceptionV3)

A quarta rede neural demonstrada abaixo, foi desenvolvida utilizando o auxílio da rede neural *InceptionV3*, com o intuito de melhorar o desempenho em comparação ao resultado das redes neurais anteriores, sendo este demonstrado no capítulo de resultados, diferente da rede neural *MobileNetV2*, a *InceptionV3* é possui mais imagens de parâmetro e mais ajustes de pesos na sua função para o treinamento. Foi constatado que ao utilizar os pesos e parâmetros desta rede neural, foi relatado um aumento considerável inclusive em relação à rede neural anterior, sendo esta a rede escolhida para ser a solução do aplicativo.

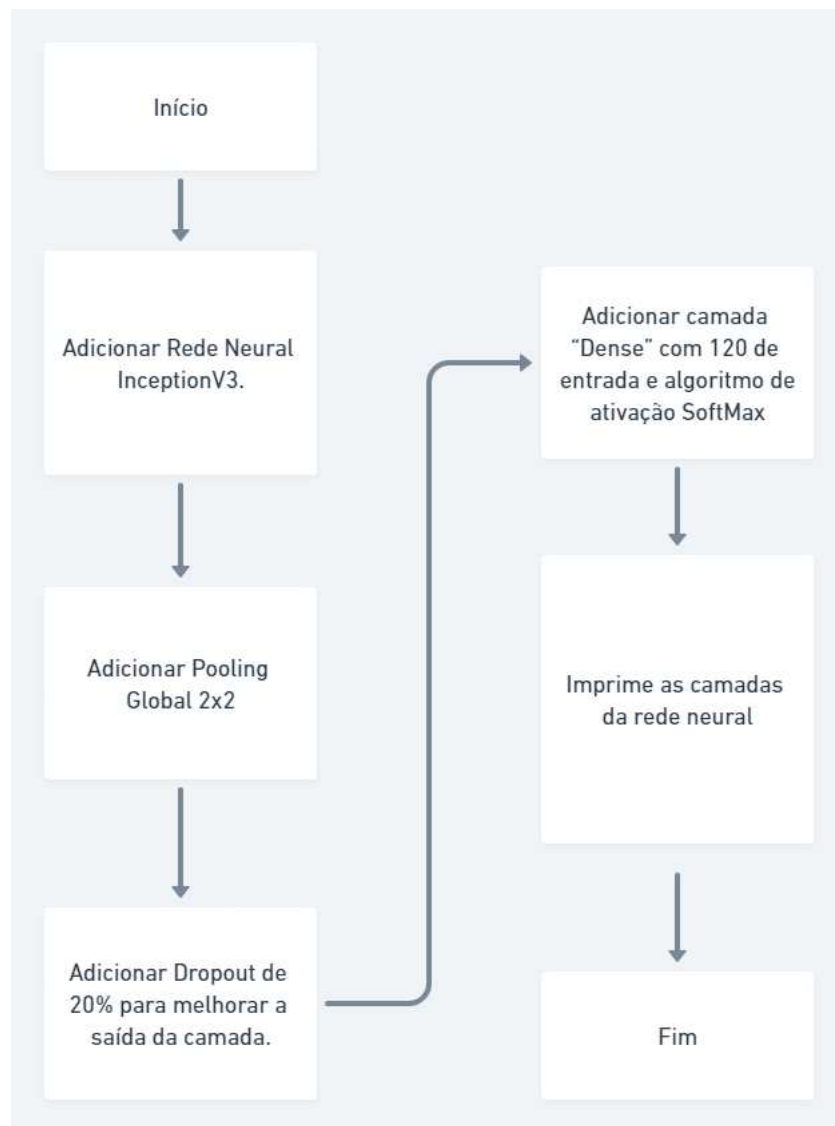


Figura 21 – Fluxograma Implementação da quarta rede neural.

**Fonte:** Autoria Própria (2022).

Tendo sua configuração demonstrada na figura abaixo.

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
inception_v3 (Functional)	(None, None, None, 2048)	21802784
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dropout (Dropout)	(None, 2048)	0
dense (Dense)	(None, 120)	245880

```
Total params: 22,048,664  
Trainable params: 245,880  
Non-trainable params: 21,802,784
```

Figura 22 – Impressão da configuração da quarta rede neural.

**Fonte:** Autoria Própria (2022).

Como demonstrado abaixo, a aplicação da rede "inception" é similar a rede *MobileNetV2*, executada na sequência a sua entrada no contexto geral da rede, com o intuito de obter os resultados e auxiliar no treinamento da rede neural que busca identificar as raças do cachorro.

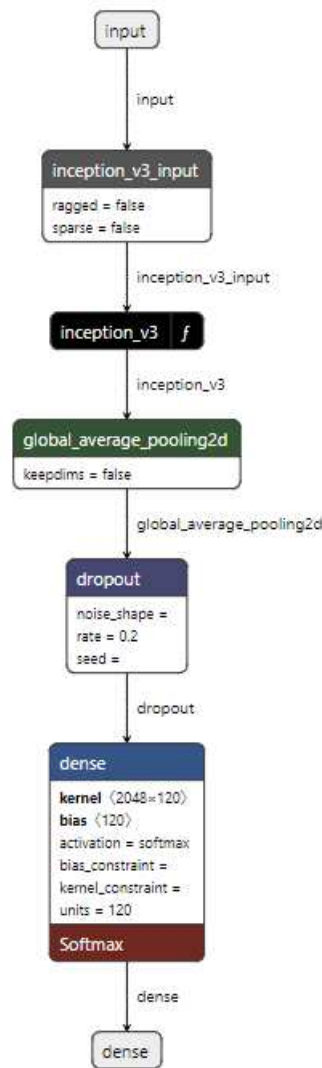


Figura 23 – Camadas da quarta rede neural via Netron.

**Fonte:** Autoria Própria (2022).

Porém, por ser uma rede mais complexa, utilizando mais imagens para sua validação, a mesma teve uma eficácia maior e mais assertiva em relação à *MobileNetV2*.

## 3.5 Agrupamento dos usuários utilizando K-means

Com o intuito de localizar os donos mais próximos de um animal encontrado na rua, foi utilizado o algoritmo de aprendizado de máquina não-supervisionado K-means, os usuários no aplicativo foram agrupados conforme o parâmetro de proximidade dos usuários entre si, deste modo a partir do momento que um animal é encontrado, ao invés de realizar a verificação entre cada usuário, é verificado o grupo na totalidade apenas realizando o filtro no mesmo para encontrar um dono que perdeu um animal com as características analisadas pela rede neural.

### 3.5.1 Geração sintética dos dados para agrupamento dos usuários

Com a intenção de proporcionar uma identificação mais assertiva no funcionamento do módulo de comunicação aplicativo, foram gerados de dados sintéticos dos donos de animais, com o intuito de validar o algoritmo de máquina de estado finito que proporciona disparo de notificação e iniciar sistema de conversa entre usuários. Abaixo, temos o algoritmo utilizado na geração das coordenadas dos usuários, feito neste algoritmo uma limitação de área da cidade de Palmas no estado do Tocantins, onde os usuários gerados devem pertencer a esta área limitada, onde para cada ponto randômico dentro da área deste polígono, é gerado um novo usuário com dados falsos utilizando a biblioteca *faker*, da linguagem *Javascript*, que gera dados de usuários de maneira randômica para testes.

---

**Algoritmo 1:** Geração das coordenadas para usuários fictícios

---

**input** : Polígono que limita a área onde os usuários podem ser gerados

**output** : Geração dos dados dos usuários fictícios

**data** : Vetor com a lista de coordenadas

**param** : Número de usuários que devem ser gerados

```
1 poligono  $\leftarrow$  Poligono([vértices]);  
2 ponto  $\leftarrow$  Ponto([vértices]);  
3 pontoAleatorio = Ponto([random.uniform(latitude, longitude)])  
   faker = Faker()  
  
4 while data.length < param do  
5   | if pontoAleatorio  $\in$  poligono then  
6   |   | /*remove-o da lista de usuários da sala*/  
   |   | coordenadas.adicionar(pontoAleatorio);  
7   | end  
8 end  
  
9 for coordenada  $\in$  coordenadas do  
10  | usuarios.adicionar(Usuario(faker.name, faker.animais, pontoAleatorio));  
11 end  
12 usuarios.salvar();
```

---

A partir do algoritmo exibido acima, foram gerados 40 usuários aleatórios, onde a posição dos mesmos está em um polígono que possui a lista de vértices limitadas a área de Palmas, como mostrado na figura abaixo.

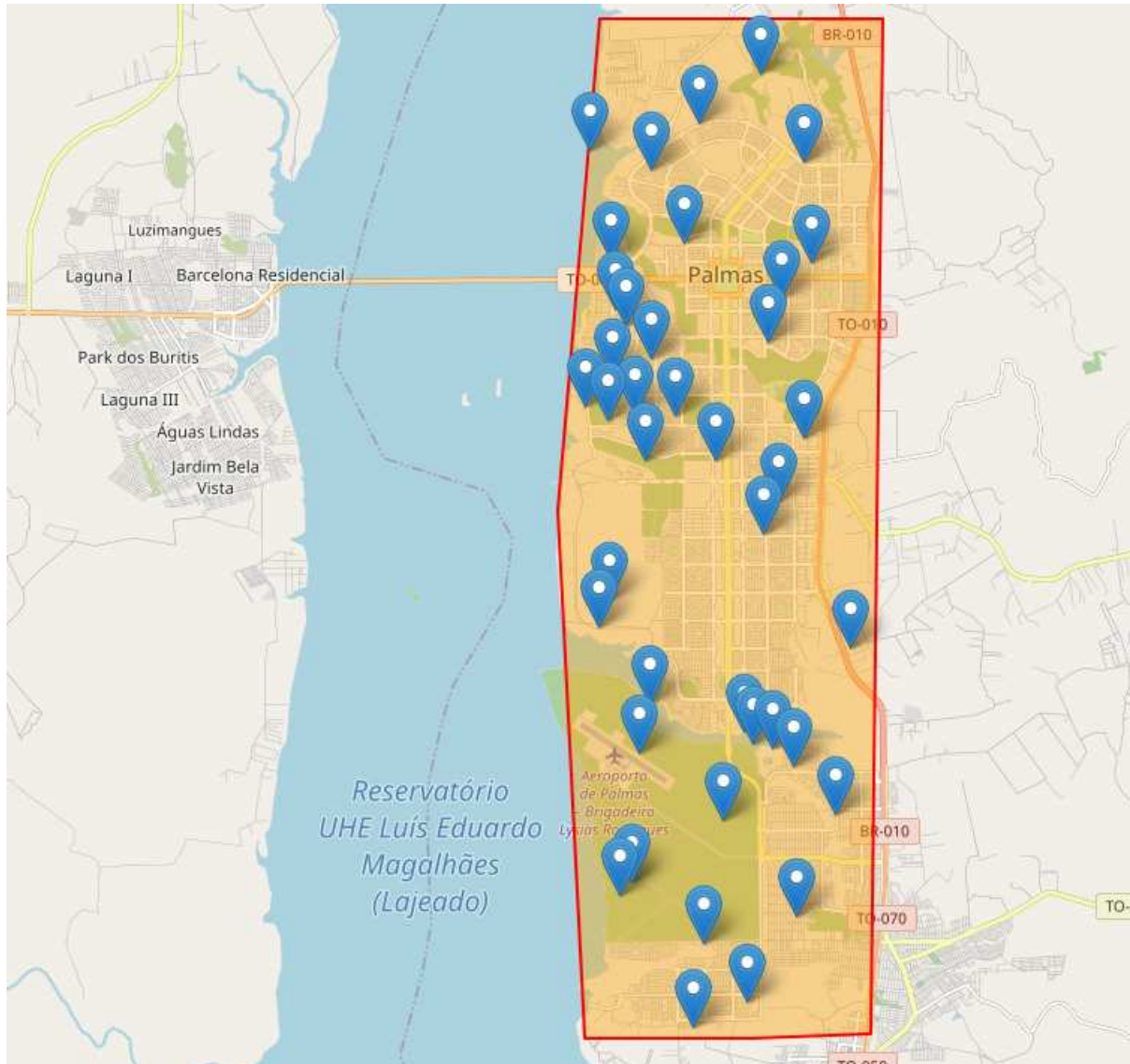


Figura 24 – Disposição dos usuários fictícios no mapa de Palmas.

**Fonte:** Autoria Própria (2022).

Posteriormente estes usuários serão agrupados por proximidade no intuito de realizar os filtros e notificações dos mesmos em situação de localização dos animais perdidos.



### 3.5.2 Elbow method

O método elbow é baseado na ideia de que a melhor solução de agrupamento é aquela que produz a maior variação explicada. Quando o número de clusters aumenta, a variação explicada também aumenta, até um certo ponto em que aumentar o número de clusters não produz nenhum benefício adicional. Esse ponto é tomado como o melhor número de clusters a utilizar, como descrito no trabalho dos acadêmicos de engenharia elétrica Hestry Humaira e Rasyidah Rasyidah no departamento de tecnologia do campo de Unand na Índia, usado como base para validação do número de clusters gerados (HUMAIRA; RASYIDAH, 2020). A figura abaixo mostra a validação realizada nos dados fictícios gerados, com diferentes números de clusters, e como é possível visualizar, a partir do terceiro e quarto número de clusters é notável uma tendência de estabilidade no algoritmo, onde quanto mais clusters forem adicionados, menos relevante será a aplicação do algoritmo.

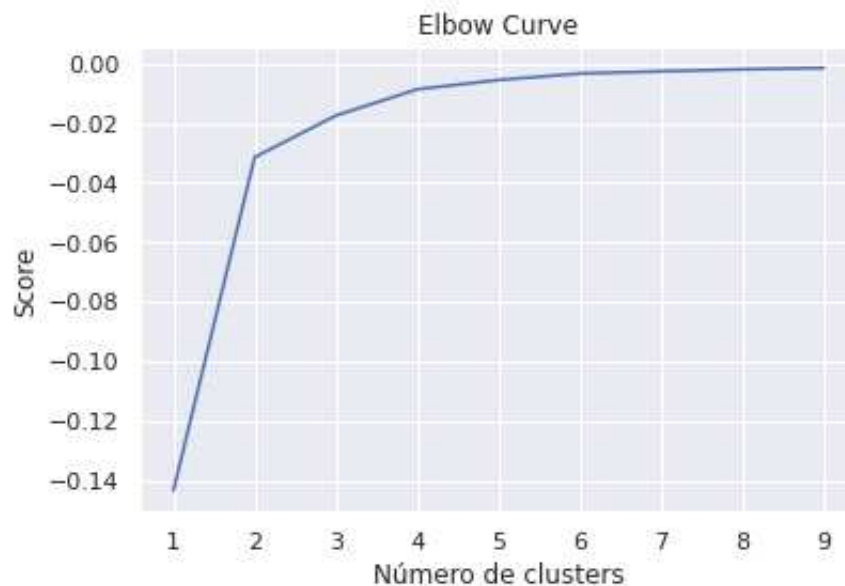


Figura 25 – Aplicação do método Elbow para definição do número de centroids a ser utilizado.

**Fonte:** Autoria Própria (2022).

Para a realização deste trabalho, foi decidido utilizar quatro clusters para os dados fictícios gerados, com o intuito de separar de maneira mais visual a disposição dos grupos de usuários dentro da região selecionada.

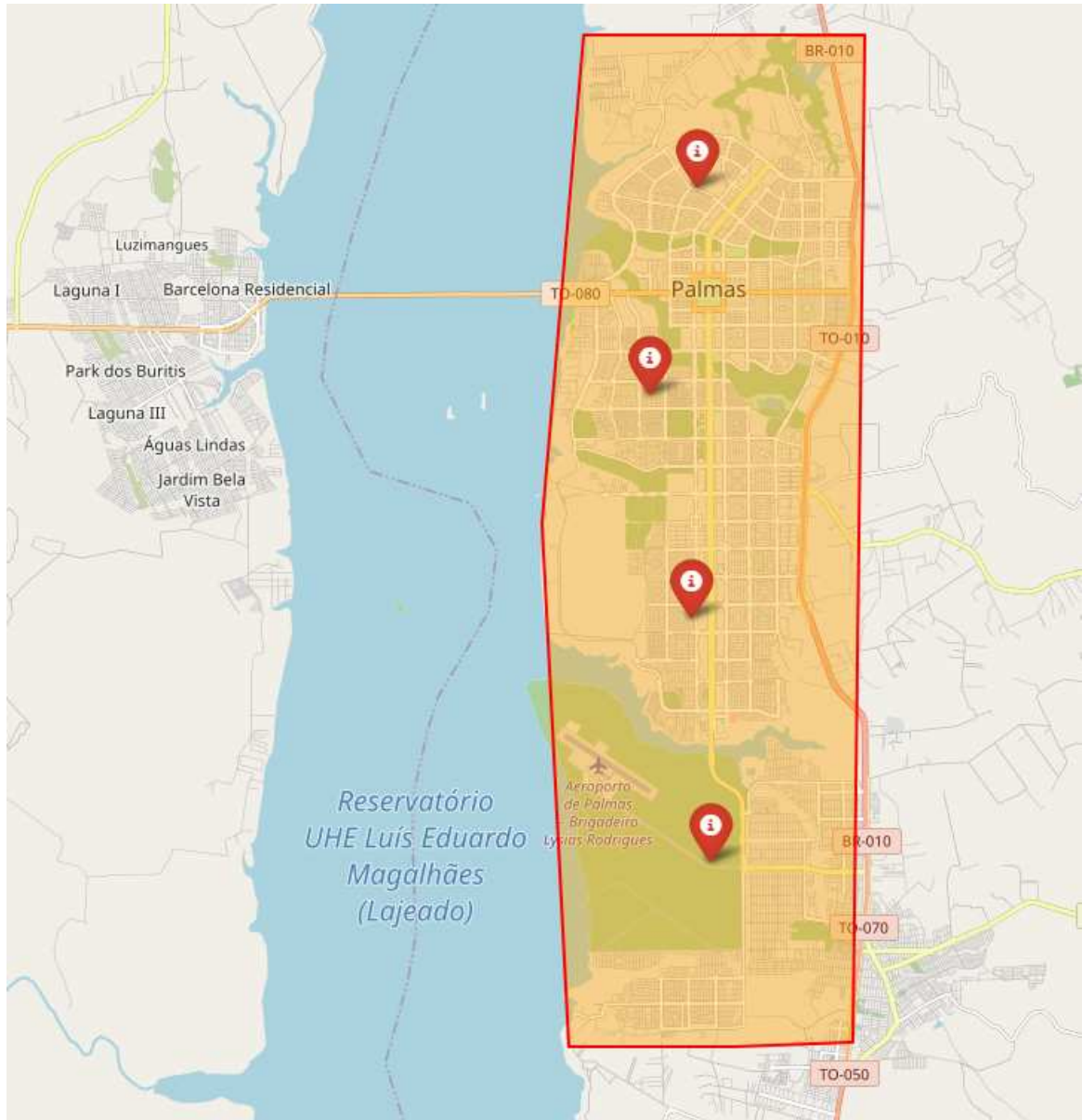


Figura 26 – Disposição dos centroides gerados no mapa com 4 clusters.

**Fonte:** Autoria Própria (2022).

Como pode ser notado na figura 26, ao utilizar 4 clusters nos dados fictícios dos usuários gerados, é possível notar uma disposição dos 4 centroides de modo que está sendo dividido a região em 4 área, facilitando posteriormente testes em cima da aplicação, onde é possível simular perda de animais tanto nas extremidades da cidade, quanto o comportamento em relação aos centroides próximos entre si.

### 3.5.3 Adição de um usuário novo do aplicativo no seu respectivo grupo

Apesar de a geração de dados fictícios para compor os grupos de usuários, foi utilizado uma cloud function do próprio *Firebase* como gatilho para cada usuário novo adicionado via aplicativo, o mesmo será adicionado em seu respectivo cluster onde estão os usuários gerados de maneira fictícia. Para realizar tal, o cálculo que foi aplicado, é a fórmula de distância entre pontos na geometria esférica. Como a geometria esférica e a geometria Euclideana, esta que é a geometria normalmente utilizada nos cálculos matemáticos corriqueiros do dia a dia, se diferem, as fórmulas para o cálculo de distâncias entre dois pontos também o vão ser. A distância entre dois pontos na geometria Euclideana é o comprimento de um segmento de recta que une os mesmos. Por outro lado, na geometria esférica a fórmula para calcular a distância entre dois pontos equação 3.5.3 é:

$$d = R * \arcsin(\sin x_1 * \sin y_1 + \cos x_1 * \cos y_1 * \cos x_2 - y_2)$$

onde R é o raio da esfera,  $x_1$  e  $y_1$  são os respectivos valores de latitude dos dois pontos, e  $x_2$  e  $y_2$  são os respectivos valores de longitude dos pontos. No caso, como estamos trabalhando com latitude e longitude, o raio da esfera que será utilizado será o valor do raio do planeta terra 6371 (seis mil trezentos e setenta e um) quilômetro. Para adicionar um usuário novo em um dos clusters criados, os valores atribuídos a  $x_1$  e  $y_1$  serão os valores do dispositivo do usuário ao realizar o cadastro, enquanto para cada centroide do respectivo cluster será executado um algoritmo para determinar qual dos clusters o usuário está mais próximo, como demonstrado no algoritmo abaixo:

---

**Algoritmo 2:** Alocação de um usuário para o cluster mais próximo.

---

**input** : Usuário novo cadastrado no sistema  
**output** : Adição de um novo usuário no cluster  
**data** : instância com todos os clusters

```
1 Function CalcularDistancia(lat1, lon1, lat2, lon2):  
2   raioDaTerra  $\leftarrow$  6371  
3   distanciaEntreLatitudes  $\leftarrow$  (lat2 - lat1) * ( $\pi/180$ )  
4   distanciaEntreLongitudes  $\leftarrow$  (lon2 - lon1) * ( $\pi/180$ )  
5   area  $\leftarrow$   $\sin(\text{distanciaEntreLatitudes}/2) * \sin(\text{distanciaEntreLatitudes}/2) + \cos(\text{lat1} * (\pi/180)) * \cos(\text{lat2} * (\text{Math.PI}/180)) * \text{Math.sin}(\text{distanciaEntreLongitudes}/2) * \text{Math.sin}(\text{distanciaEntreLongitudes}/2)$   
6   circunferencia  $\leftarrow$   $2 * \tan(\sqrt{\text{area}}, \sqrt{1 - \text{area}})$   
7   distancia  $\leftarrow$  raioDaTerra * circunferencia  
8   return distancia;  
9 End Function  
10 foreach cluster  $\in$  clusters do  
11   distanciasDosClusters.adicionar(  
    {distancia :  
      CalcularDistancia(usuario.latitude,  
        usuario.longitude,  
        cluster.latitude,  
        cluster.longitude),  
    cluster : cluster})  
12 end  
13 menorDistancia  $\leftarrow$  100  
14 clusterParaSerAdicionado  $\leftarrow$  nulo  
15 foreach distancia  $\in$  distanciasDosClusters do  
16   if distancia.distancia < menorDistancia then  
17     menorDistancia  $\leftarrow$  distancia.distancia  
18     clusterParaSerAdicionado  $\leftarrow$  distancia.cluster  
19 end  
20 clusterParaSerAdicionado.adicionar(usuario);
```

---

Como visto no algoritmo acima, a partir do momento onde é calculado a distância entre cada centroides e o usuário novo adicionado, essa informação é armazenada em um vetor, onde é posteriormente utilizado um método escrito em *Javascript* para adicionar a partir do cluster mais próximo o novo usuário criado.

## 3.6 Máquina de estados finitos para identificação do animal perdido

Um autômato finito é um tipo de sistema de estado finito constituído a partir de um conjunto de passos para realizar alguma tarefa, transições entre estados e eventos ou ações que acionam as transições para outros autômatos. Um autômato finito é usado para modelar sistemas que se comportam de acordo com um conjunto de regras. Estas regras são implementadas como transições entre estados. Quando um evento ocorre, o autômato finito altera o seu estado, levando a uma nova transição. O autômato finito, então, continua a seguir as regras até que seu estado seja alterado novamente. (BONATO, 2020)

Para o desenvolvimento do aplicativo, foi utilizada uma máquina de estados finitos para aplicar em cada um dos seus autômatos, uma ação visando identificar o dono de um animal perdido, e realizar as ações de notificação e início de conversa entre o possível dono do animal identificado e o usuário que achou, e em caso de anonimato, início de conversa com o próprio sistema.

### 3.6.1 Ações ao identificar de um animal perdido

Como descrito acima, a máquina de estado finito atuará no contexto de notificação dos donos dos animais, seguindo uma sequência de regras montadas para avisar os donos que perderam seus animais, sem influenciar, caso não seja necessário, os demais donos de animais cadastrados no sistema. Na figura abaixo, é possível verificar a entrada e as sequências que deverão ser tomadas pela máquina de estados finita antes de notificar os donos dos animais. A máquina de estados será acionada a partir da identificação de um animal em uma coordenada, sendo esta identificação realizada por uma API alocada em um contêiner *Docker* (DOCKER, 2022), hospedado no ambiente *Google Cloud* (GOOGLE, 2022e), onde por meio de uma função HTTP hospedada no *Firebase*, percorrerá os clusters cadastrados e verificará em cada um dos clusters a presença de donos de animais que perderam seus pets.

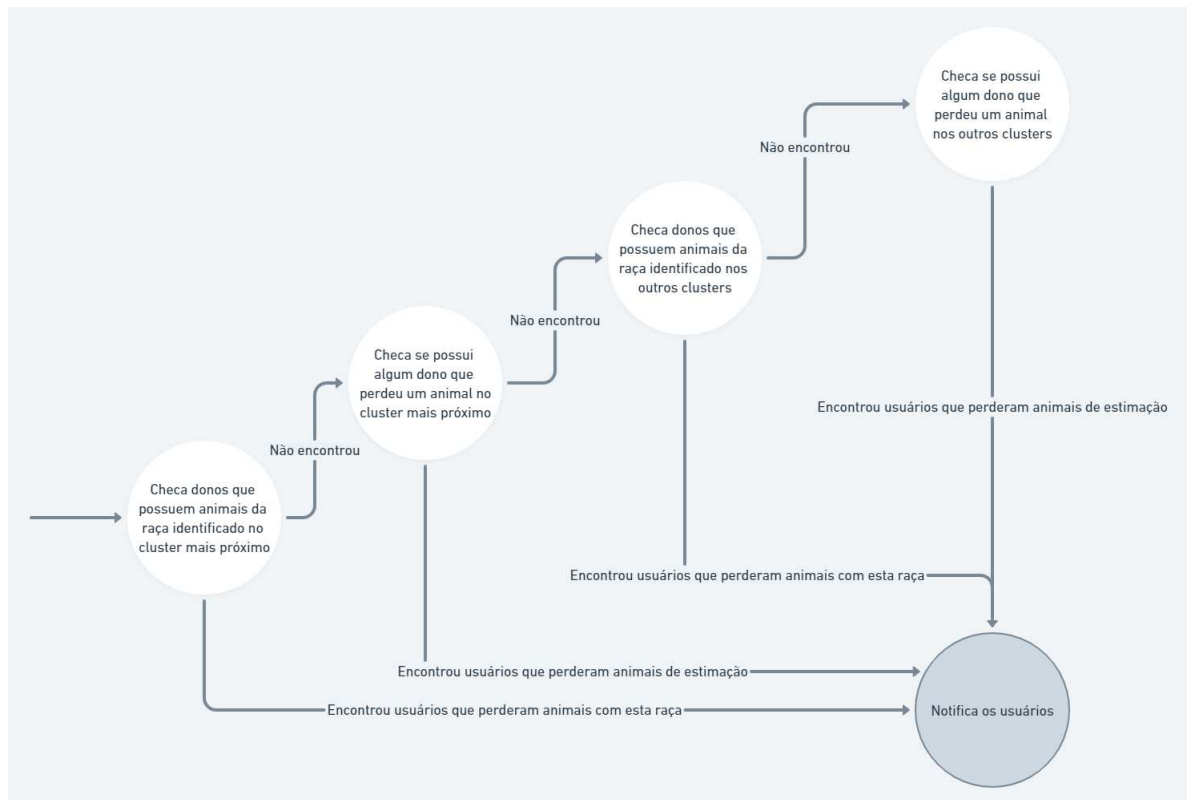


Figura 27 – Máquina de estado finito para notificação dos donos de animais.

**Fonte:** Autoria Própria (2022).

A implementação da máquina exibida na figura acima, foi feita pela biblioteca *FSM* (FSM, 2022) da linguagem *Javascript*, devido à facilidade decorrente da implementação de tal algoritmo em uma *Cloud Function* do *Firebase*, por sua tecnologia nativa também ser *Javascript*. Além da implementação da máquina de estados finita, devido à possibilidade de um animal ser encontrado e o dono ainda não estar cadastrado no sistema, foi utilizado uma estratégia de execução da máquina d estados a cada 24 horas, por uma *Cron jobs* através da hospedagem ofertada pelo *Firebase*, com o tempo definido através da seguinte sequência de caracteres: *0 23 \* \* \**, onde os "\*" representam qualquer nenhuma alteração na casa dias e meses, e o número 0 e 23 representam respectivamente 0 minutos e 23 horas, neste caso gerando uma função que sempre executará às 23 horas e 0 minutos.

O uso de cron jobs é amplamente utilizado para tarefas automatizadas, como backups, envio de e-mails, processamento de trabalhos, etc. Isso permite que a execução de tarefas seja automatizada e economiza tempo e recursos. Além disso, as tarefas podem ser agendadas para executar durante períodos de baixa utilização, o que pode ser útil para evitar o congestionamento de servidores ou dentro das aplicações (UBUNTU, 2016).

Para a realização de tal tarefa agendada, é salvo dentro do *Cloud Firestore* uma coleção de *logs* aonde temos as informações de quando a foto de um determinado animal é enviada, assim como os clusters, usuários e se aquele animal a qual pertence à foto já retornou ao seu dono, ação esta realizada pelo aplicativo, caso ainda esteja com a identificação de animal perdido, a função executará a máquina de estado novamente a partir do log salvo.

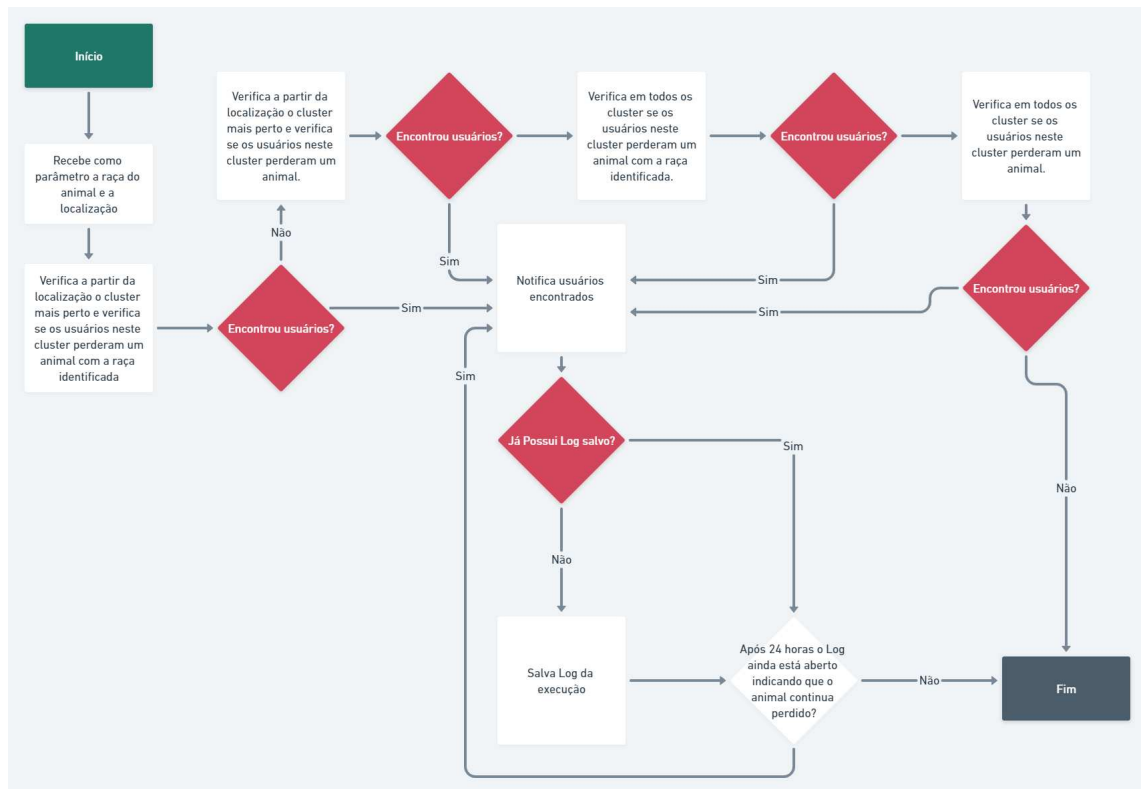


Figura 28 – Representação do algoritmo da Máquina de estado finito.

**Fonte:** Autoria Própria (2022).

Na figura 28 é possível ver a representação do funcionamento do algoritmo, passando por cada estado da máquina de estados finitos, até a notificação e a abertura de conversa entre usuários e sistema.

### 3.7 UML (*Unified Model Language - Linguagem de Modelagem Unificada*)

A linguagem UML é uma linguagem de modelagem de projetos utilizada na engenharia de software que visa descrever características e funcionamentos do sistema através de modelos visuais e documentos descritivos, de modo a auxiliar os desenvolvedores no desenvolvimento do projeto, evitando redundâncias e retrabalhos no ciclo de vida de desenvolvimento de um software. Sendo ressaltado por Guedes, no livro "UML 2 - Uma abordagem prática", o UML é:

"[...] A UML (Unified Modeling Language – Linguagem de Modelagem Unificada) tornou-se, nos últimos anos, a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software.[...]"([GUEDES, 2018](#))

Sendo assim essencial para o desenvolvimento e análise da regra de negócio do sistema.



### 3.7.1 Diagrama de Contêineres

A figura abaixo representa o funcionamento do sistema separando suas funcionalidades por contêineres, onde é descrito quais serão os serviços e as tecnologias utilizadas para o funcionamento e execução de como será desenvolvido o aplicativo no futuro.

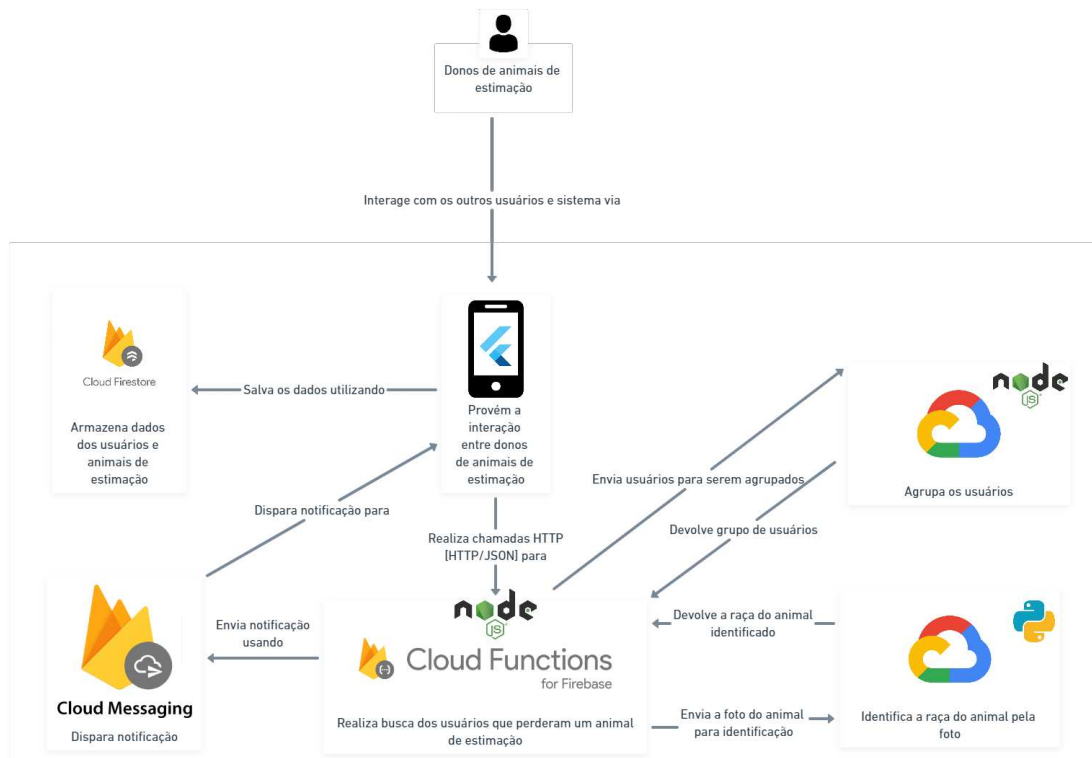


Figura 29 – Diagrama de container.

**Fonte:** Autoria Própria (2022).

Como pode ser percebido na Figura 29, o sistema está atrelado principalmente aos serviços ofertados pelo *Firebase*, com exceção da execução dos algoritmos de aprendizado de máquina, as quais foram separados em contêineres no Google Cloud a fim de obter mais processamento.

### 3.7.2 UML de Caso de Uso

O UML de caso de uso visa descrever de maneira gráfica funcionalidade de um sistema em fase de desenvolvimento. Assim, facilitando ao desenvolvedor, a obtenção de uma visão clara sobre o sistema em que o mesmo está trabalhando, além de ajudar o usuário final na percepção de funcionalidades do sistema. (TRABALHO, 2022)

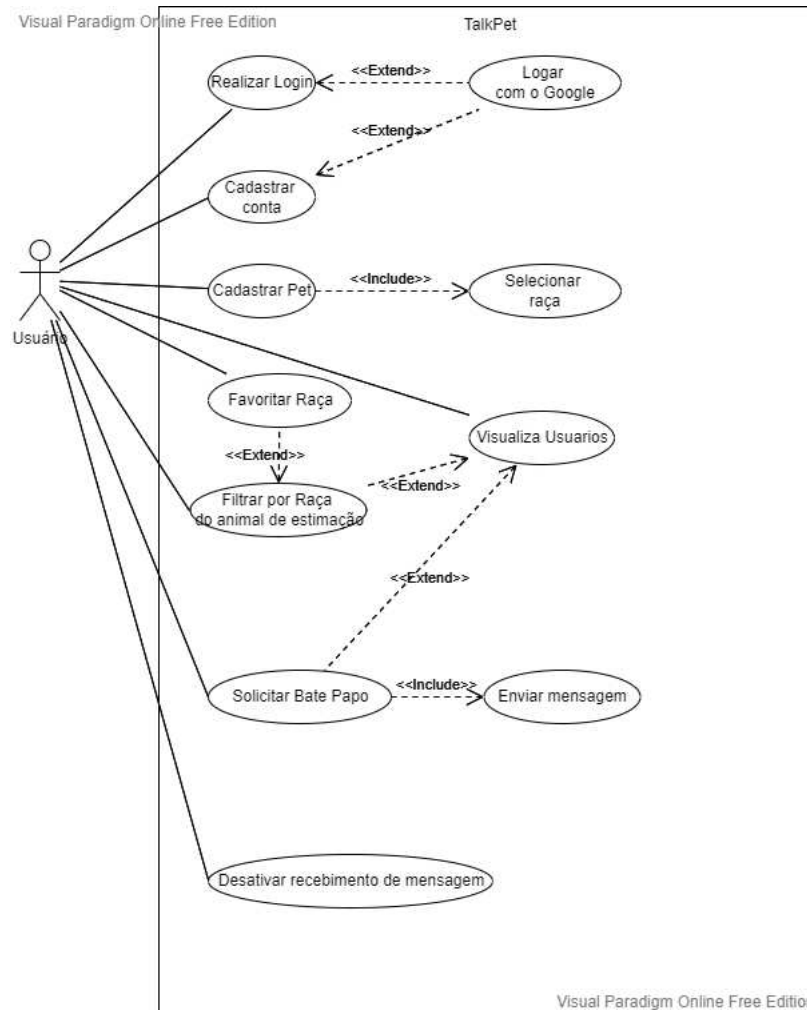


Figura 30 – UML de Caso de Uso do Sistema.

O modelo UML de caso de uso do sistema apresentado na Figura 30, retratando todos os comportamentos do mesmo e quais são as exigências necessárias para executar tais procedimentos, funções essas sendo complementadas e descritas no documento de requisitos funcionais.

### 3.7.3 UML de Pacotes

Para o desenvolvimento do sistema, decidiu-se utilizar uma arquitetura limpa, como mencionado por MARTIN em seu livro "Clean Architecture: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series)":

"[...] A arquitetura limpa é se concentra na separação de conceitos. Dividindo o software em camadas onde dada um tem pelo menos uma camada para regras de negócios e outra camada para interfaces do usuário e do sistema.[...]"(MARTIN, 2017)

Proporcionar uma separação de camadas na estrutura do aplicativo, permitindo testes unitário e independência das implementações dos recursos do sistema, vantajosa para o desenvolvimento desta aplicação.

Abaixo podemos observar a separação de camadas desta arquitetura.

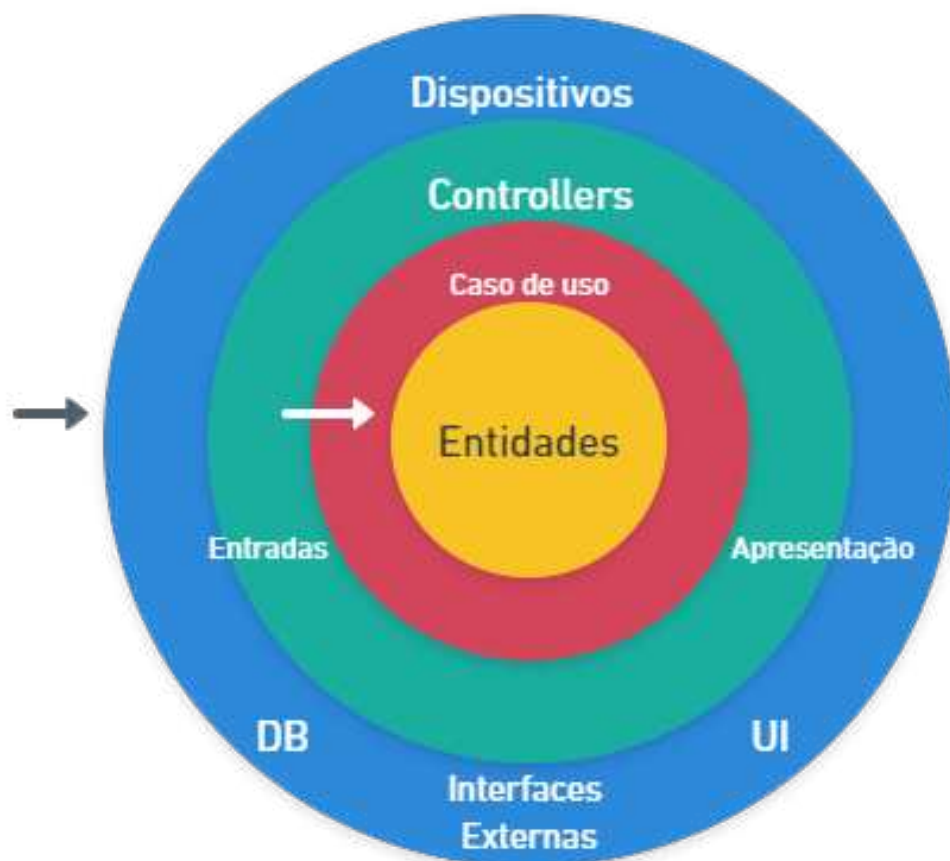


Figura 31 – Arquitetura do Aplicativo (Clean Architecture).

**Fonte:** Autoria Própria (2022).

Baseado na arquitetura, para o desenvolvimento da aplicação foi optado por utilizar uma variação da mesma focada na arquitetura de providers, devido à utilização da biblioteca

de gerência de estado riverpod.

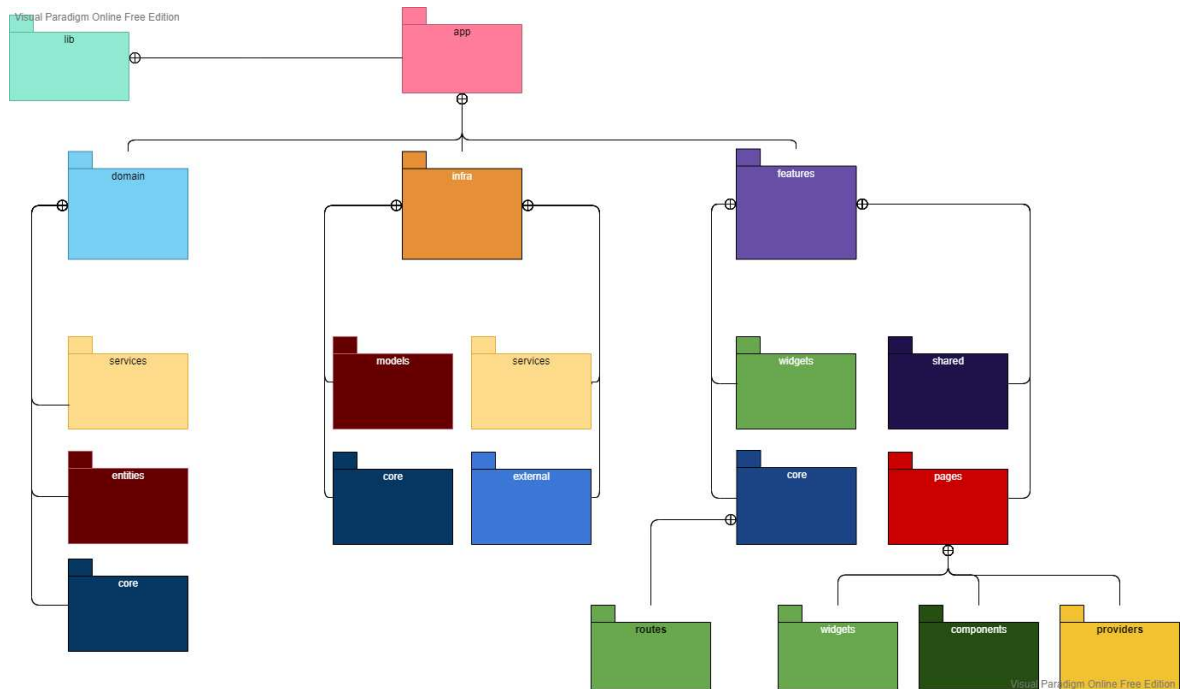


Figura 32 – UML de Pacotes do Sistema.

**Fonte:** Autoria Própria (2022).

A arquitetura de software desenvolvida para o aplicativo foi desenvolvida no intuito de maximizar a independência das camadas da aplicação, onde cada pasta tem sua responsabilidade dentro do contexto geral, onde para cada mudança é notificado em tempo de compilação no projeto inteiro de maneira a evitar o máximo possível a geração de erros do projeto.

## 3.8 Diagrama de Atividades

O diagrama de atividades tem como finalidade fornecer uma visualização de um comportamento de um sistema descrevendo a sequência de ações a serem realizadas para alcançar o objetivo da mesma. (CORPORATION, 2021)

Sua utilização é necessária para demonstrar atividades nas quais não foram detalhadas com clareza no diagrama de caso de uso. As atividades que serão detalhadas serão a atividade de cadastrar, realizar 'login', seleção de raça do animal e desativar recebimento de mensagens.

### 3.8.1 Diagrama de Atividades do Login de um usuário

O diagrama de atividades a seguir descreve e detalha como deve ser feito o cadastro de novos usuários com provedor do Google. Para a utilização do sistema é necessário que o usuário possua uma conta Google, para poder entrar no aplicativo como um usuário novo já possuindo informações como telefone, foto de perfil, nome e e-mail sem passar por um processo manual de cadastro.

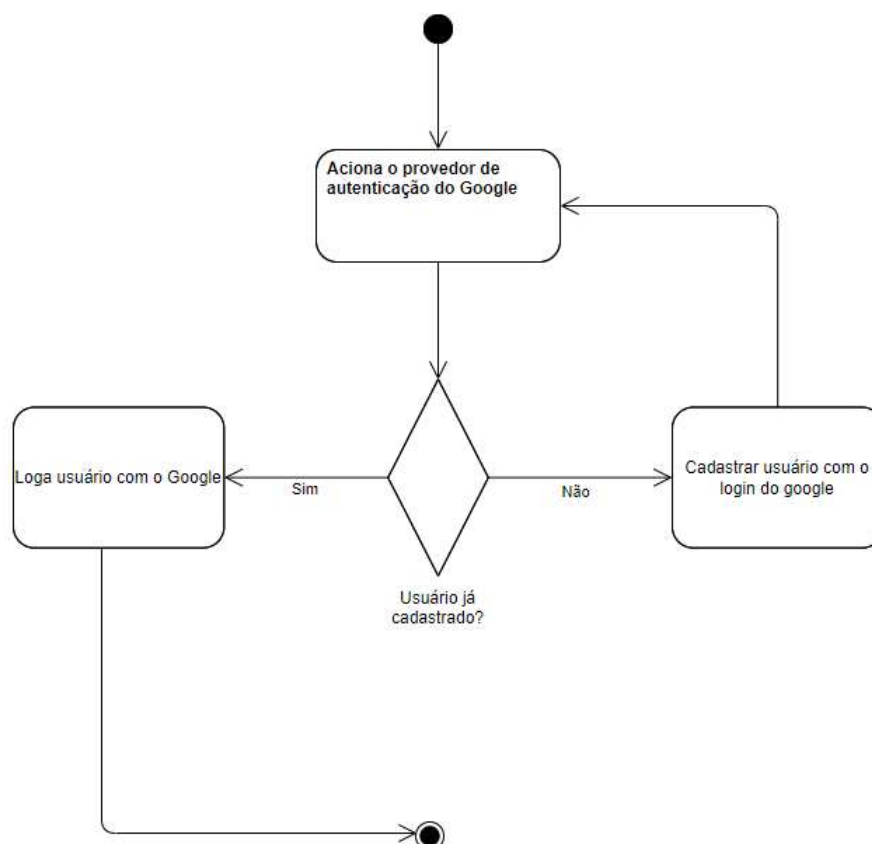


Figura 33 – Diagrama de Atividades do Cadastro no Sistema.

**Fonte:** Autoria Própria (2022).

Os usuários serão cadastrados diretamente na base de dados do Firebase, ainda sem estar vinculado a nenhum animal de estimação. Caso a senha não seja lembrada por qualquer um dos usuários, o mesmo poderá solicitar um e-mail de recuperação de senha, redefinindo a mesma pelo serviço ofertado via Firebase.

### 3.8.2 Diagrama de seleção da raça do animal

O diagrama de atividades a seguir descreve e detalha como deve ser feita a seleção da raça do animal de estimação baseada na rede neural desenvolvida.

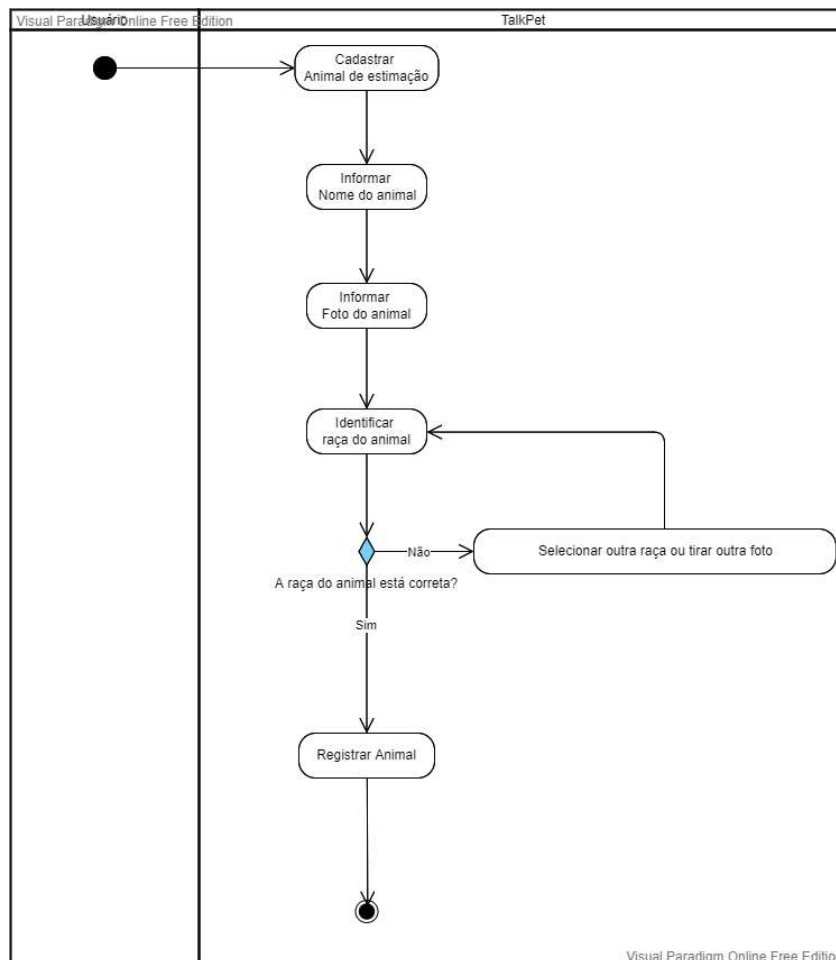


Figura 34 – Diagrama de Atividades de Seleção de raça do animal.

**Fonte:** Autoria Própria (2022).

Quando o usuário adicionar uma foto do animal, a rede neural deve identificar a raça do mesmo e redirecionar o usuário para uma tela de confirmação, caso a raça selecionada não seja correta segundo o próprio dono, o mesmo tem a opção de selecionar manualmente ou tirar outra foto do animal.



## 4 Resultados

Os resultados obtidos por meio deste trabalho, contemplam soluções voltadas para a área de engenharia de software, assim como resultados relacionados ao desempenho do treinamento das redes neurais e suas implicações voltadas a qualidade do software desenvolvido e associação com a legislação brasileira do ano de 2022.

### 4.1 Desempenho redes neurais

Como descrito anteriormente, para o desenvolvimento deste trabalho foram treinadas 4 redes neurais utilizando diferentes técnicas, para cada uma delas foi definida a utilização de 10 *épocas*<sup>1</sup> para todas, pelo motivo do longo tempo de treinamento devido ao número alto de raças e imagens que necessitavam ser treinadas, segundo os testes realizados com diferentes números de épocas, foi constatado um resultado satisfatório com 10 épocas para cada uma das redes treinadas, diminuindo a perda e evitando *overfitting*<sup>2</sup>, sendo tal resultado demonstrado para cada rede abaixo.

#### 4.1.1 Primeira Rede neural

A primeira rede neural treinada levou 3.207 segundos para realizar o treinamento, aproximadamente 53 minutos, tendo uma acurácia de 5% no dataset de treinamento, com uma perda em torno de, 4.33 valor este que representa a diferença entre a saída da rede em relação ao resultado esperado, e no dataset de teste e validação essa mesma rede neural apresentou um desempenho igual a 4%, com 4.53 de perda. Abaixo será exibido os gráficos pertinentes a acurácia desta rede em relação ao dataset de treino e validação.

---

<sup>1</sup> épocas: número de vezes que vamos exibir os dados para rede neural.

<sup>2</sup> overfitting: ajuste excessivo é um comportamento indesejável de aprendizado de máquina que ocorre quando o modelo de aprendizado de máquina fornece previsões precisas para dados de treinamento, mas não para novos dados

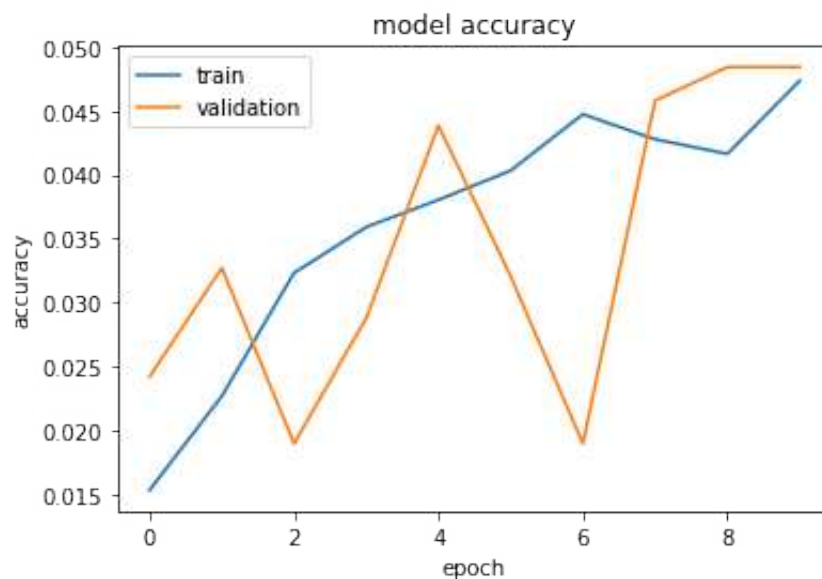


Figura 35 – Acurácia da primeira rede neural.

**Fonte:** Autoria Própria (2022).

Já com relação à perda desta rede, podemos ver a baixo saltos exponenciais da perda de informação, invalidando a utilização desta como uma rede viável.

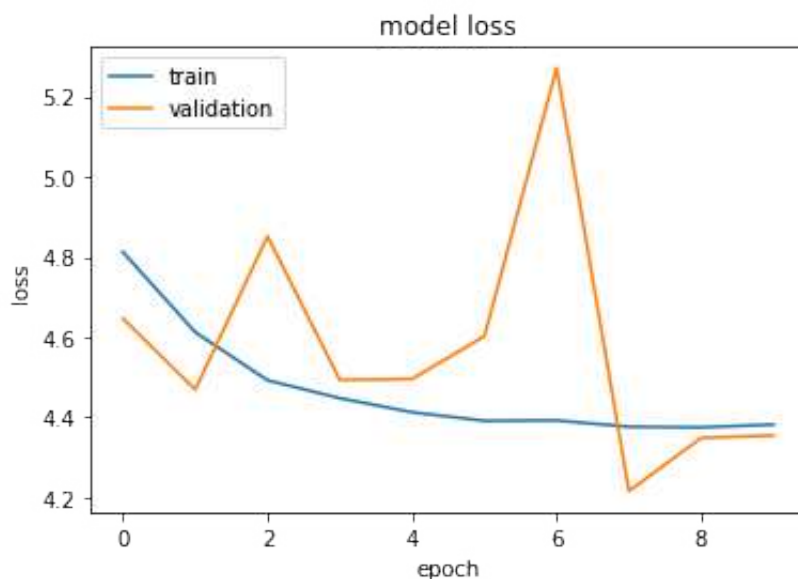


Figura 36 – Perda da primeira rede neural.

**Fonte:** Autoria Própria (2022).

Os resultados da primeira rede neural se mostraram insatisfatório para ser aplicadas como validação de raça no contexto da perda de animais perdidos, apresentando uma perda muito alta e baixa assertividade em testes realizados.

### 4.1.2 Segunda Rede neural

A segunda rede neural treinada, utilizou uma abordagem diferente da anterior, permitindo o reajuste de pesos pela própria rede, além de manter as camadas mais densas, aplicando, diferentes funções de ativação para maximizar a eficiência. Esta rede gerou uma acurácia parecida com a anterior, 5%, além de minimizar a inconstância da perda, se tornando mais gradual, onde provavelmente ao aumentar o número de épocas para esta rede, seria possível melhorar o desempenho. Abaixo será exibido os gráficos pertinentes a acurácia desta rede em relação ao dataset de treino e validação.

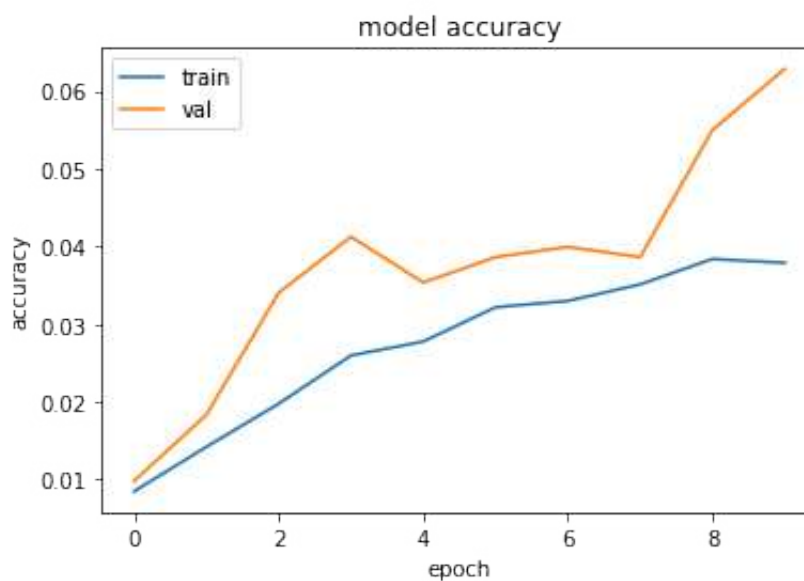


Figura 37 – Acurácia da segunda rede neural.

**Fonte:** Autoria Própria (2022).

Já com relação à perda desta rede, como mencionado antes, se comporta de uma maneira mais homogênea, não apresentando saltos como na primeira rede testada.

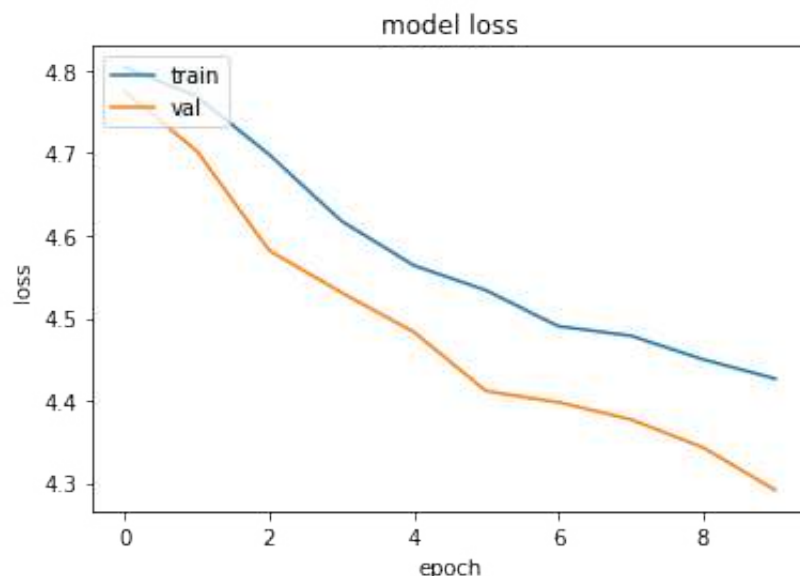


Figura 38 – Perda da segunda rede neural.

**Fonte:** Autoria Própria (2022).

Apesar de uma perda decrescente gradual, esta rede neural teve um desempenho inferior à rede anterior, com uma acurácia de apenas 3%, além disto o tempo de treinamento da mesma também teve um aumento considerável, em torno 7.287 segundos, o que em minutos é equivalente a 171 minutos e em horas 2 horas e 1 minuto.

#### 4.1.3 Terceira Rede neural (MobileNetV2)

Devido à baixa acurácia apresentada pelas redes anteriores, a terceira rede neural foi treinada com o auxílio dos pesos e resultados do modelo *MobileNetV2*. Ao utilizar este modelo no treinamento da rede neural, foi notável o aumento considerável na acurácia da rede, atingindo a acurácia de 81% para as imagens de treino, com uma perda em torno de 73%, muito menor que seus antecessores. Ao fazer a validação utilizando o dataset de validação, obteve-se uma acurácia de 70% e uma perda de 1.5, sendo muito menor que os resultados anteriores das outras redes, se mostrando mais assertiva, apesar do aumento da perda a utilização do modelo *MobileNetV2* como auxílio no treinamento se tornou uma opção viável para a identificação dos animais nesta problemática. Além disto, semelhante à primeira rede, esta rede neural teve um tempo de treinamento estimado em 3.042 segundos, aproximadamente 50 minutos. Abaixo será exibido os gráficos pertinentes a acurácia desta rede em relação ao dataset de treino e validação.

Esta rede neural apresentou uma perda acentuada na parte de validação, porém apesar desta perda acentuada, ainda permaneceu com um bom resultado.

Contudo, é válido ressaltar, que a estratégia utilizada para comunicação da rede neural com o aplicativo, impossibilitou a utilização do modelo *MobileNetV2* para esta

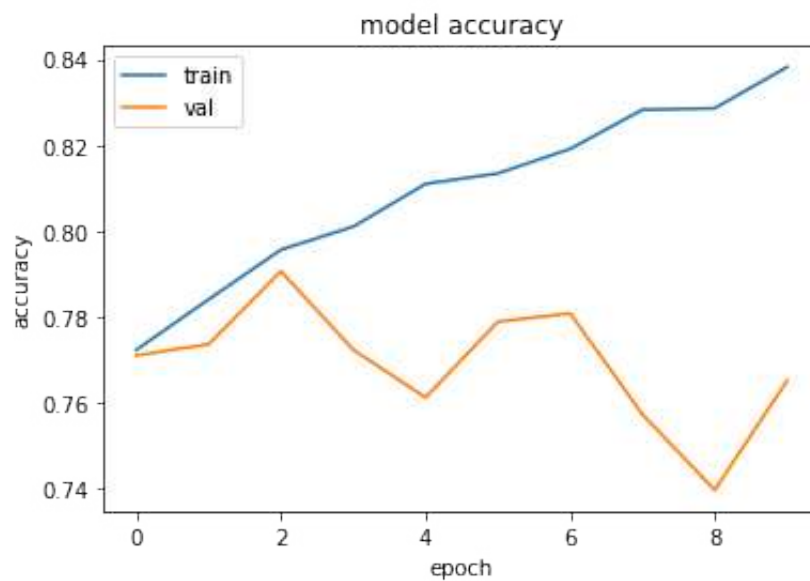


Figura 39 – Acurácia da terceira rede neural.

**Fonte:** Autoria Própria (2022).

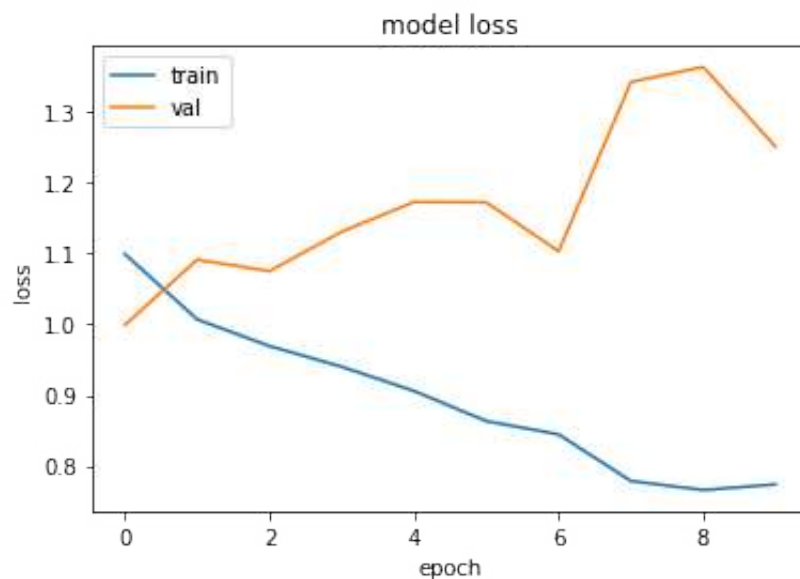


Figura 40 – Perda da terceira rede neural.

**Fonte:** Autoria Própria (2022).

problemática, devido ao fato da incompatibilidade da leitura do modelo enquanto é realizada a requisição via API utilizando a biblioteca *FastAPI* em sua versão 0.88.0 da linguagem Python. Em futuras versões espera-se que este problema de leitura com este modelo de auxílio se resolva, porém, devido ao problema relatado foi necessário o treinamento utilizando outro modelo de pesos auxiliar.

#### 4.1.4 Quarta Rede neural (InceptionV3)

Devido às dificuldades na implementação de uma API utilizando o modelo *MobileNetV2*, foi utilizado outro modelo para o treinamento, visando manter uma acurácia alta e permitindo a comunicação entre aplicativo e API sem maiores dificuldades. O modelo escolhido para substituir o MobileNet foi o modelo *InceptionV3*, onde o funcionamento similar ao MobileNet e consumo e reajuste de pesos a partir da mesma biblioteca de imagem do Google, este modelo ofertou uma acurácia de 81% no dataset de treino e uma acurácia de 73% no dataset de validação e testes. Abaixo será exibido os gráficos pertinentes a acurácia desta rede em relação ao dataset de treino e validação.

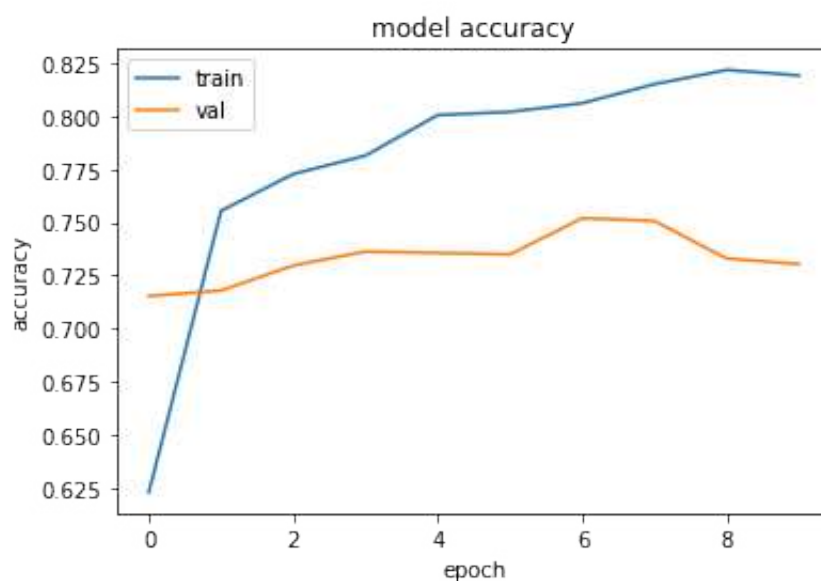


Figura 41 – Acurácia da quarta rede neural.

**Fonte:** Autoria Própria (2022).

A perda desta rede neural, se mostrou menor em torno de 1 a 1.8, o que é um valor baixo, e satisfatório na utilização para solucionar a problemática.

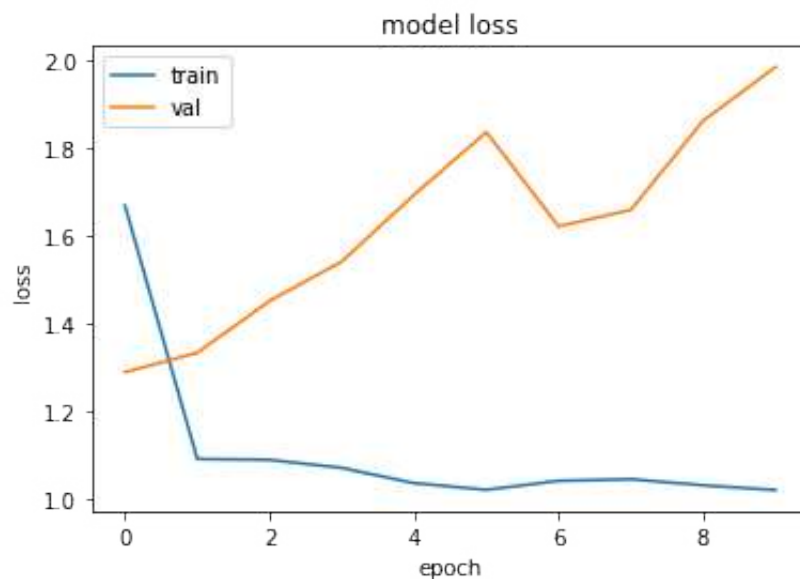


Figura 42 – Perda da quarta rede neural.

**Fonte:** Autoria Própria (2022).

A fim de confirmar a validação do modelo, foi feito um teste em massa com 2.556 imagens de cachorros disponíveis no dataset de teste, disponíveis para validação, os resultados confirmaram a acurácia mostrada, tendo 2.017 imagens com a predição correta, onde a coluna *breed* representa a raça correta do animal e a coluna *prediction* representa qual raça a rede neural identificou, como demonstrado na figura abaixo.

	id	breed	group	prediction	Compare
0	000bec180eb18c7604dcecc8fe0dba07.jpg	boston_bull	Terrier	boston_bull	Correct
2	003df8b8a8b05244b1d920bb6cf451f9.jpg	basenji	Hound	basenji	Correct
3	004396df1acd0f1247b740ca2b14616e.jpg	shetland_sheepdog	Herding	shetland_sheepdog	Correct
5	006cc3ddb9dc1bd827479569fcdc52dc.jpg	bluetick	Hound	bluetick	Correct
7	0097c6242c6f3071762d9f85c3ef1b2f.jpg	bedlington_terrier	Terrier	bedlington_terrier	Correct
...	...	...	...	...	...
2551	ffa0055ec324829882186bae29491645.jpg	maltese_dog	Toy	maltese_dog	Correct
2552	ffa6a8d29ce57eb760d0f182abada4bf.jpg	english_foxhound	Hound	english_foxhound	Correct
2553	ffbfb7536ba86dcef3f360bda41181b4.jpg	weimaraner	Sporting	weimaraner	Correct
2554	ffcde16e7da0872c357fbc7e2168c05f.jpg	airedale	Terrier	airedale	Correct
2555	fff43b07992508bc822f33d8ffd902ae.jpg	chesapeake_bay_retriever	Sporting	chesapeake_bay_retriever	Correct

2017 rows x 5 columns

Figura 43 – Imagens com predição correta da quarta rede neural.

**Fonte:** Autoria Própria (2022).

Enquanto as incorretas estão em torno de 539 imagens, representando 21% do total de 2.556 imagens categorizadas incorretamente, como pode ser visto na figura abaixo.

	id	breed	group	prediction	Compare
1	00290d3e1fdd27226ba27a8ce248ce85.jpg	bedlington_terrier	Terrier	lakeland_terrier	Not Correct
4	0067dc3eab0b3c3ef0439477624d85d6.jpg	walker_hound	Hound	english_foxhound	Not Correct
6	007b8a07882822475a4ce6581e70b1f8.jpg	redbone	Hound	rhodesian_ridgeback	Not Correct
11	00f34ac0a16ef43e6fd1de49a26081ce.jpg	walker_hound	Hound	basset	Not Correct
17	0162107acd8f2588c0944b791d61bb0c.jpg	affenpinscher	Toy	giant_schnauzer	Not Correct
...	...	...	...	...	...
2534	fdccec2dc716306a12b773e7689887c0.jpg	staffordshire_bullterrier	Terrier	bloodhound	Not Correct
2535	fdce4d488a629164ee8a9f2a0b81905f.jpg	norwegian_elkhound	Hound	german_shepherd	Not Correct
2537	fdcb09d408084b1289bf572bd4071c9c.jpg	airedale	Terrier	otterhound	Not Correct
2540	fe54e87e65fe0c68670c0dd1a923f1f0.jpg	bouvier_des_flandres	Herding	scotch_terrier	Not Correct
2550	ff63fa05a58473138848f80840064d23.jpg	affenpinscher	Toy	bouvier_des_flandres	Not Correct
539 rows x 5 columns					

Figura 44 – Imagens com predição incorreta da quarta rede neural.

**Fonte:** Autoria Própria (2022).

O tempo de treinamento do modelo *InceptionV3* se mostrou similar a rede anterior, tendo um tempo estimado de 3.197 segundos, valor este convertido em aproximadamente 53 minutos de treino, a utilização desta rede na aplicação foi realizado via um contêiner no Google cloud, onde é enviado a imagem obtida no aplicativo e retornado em formato *JSON*<sup>3</sup>.

#### 4.1.4.1 Matriz Confusão

A matriz de confusão mostra a quantidade de vezes que um algoritmo classificou correta e incorretamente cada classe. O principal uso da matriz de confusão é para avaliar a precisão de um algoritmo de classificação. Ela fornece uma visão geral do desempenho do algoritmo ao avaliar cada classe individualmente. Por exemplo, se uma matriz de confusão tem quatro linhas, o algoritmo pode ter classificado corretamente três das classes e incorretamente classificado a quarta. A matriz de confusão também permite que você compare a precisão entre classes, pois mostra a comparação entre as instâncias classificadas correta e incorretamente de cada classe. Isso pode ajudar a identificar possíveis problemas ou tendências em um algoritmo de classificação. Esse tipo de matriz pode ser utilizado em diversas áreas de pesquisa com o intuito de realizar a validação de algoritmos, um exemplo é o trabalho postado no *Centro de Pesquisas Avançadas em Qualidade de Vida* (JUNIOR, 2022), onde esta matriz é utilizada para validação de um algoritmo no auxílio da locomoção humana.

Para a validação da rede neural utilizada, será utilizado uma matriz de confusão

<sup>3</sup> JSON: JavaScript Object Notation, formato de representação de dados para troca dos mesmos entre sistemas



para avaliação das predições realizadas pela rede neural junto ao modelo *InceptionV3*. Contudo, por se tratar de uma rede neural que analisa um número muito alto de raças, para melhor visualização foi decidido agrupar estas raças conforme a sua categoria, para a base deste agrupamento, foi utilizado uma postagem realizada pela empresa Petz ([PETZ, 2022a](#)), onde os mesmos descrevem as principais categorias de raças de animais existentes principalmente no território brasileiro. ([PETZ, 2022b](#))

Como o dataset utilizado não possui a informação sobre a categoria do animal, foi criado um algoritmo, descrito na figura abaixo, da separação de cada raça para seu respectivo grupo, vale ressaltar que para manter o padrão da aplicação as raças também foram traduzidas para o inglês.

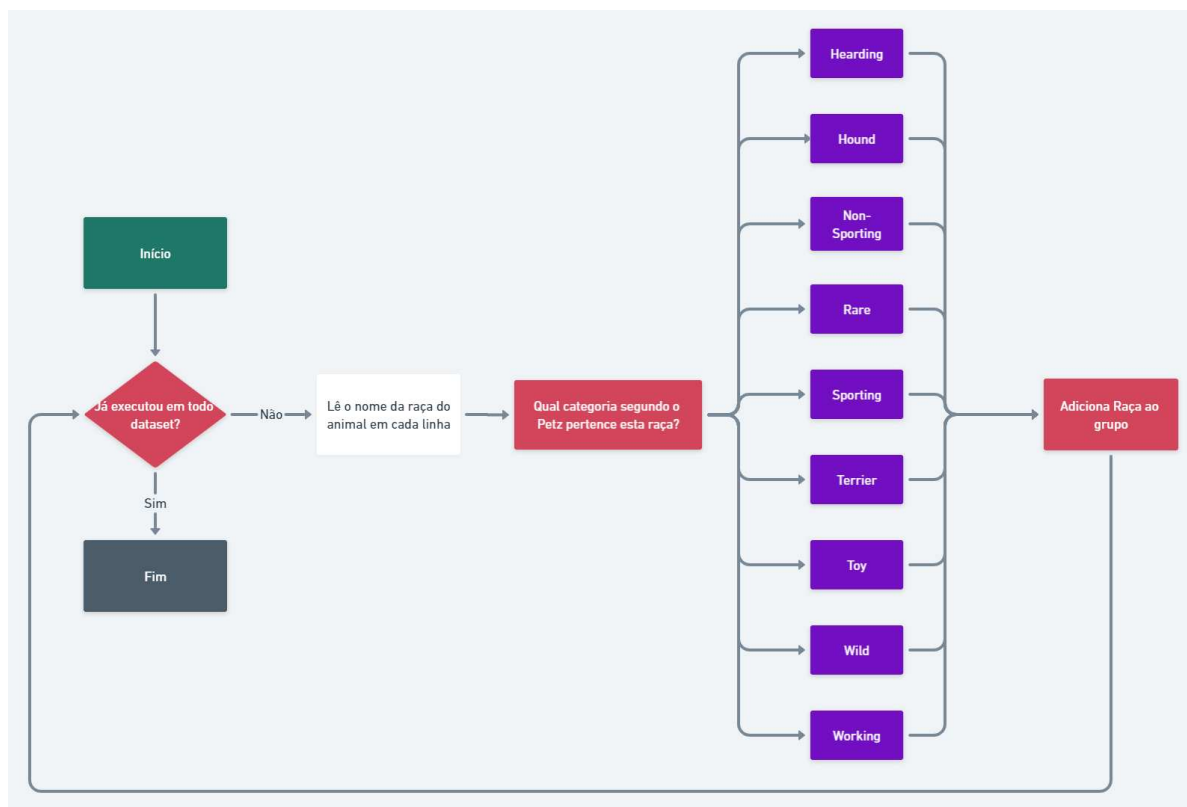


Figura 45 – Fluxograma de separação das imagens em grupos.

**Fonte:** Autoria Própria (2022).

As categorias de raças escolhidas para agrupamento das raças e suas respectivas quantidades após agrupamento são:

- **Terrier:** 3370
- **Hound:** 1818
- **Working:** 3295
- **Sporting:** 2797
- **Toy:** 3068
- **Herding:** 1921
- **Non-Sporting:** 1523
- **Rare:** 459

- Wild: 475

Abaixo é exibido a separação das raças do dataset em categorias e sua distribuição:

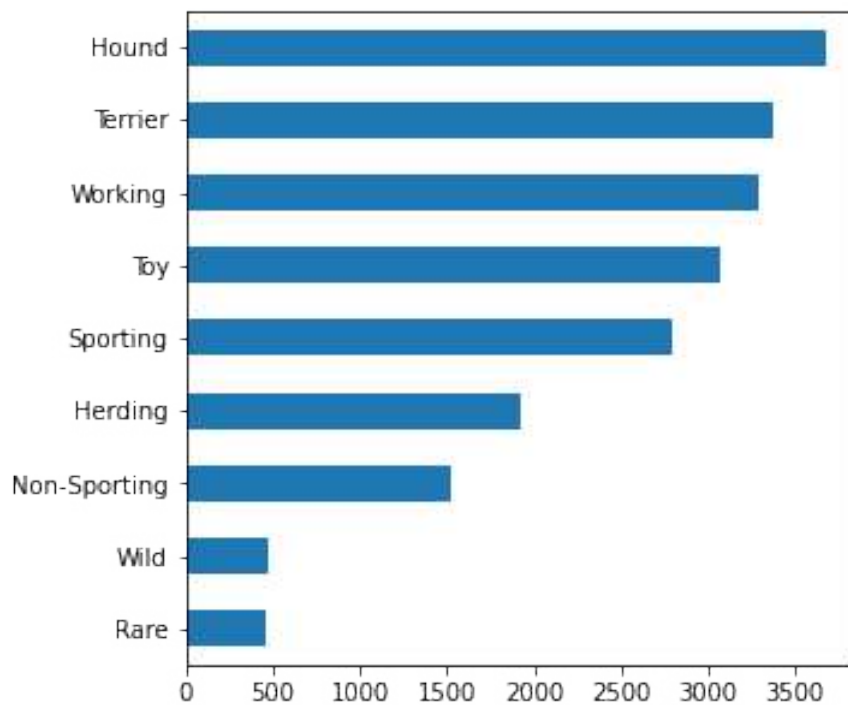


Figura 46 – Categoria das raças após agrupamento.

**Fonte:** Autoria Própria (2022).

A partir do agrupamento das raças, foi montada a visualização da matriz de confusão, obtendo os seguintes resultados:

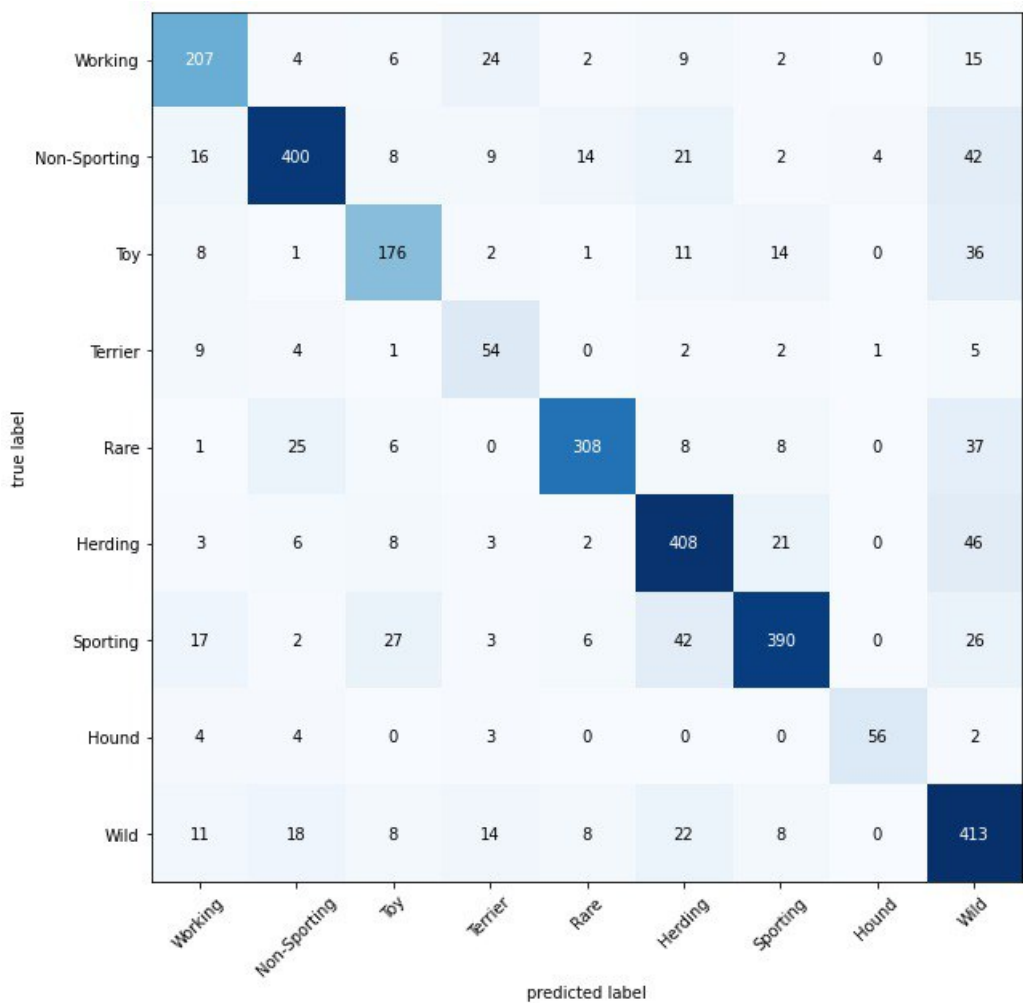


Figura 47 – Matriz de Confusão.

**Fonte:** Autoria Própria (2022).

A partir da visualização da matriz acima, é possível inferir que a probabilidade de acerto para cada categoria de raça é:

- **Terrier:** 54/3370 aproximadamente 1%
- **Hound:** 56/1818 aproximadamente 3%
- **Working:** 207/3295 aproximadamente 6%
- **Sporting:** 390/2797 aproximadamente 13%
- **Toy:** 176/3068 aproximadamente 5%
- **Herding:** 408/1921 aproximadamente 21%

- **Non-Sporting:** 400/1523 aproximadamente 26%
- **Rare:** 408/459 aproximadamente 88%
- **Wild:** 413/475 aproximadamente 86%

## 4.2 Protótipos de Telas Desenvolvidas

A ferramenta escolhida para prototipação da ‘interface’ do sistema é o Figma, o Figma é uma ferramenta gratuita desenvolvida para ajudar profissionais de *UI*<sup>4</sup>, *UX*<sup>5</sup> e desenvolvedores a interagirem entre si e melhorar a experiência quando o assunto é desenvolvimento e criação de ‘interfaces’.([WALLACE, 2022](#))

Entre toda a gama de ferramentas disponível pela ferramenta, podemos destacar uma vasta biblioteca de componentes e Design System para utilização nos protótipos, o que acelera bastante a prototipação e permite a criação de ‘design’ minimamente agradáveis mesmo para pessoas que não possuem tanta experiência quando se trata de ‘interface’ do usuário.

### 4.2.1 Telas Iniciais do Sistema

Abaixo temos as telas para cadastro no sistema, onde os usuários poderão acessar a aplicação através do seu e-mail do Google, ou, sem estar precisar fazer o processo de login, enviar a foto de um animal de achado para os donos de animais dentro do sistema serem notificados.

---

<sup>4</sup> UI: ‘interface’ do Usuário

<sup>5</sup> UX: experiência do usuário.

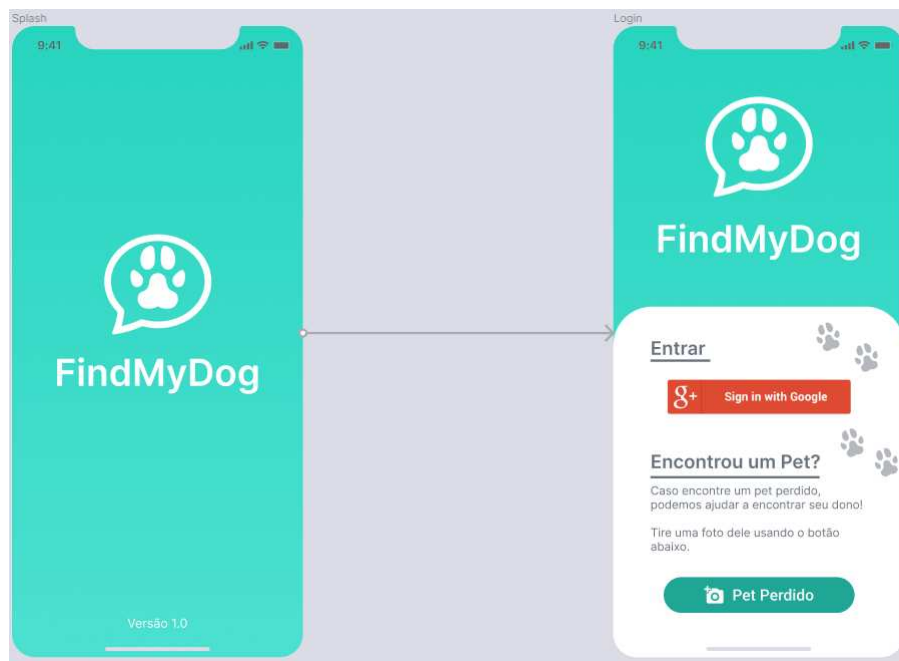


Figura 48 – Telas Iniciais do Sistema.

**Fonte:** Autoria Própria (2022).

Foi decidido a utilização do provedor do Google para utilização da aplicação devido a praticidade do retorno de informações dos mesmos, evitando fluxo externos para cadastro de imagem e informação dos usuários para troca de mensagem.

#### 4.2.2 Telas de cadastro do animal de estimação

Abaixo temos as telas de cadastro do animal de estimação, onde o mesmo enviará uma foto para realizar a identificação da raça do animal.



Figura 49 – Telas de identificação de raça.

**Fonte:** Autoria Própria (2022).

Assim posteriormente caso perca o animal, ser notificado ou pelo sistema, ou pelos usuários que encontraram animais semelhantes ao animal perdido, permitindo troca de informação entre os mesmo.

### 4.2.3 Telas de visualização dos animais perdidos

Abaixo é possível ver as telas respectivas á visualização de quais animais ainda estão perdidos, onde é possível cadastrar um novo animal, assim como visualizar aqueles que ainda estão na situação de "perdidos".

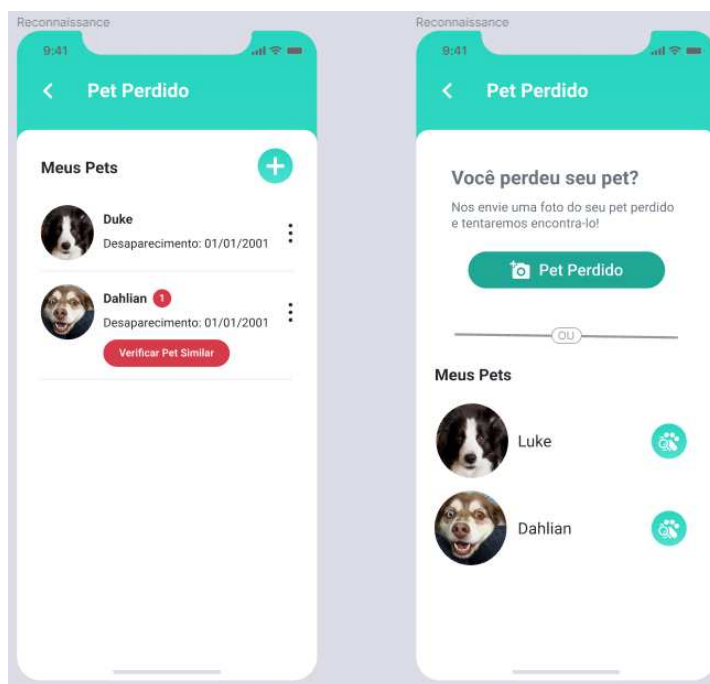


Figura 50 – Telas de listagem de animais perdidos.

**Fonte:** Autoria Própria (2022).

Ao sistemar executar a máquina de estado finito e identificar este usuário como possível dono de uma dos animais encontrados, será realizado a marcação para alertá-lo sobre o encontro do mesmo.



#### 4.2.4 Telas de notificação dos animais perdidos pelo sistema

Abaixo é possível visualizar o fluxo que ocorre ao identificar um animal na rua, a partir da tela de listagem dos usuários ou a tela de login do sistema, é possível tirar uma foto de um animal.

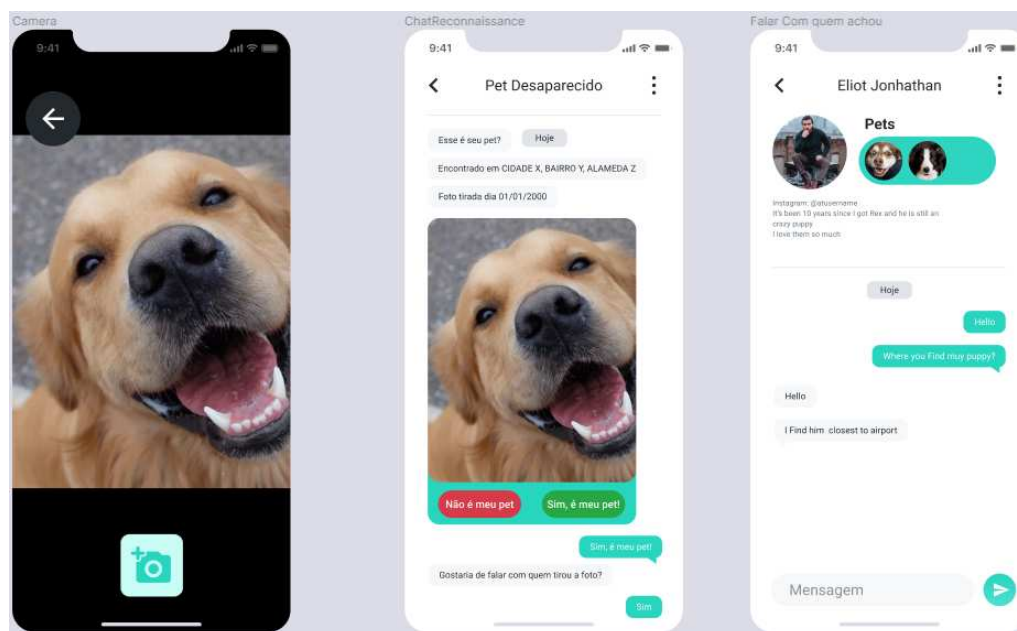


Figura 51 – Telas de notificação de usuário.

**Fonte:** Autoria Própria (2022).

O sistema irá por meio da máquina de estado finita localizar o dono que perdeu um animal com aquelas características e o mesmo poderá confirmar se o animal encontrado é realmente dele, e entrar em contato direto com o usuário que encontrou o animal.

#### 4.2.5 Telas de Chat e Perfil do usuário

Abaixo temos a tela de listagem dos usuários conforme contato anterior entre os mesmos, através da localização de um animal perdido, sua geolocalização do usuário, podendo filtrar por usuário a qual foi contatado anteriormente. Junto a isto, após realizar a listagem, podemos ver abaixo a tela de chat entre os usuários, onde os mesmos poderão trocar informação após o processo de notificação pelo sistema.

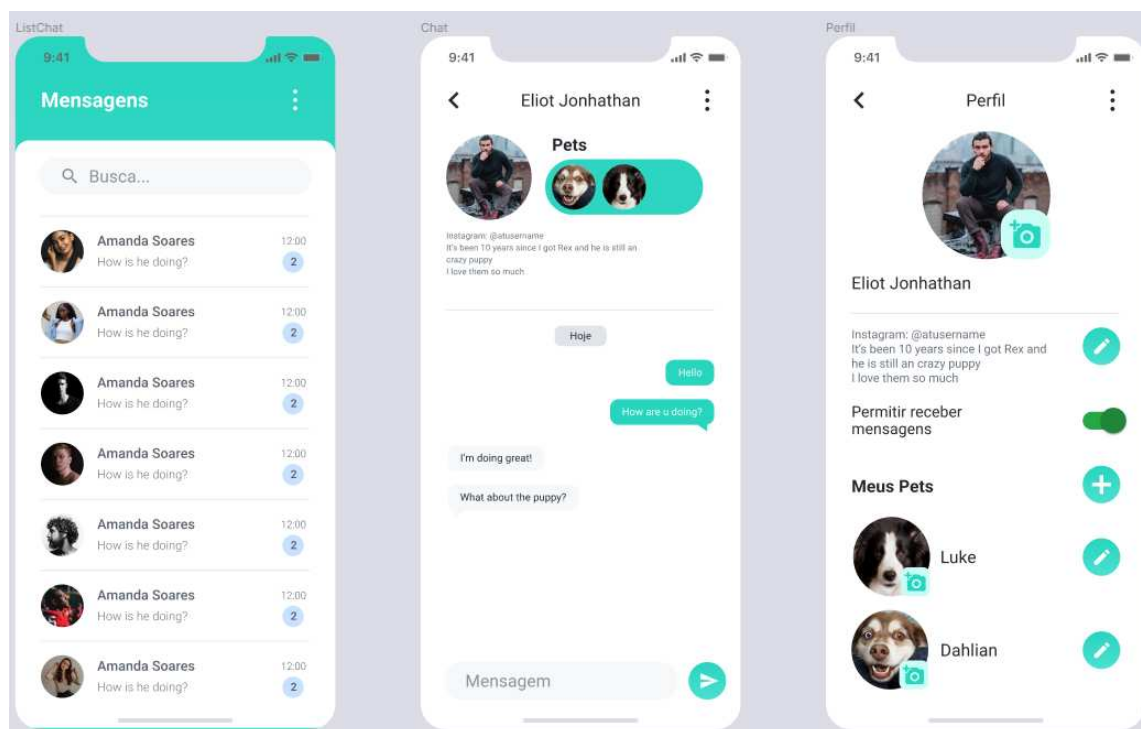


Figura 52 – Telas de Chat e Perfil do usuário.

**Fonte:** Autoria Própria (2022).

Contando também temos a tela de perfil onde o usuário pode ativar e desativar o recebimento de mensagens, ficando disponíveis ou não para o recebimento de mensagens de outros usuários realizada pelo algoritmo, além de poder cadastrar um novo animal em seu perfil.

### 4.3 Validação junto a Lei Geral de Proteção de dados

A LGPD busca promover a transparência e responsabilidade dos controladores de dados, obrigando-os a seguir as diretrizes estabelecidas pelo decreto. Isso inclui fornecer informações claras sobre como os dados pessoais são coletados, usados, armazenados e compartilhados, além de estabelecer responsabilidades para garantir que as informações pessoais sejam devidamente protegidas e não sejam usadas de forma inadequada. Outra importante medida prevista pela lei é a obrigatoriedade de o titular dos dados pessoais ser informado sobre o uso de seus dados pessoais, a qualquer momento. (BRASIL, 2019)

"[...] Art. 7º O tratamento de dados pessoais somente poderá ser realizado nas seguintes hipóteses: I - mediante o fornecimento de consentimento pelo titular; [...] Art. 8º O consentimento previsto no inciso I do art. 7º desta Lei deverá ser fornecido por escrito ou por outro meio que demonstre a manifestação de vontade do titular. § 1º Caso o consentimento seja fornecido por escrito, esse deverá constar de cláusula destacada das demais cláusulas contratuais. § 2º Cabe ao controlador o ônus da prova de que o consentimento foi obtido consoante o disposto nesta Lei. [...] Art. 17. Toda pessoa natural tem assegurada a titularidade de seus dados pessoais e garantidos os direitos fundamentais de liberdade, de intimidade e de privacidade, nos termos desta Lei.[...]"

Para o vigente trabalho atender as normas da Lei Geral de Proteção de Dados, será necessário vincular um termo de uso de dados sobre o aplicativo, devido ao fato que determinadas partes da aplicação observarem de maneira direta as mensagens enviadas pelo usuário. Como demonstrado nas telas do aplicativo exibidas abaixo.



Figura 53 – Tela de confirmação do animal.

**Fonte:** Autoria Própria (2022).

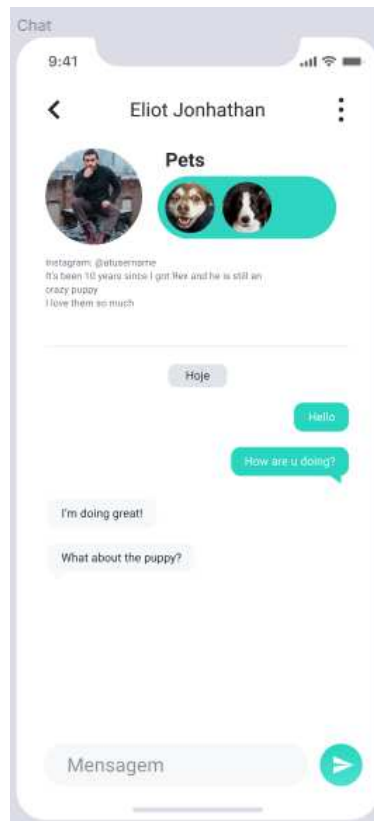


Figura 54 – Tela de conversa entre donos.

**Fonte:** Autoria Própria (2022).

Essa necessidade, é necessária principalmente pelas ações de confirmação de encontro do animal perdido pelo dono do animal, onde é aberto um sistema de conversa entre o sistema e o usuário, ou entre outros usuários para confirmação se o animal encontrado é realmente segundo o dono, o animal a qual o mesmo perdeu.

## 4.4 Testes de qualidade de código

Com o intuito de atestar a qualidade do software desenvolvido, foi utilizado o software Sonarqube, sendo o mesmo instalado na própria máquina onde o código foi escrito, gerenciando bugs e o próprio projeto a partir da *CLI*<sup>6</sup> do *Sonarqube*.

Antes de exibir os resultados da análise, é necessário descrever segundo sua documentação oficial ([SONARQUBE, 2022](#)) o *Sonarqube* como um software com a proposta de ajudar a identificar problemas de codificação como bugs, vulnerabilidades e violações de qualidade do código. A ferramenta também fornece métricas de qualidade do código para ajudá-lo a entender o nível de qualidade do seu código. O SonarQube também oferece recursos de análise de código para detectar problemas de segurança em tempo real, como erros de input e injeção de *SQL*, e oferece recursos de controle de versão para rastrear mudanças no seu código e monitorar o desempenho de cada versão.



Figura 55 – Ícone Sonarqube.

**Fonte:** Sonarqube ([SONARQUBE, 2022](#)).

---

<sup>6</sup> CLI: interface de linha de comando

Abaixo segue os resultados realizados pelo *Sonarqube* para os projetos das funções e APIS deste trabalho, infelizmente o *Sonarqube* não possui suporte para a linguagem *Dart*, apesar de existirem pacotes para a execução do mesmo nesta linguagem feita pela comunidade, a adição do plugin da linguagem no computador testado gerou erros internos por incompatibilidade com novos processadores *M1* da fabricante *Apple*.

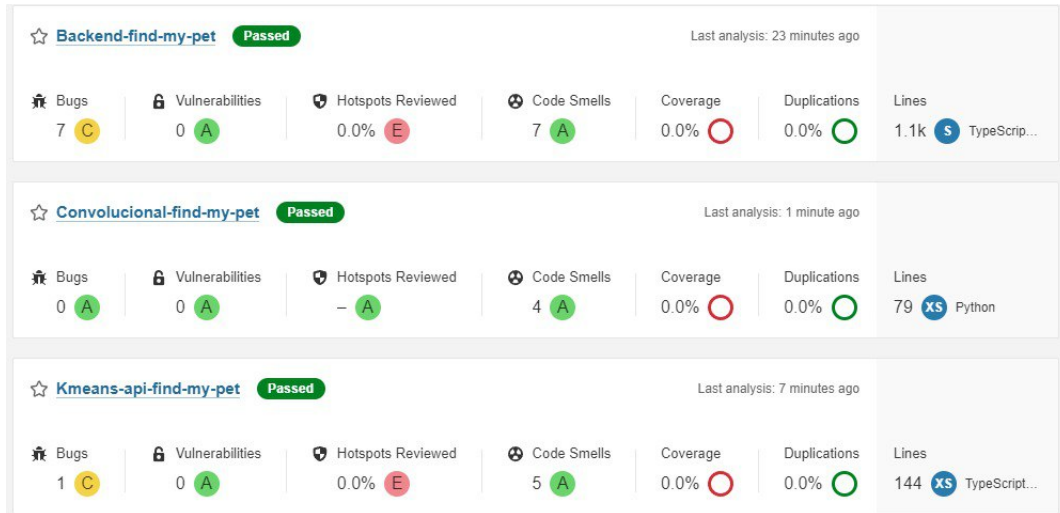


Figura 56 – Tela de conversa entre donos.

**Fonte:** Autoria Própria (2022).

Foram encontrados alguns *bugs*<sup>7</sup> no sistema das funções do *Firebase*, na imagem acima intitulado *Backend-find-my-pet*, e API onde é realizada o agrupamento dos usuários, porém os bugs encontrados se referem a versões mais atualizadas de bibliotecas que podem ser utilizadas no sistema atribuindo uma nota "C", a qual descreve postos que devem ser levados com atenção. Contudo, essas atualizações de bibliotecas que não foi acatada nestes sistemas a primeiro momento devido à incompatibilidade de algumas bibliotecas entre si, questão esta que deve ser melhor analisada como um trabalho futuro.

<sup>7</sup> bugs: termo utilizado para referir a falhas no sistema

Em relação ao *Code Smell*<sup>8</sup>, em todas as aplicações foram identificados alguns pontos de melhora, porém ainda assim foi atribuída a nota "A", estando dentro dos padrões aceitáveis a serem estabelecidos.

## 4.5 Ressalvas da aplicação

É possível obter falhas através da rede neural, tentando identificar outros animais ou objetos dentro da rede, onde a mesma fará a atribuição de uma raça para o input enviado, esta peculiaridade sendo possível resolver fazendo validação do input enviado antes de identificar a raça.

---

<sup>8</sup> Code Smell: refere-se a qualquer trecho de código que pode ser alterado para melhorar a escrita

## 5 Conclusão

Este trabalho tem como proposta o desenvolvimento de uma solução para os problemas relacionados a perda de animais de estimação utilizando redes neurais convolucionais, apresentando uma pesquisa bibliográfica e descrevendo os processos metodológicos e desenvolvimento da aplicação.

Focando no desenvolvimento desta solução, foi constatado o enfrentamento de alguns problemas em relação ao treinamento da rede neural e seus parâmetros utilizados visando identificar os animais perdidos. Pois, para ser uma solução viável, o ideal seria realizar o treinamento das redes para a identificação de mais características dos animais, visando principalmente o fato que a população brasileira tem preferência por cachorros *vira-latas*, fato este descrito no "*Petcenso*" relatado pela reportagem do Jornal Band (BAND, 2022) onde constata que 40% dos brasileiros preferem essa categoria de animal. Este fato dificulta a implementação desta solução, devido aos *vira-latas* serem cães de raça desconhecida, que não possuem um pedigree definido. Embora não se saiba ao certo a sua origem, são normalmente resultados de um cruzamento entre diversas raças ou mesmo cruzamentos entre outros *vira-latas*. Geralmente, possuem características diferentes, como pelagem, tamanho, peso, formato da cabeça e comportamento, impossibilitando uma identificação precisa dos mesmos via rede neural.

Além do problema relatado anteriormente sobre os *vira-latas*, a aplicação contempla atualmente apenas um tipo de animal, sendo estes os cachorros, para a vigente solução ser aplicada como uma solução válida, seria ideal contemplar mais tipos de animais corriqueiros no cotidiano da população, como gatos, entre outros.

Ao finalizar o desenvolvimento, foi realizado um estudo de caso disponibilizado para 107 pessoas dos estados do Tocantins, Maranhão, Pernambuco, Bahia, Rio de Janeiro e São Paulo com o intuito de recolher opiniões de maneira anônima para validação e viabilidade do método aplicado para o desenvolvimento da aplicação. Obtendo assim respostas que auxiliam na melhoria da aplicação, meios mais comuns utilizados atualmente para contemplar o problema de animais perdidos e viabilidade de uma aplicação deste tipo.

Conclui-se que, apesar da solução apresentar uma nova forma de comunicação visando a problemática aplicada, para a proposta apresentada ser realmente viável, será necessário adequar a aplicação para o reconhecimento de mais parâmetros dos animais perdidos tanto como outros tipos de animais.



## 5.1 Trabalhos futuros

Sugestão de próximos passos que podem ser implementados em trabalhos futuros:

- Correção de bugs do sistema.
- Implementação de novas redes neurais ou mais parâmetros na identificação dos animais.
- Criação de um termo de uso do aplicativo visando adequação as normas da LGPD.
- Implementação da solução para mais tipos de animais.

# Referências

ADMIN, F. *Firebas Admin*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.npmjs.com/package/firebase-admin>>.

ATENSTAEDT, R. Word cloud analysis of the bjgp: 5 years on. *British Journal of General Practice*, 2017.

AUTH, F. *Firebase Auth*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/firebase\\_auth](https://pub.dev/packages/firebase_auth)>.

AXIOS. *Axios*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://axios-http.com/ptbr/docs/intro>>.

BAND. *Brasileiros elegem o vira-lata a raça de cachorro preferida do país*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.band.uol.com.br/noticias/jornal-da-band/ultimas/brasileiros-elegem-o-vira-lata-a-raca-de-cachorro-preferida-do-pais-16503721>>.

BONATO, P. V. *Maquinas de Estado Elementos de Lógica Digital II*. 2020. [Online; accessed 22-novembro-2022]. Disponível em: <[http://wiki.icmc.usp.br/images/7/7d/Aula\\_5\\_-\\_StateMachine.pdf](http://wiki.icmc.usp.br/images/7/7d/Aula_5_-_StateMachine.pdf)>.

BRASIL. Lei n.º 13.709, de 14 de agosto de 2018. *Diário Oficial [da] República Federativa do Brasil*, Brasília, DF, 2019. ISSN 1677-7042. Disponível em: <[https://www.planalto.gov.br/ccivil\\_03/\\_Ato2019-2022/2019/Lei/L13853.htm#art1](https://www.planalto.gov.br/ccivil_03/_Ato2019-2022/2019/Lei/L13853.htm#art1)>.

CORPORATION, I. *Diagramas de Atividades*. 2021. [Online; accessed 22-maio-2022]. Disponível em: <<https://www.ibm.com/docs/pt-br/rational-soft-arch/9.7.0?topic=diagrams-activity>>.

DEEPLARNINGBOOK. *Introdução as Redes Neurais Convolucionais*. 2022. [Online; accessed 22-novembro-2021]. Disponível em: <<https://www.deeplearningbook.com.br/introducao-as-redes-neurais-convolucionais/>>.

DIO. *Dio*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://pub.dev/packages/dio>>.

DOCKER. *Docker*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://www.docker.com/>>.

EDUCATION, I. C. *Redes Neurais*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <[ibm.com/br-pt/cloud/learn/neural-networks](https://ibm.com/br-pt/cloud/learn/neural-networks)>.

FAKER. *Faker*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://github.com/faker-js/faker>>.

FASTAPI. *Fastapi*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://fastapi.tiangolo.com/>>.

FIREBASE. *Firebase Core*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/firebase\\_core](https://pub.dev/packages/firebase_core)>.

FIRESTORE, C. *Cloud Firestore*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/cloud\\_firestore](https://pub.dev/packages/cloud_firestore)>.

FISTAROL, J. R. *APLICATIVO PARA AUXILIAR A ENCONTRAR LARES PARA OS ANIMAIS DE RUA: SavingPet*. [S.l.]: Instituto Federal de Santa Catarina, 2018.

FLORINDO, J. B. *Redes Neurais Convolucionais*. 2019. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.ime.unicamp.br/~jbflorindo/Teaching/2018/MT530/T10.pdf>>.

FSM, E. *Edium Fsm*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.npmjs.com/package/@edium/fsm>>.

FUNCTIONS, F. *Firebase Functions*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.npmjs.com/package/firebase-functions>>.

GEOLOCATOR. *Geolocator*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://pub.dev/packages/geolocator>>.

GOOGLE. *Cloud Functions for Firebase*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://firebase.google.com/products/functions?gclid=ds&gclid=ds&gclid=CILTsPSUgvgCFdVmgQodngJLw>>.

GOOGLE. *Firebase*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://firebase.google.com/?hl=pt>>.

GOOGLE. *Firebase Serviços*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://firebase.google.com/docs?hl=pt>>.

GOOGLE. *Google*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/google\\_sign\\_in](https://pub.dev/packages/google_sign_in)>.

GOOGLE. *Google Cloud*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://cloud.google.com/why-google-cloud?hl=pt-br>>.

GOOGLE. *Guia avançado do Inception v3*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=pt-br>>.

GUEDES, G. T. *UML 2-Uma abordagem prática*. [S.l.]: Novatec Editora, 2018. 17–87 p.

HUGO, V. *Pet.me*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <[https://play.google.com/store/apps/details?id=com.pet.vitorcuogo.petapp&hl=pt\\_BR&gl=US](https://play.google.com/store/apps/details?id=com.pet.vitorcuogo.petapp&hl=pt_BR&gl=US)>.

HUMAIRA, H.; RASYIDAH, R. *Determining The Appropriate Cluster Number Using Elbow Method for K-Means Algorithm*. 2020. [Online; accessed 20-maio-2022]. Disponível em: <[https://www.researchgate.net/publication/339670247\\_Determining\\_The\\_Appropriate\\_Cluster\\_Number\\_Using\\_Elbow\\_Method\\_for\\_K-Means\\_Algorithm](https://www.researchgate.net/publication/339670247_Determining_The_Appropriate_Cluster_Number_Using_Elbow_Method_for_K-Means_Algorithm)>.

IBGE. Instituto brasileiro de geografia e estatística.pesquisa nacional de saúde: Informações sobre domicílios, acesso e utilização dos serviços de saúde: Brasil, grandes regiões e unidades da federação. *Rio de janeiro*, 2019.

IBGE. Uso de internet, televisão e celular no brasil. *IBGE Educa*, 2019.

JUNIOR, V. Determinação das métricas usuais a partir da matriz de confusão de classificadores multiclasse em algoritmos inteligentes nas ciências do movimento humano. *Revista CPAQV - Centro de Pesquisas Avançadas em Qualidade de Vida*, 2022. Disponível em: <<http://www.cpaqv.org/revista/CPAQV/ojs-2.3.7/index.php?journal=CPAQV&page=article&op=view&path\%5B\%5D=939>>.

JUSBRASIL. *Brasil tem 30 milhões de animais abandonados*. 2022. [Online; accessed 22-maio-2022]. Disponível em: <<https://anda.jusbrasil.com.br/noticias/100681698/brasil-tem-30-milhoes-de-animais-abandonados>>.

KAGGLE. *Kaggle*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.kaggle.com/competitions/dog-breed-identification/data>>.

KENJI, B. *Machine Learning para Leigos*. 2019. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.venturus.org.br/machine-learning-para-leigos/>>.

KERSCHBAUMER, P. R. *Introdução a Máquinas de Estado Finita em linguagem C*. Instituto Federal Catarinense, 2021. [Online; accessed 24-novembro-2021]. Disponível em: <<https://professor.luzerna.ifc.edu.br/ricardo-kerschbaumer/wp-content/uploads/sites/43/2021/04/Maquinas-de-Estado.pdf>>.

KHOSLA, A. et al. *Novel Dataset for Fine-Grained Image Categorization: Stanford Dogs*. [S.l.]: Computer Science Department, Stanford University, Stanford, CA, 2011.

KMEANS, T. *Thing Kmeans*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://github.com/thi-ng/umbrella/tree/develop/packages/k-means>>.

LAKATOS, M. de Andrade Marcon e E. M. *Fundamentos de Metodologia Científica*. [S.l.]: Atlas, 2017.

LIMA, M. et al. *Introdução ao reconhecimento de imagens*. 2020. [Online; accessed 22-novembro-2022]. Disponível em: <<https://lamfo-unb.github.io/2020/12/05/Captcha-Break/>>.

LOUREIRO, R. *Pesquisa revela os aplicativos de mensagens mais utilizados no Brasil*. Revista Exame, 2019. [Online; accessed 24-novembro-2021]. Disponível em: <<https://exame.com/tecnologia/pesquisa-revela-os-aplicativos-de-mensagens-mais-utilizados-no-brasil/>>.

MARINHO, L. H. *Iniciando com Flutter Framework - Desenvolva aplicações móveis no Dart Side!* [S.l.]: CASA DO CÓDIGO, 2020. ISBN 978-65-86110-26-5.

MARTIN, R. *Clean Architecture: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series)*. [S.l.]: Pearson Education, 2017. 213–222 p. (Traduzido por Yhan).

MESSAGING, F. *Firebase Messaging*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/firebase\\_messaging](https://pub.dev/packages/firebase_messaging)>.

NAPOLI, M. L. *Beginning flutter (a hands on guide to app development) || adding the firebase and firestore backend*. John Wiley & Sons, Inc, p. 376–384, 2019.

NUMPY. *Numpy*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://numpy.org/>>.

OFICIAL, D. P. *Dart overview - Documentação Oficial / Dart Pub Oficial*. 2011. [Online; accessed 14-novembro-2021].

OFICIAL, F. *FAQ Flutter - Documentação Oficial / Flutter Oficial*. 2017. [Online; accessed 14-novembro-2021]. Disponível em: <<https://docs.flutter.dev/resources/faq>>.

PÁDUA, W. d. Engenharia de software: fundamentos, métodos e padrões. *Rio de janeiro: LTC*, 2003.

PETZ. *Petz*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.petz.com.br/>>.

PETZ. *Raças de cachorro: saiba como elas são definidas*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.petz.com.br/blog/racas/caes-racas/racas-de-cachorro/>>.

PICKER, I. *Image Picker*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/image\\_picker](https://pub.dev/packages/image_picker)>.

PUPZ. *Pupz*. 2022. [Online; accessed 22-maio-2022]. Disponível em: <<https://meupetconectado.com.br/>>.

PYTHON. *Python*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://www.python.org/about/>>.

RASHEVSKAYA, A. *Why Should You Build Your Next App with Flutter?* 2020. [Online; accessed 22-novembro-2022]. Disponível em: <<https://litslink.com/blog/why-should-you-build-your-next-app-with-flutter>>.

RIVERPOD. *A Reactive Caching and Data-binding Framework*. 2022. [Online; accessed 22-maio-2022]. Disponível em: <<https://riverpod.dev/>>.

ROCHA, J. S. R. D. “Cadê meu bichinho?”: *Um sistema georreferenciado para encontrar animais de estimação perdidos*. [S.l.]: Universidade Federal do Rio Grande do Sul, 2019.

SAMMUT, G. I. W. C. *Encyclopedia of Machine Learning*. [S.l.]: Springer, 2010.

SANDLER, M. et al. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. 2019. [Online; accessed 22-novembro-2021]. Disponível em: <<https://arxiv.org/abs/1801.04381>>.

SANTANA, L. *CONGRESSO INTERNACIONAL DE DIREITO AMBIENTAL*. Medium, 2004. [Online; accessed 20-maio-2022]. Disponível em: <<http://site.conpedi.org.br/publicacoes/t5ssa9m9/hq1sc2v4/lBFt3b0fyRph0dMv.pdf>>.

SILVA, M. R. da. *Veja como se cadastrar no Flockr, a rede social pet*. Techmundo, 2021. [Online; accessed 25-maio-2021]. Disponível em: <<https://www.tecmundo.com.br/redes-sociais/229674-veja-cadastrar-flockr-rede-social-pet.htm>>.

SINGH, A.; BHADANI, R. *Mobile Deep Learning with TensorFlow Lite, ML Kit and Flutter: Build scalable real-world projects to implement end-to-end neural networks on Android and iOS*. [S.l.]: Packt Publishing, 2020.

SINHA, S. *State in Flutter*. [S.l.]: leanpub.com, 2021.

SONARQUBE. *Sonarqube*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <<https://www.sonarqube.org/>>.

STORAGE, F. *Firebase Storage*. 2022. [Online; accessed 22-novembro-2022]. Disponível em: <[https://pub.dev/packages/firebase\\_storage](https://pub.dev/packages/firebase_storage)>.

STRONGLOOP/IBM. *Documentação do Express Js/ Express FAQ*. 2017. [Online; accessed 21-novembro-2021]. Disponível em: <<https://expressjs.com/en/starter/faq.html>>.

TENSORFLOW. *TensorFlow*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://www.tensorflow.org/?hl=pt-br>>.

TRABALHO, J. B. do. *Conceito: Modelo de Caso de Uso*. 2022. [Online; accessed 22-maio-2022]. Disponível em: <[https://www.trt9.jus.br/pds/pdstrt9/guidances/concepts/use\\_case\\_model\\_CD178AF9.html](https://www.trt9.jus.br/pds/pdstrt9/guidances/concepts/use_case_model_CD178AF9.html)>.

UBUNTU. *CronHowTo*. 2016. [Online; accessed 22-novembro-2022]. Disponível em: <<https://help.ubuntu.com/community/CronHowto>>.

WALLACE, E. *Prototyping features*. 2022. [Online; accessed 20-maio-2022]. Disponível em: <<https://www.figma.com/prototyping/>>.

# Anexos

## 5.2 Anexo A - Questionário para Validação de uma Aplicação de auxílio na localização de animais perdidos

Para criação do estudo de caso e validação da ideia proposta para solução do problema relacionado a perda de animais de estimação, foi criado um formulário utilizando a plataforma *Google Formulários*, contando com 13 perguntas e um total de 107 respostas de diversos estados do país.

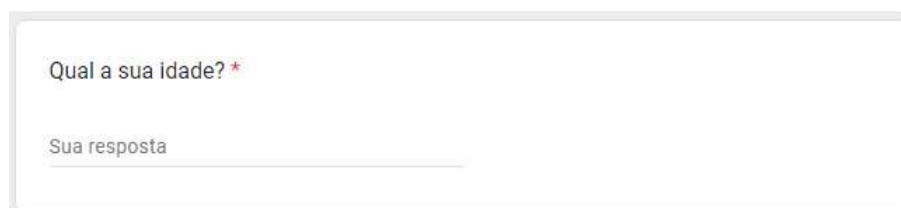


The image shows the header section of a Google Form. At the top is a photograph of a small, light-brown dog with floppy ears. Below the photo, the title 'Aplicação de auxílio na localização de animais perdidos' is displayed in a large, bold, black font. Underneath the title, a paragraph explains the form's purpose: 'O seguinte formulário tem por finalidade acadêmica recolher, de maneira anônima, informações para a validação de um estudo de casos para uma aplicação de localização de animais perdidos baseados em captura de foto e localização dos usuários.' At the bottom of the header, it states: 'Feito pelo aluno Yhan Nunes Leal da Silva, Curso de Sistemas de informação da Universidade Estadual do Tocantins'.

Figura 57 – Cabeçalho do formulário.

**Fonte:** Google Formulários.

### 5.2.1 Perguntas do formulário



The image shows the first question of the Google Form. The question is 'Qual a sua idade? \*' in a black font. Below the question is a text input field with the placeholder text 'Sua resposta' in a light gray font.

Figura 58 – Pergunta 1.

**Fonte:** Google Formulários.

Qual seu gênero? \*

☐ Masculino

☐ Feminino

☐ Outro: \_\_\_\_\_

Figura 59 – Pergunta 2.

**Fonte:** Google Formulários.

Em qual cidade você vive? Colocar como no exemplo: Palmas - TO \*

Sua resposta \_\_\_\_\_

Figura 60 – Pergunta 3.

**Fonte:** Google Formulários.

Você já passou pela experiência de perder um animal de estimação? \*

☐ Sim

☐ Não

Figura 61 – Pergunta 4.

**Fonte:** Google Formulários.

Você acredita que um aplicativo móvel pode auxiliar na localização de animais perdidos? \*

☐ Sim

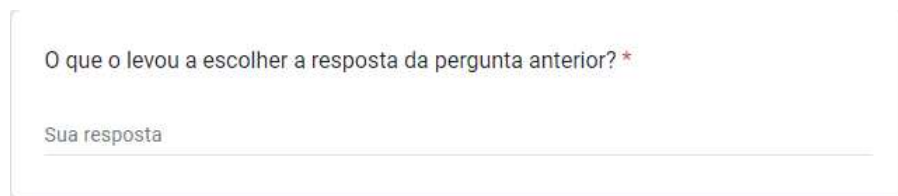
☐ Não

☐ Talvez

Figura 62 – Pergunta 5.

**Fonte:** Google Formulários.



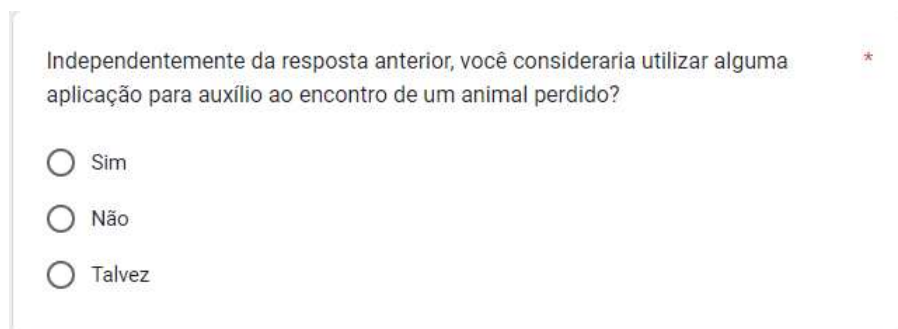
A screenshot of a Google Form question. The question text is "O que o levou a escolher a resposta da pergunta anterior? \*". Below the question is a text input field with the placeholder text "Sua resposta".

O que o levou a escolher a resposta da pergunta anterior? \*

Sua resposta

Figura 63 – Pergunta 6.

**Fonte:** Google Formulários.

A screenshot of a Google Form question. The question text is "Independentemente da resposta anterior, você consideraria utilizar alguma aplicação para auxílio ao encontro de um animal perdido? \*". Below the question are three radio button options: "Sim", "Não", and "Talvez".

Independentemente da resposta anterior, você consideraria utilizar alguma aplicação para auxílio ao encontro de um animal perdido? \*

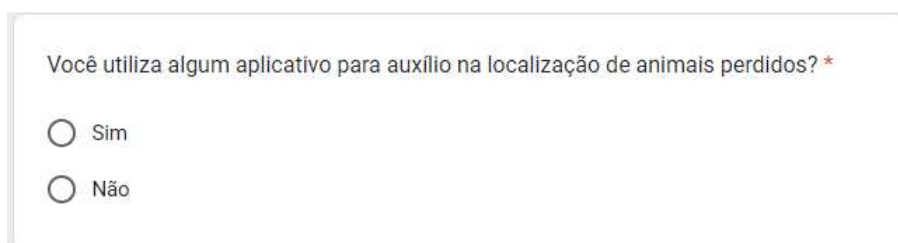
☐ Sim

☐ Não

☐ Talvez

Figura 64 – Pergunta 7.

**Fonte:** Google Formulários.

A screenshot of a Google Form question. The question text is "Você utiliza algum aplicativo para auxílio na localização de animais perdidos? \*". Below the question are two radio button options: "Sim" and "Não".

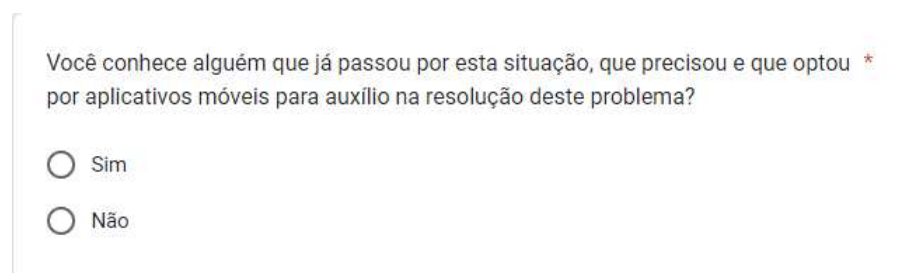
Você utiliza algum aplicativo para auxílio na localização de animais perdidos? \*

☐ Sim

☐ Não

Figura 65 – Pergunta 8.

**Fonte:** Google Formulários.

A screenshot of a Google Form question. The question text is "Você conhece alguém que já passou por esta situação, que precisou e que optou por aplicativos móveis para auxílio na resolução deste problema? \*". Below the question are two radio button options: "Sim" and "Não".

Você conhece alguém que já passou por esta situação, que precisou e que optou por aplicativos móveis para auxílio na resolução deste problema? \*

☐ Sim

☐ Não

Figura 66 – Pergunta 9.

**Fonte:** Google Formulários.

Que tipo de funcionalidades você acredita ser importante em um aplicativo de localização de animais perdidos? \*

Sua resposta

Figura 67 – Pergunta 10.

**Fonte:** Google Formulários.

Quais os principais meios de comunicação que você se depara quando o assunto são animais perdidos, seja ela campanha de doação de animais encontrados ou auxílio na localização dos mesmos? \*

☐ TV

☐ Instagram

☐ Whatsapp

☐ Facebook

☐ Jornal

☐ Rádio

☐ Twitter

☐ Outro: \_\_\_\_\_

Figura 68 – Pergunta 11.

**Fonte:** Google Formulários.

Que característica você considera pertinente para a identificação de um animal perdido? \*

☐ Raça

☐ Cor da pelagem

☐ Localização

☐ Outro: \_\_\_\_\_

Figura 69 – Pergunta 12.

**Fonte:** Google Formulários.

Quais os principais concorrentes que você consegue visualizar para aplicativos de localização de animais perdidos?

Sua resposta

Figura 70 – Pergunta 13.

**Fonte:** Google Formulários.

5.2.2 Respostas do formulário

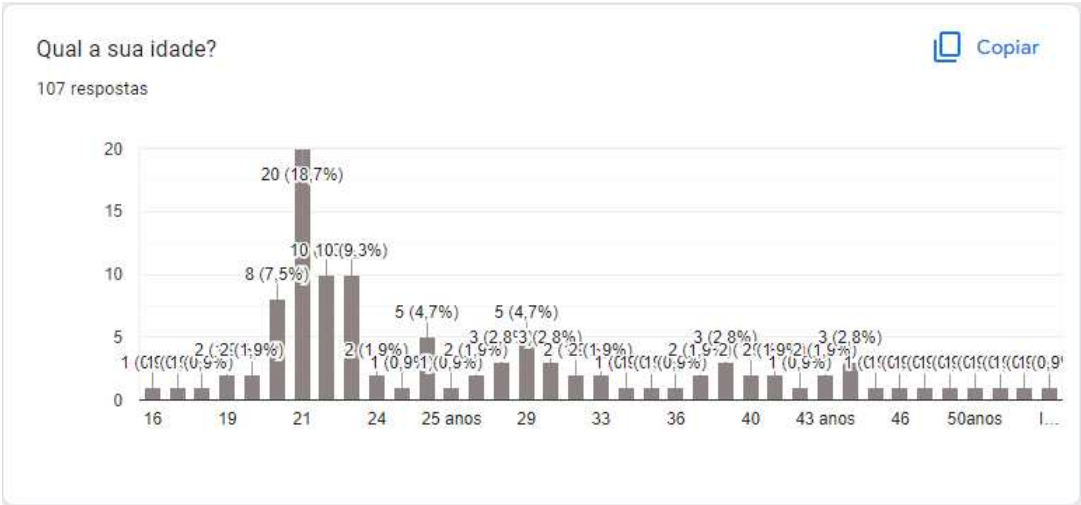


Figura 71 – Resposta 1.

Fonte: Google Formulários.

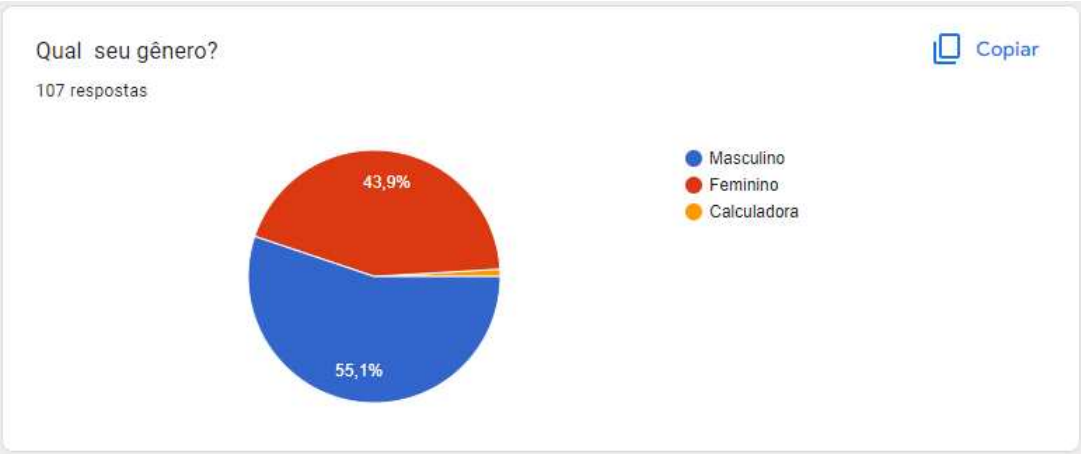


Figura 72 – Resposta 2.

Fonte: Google Formulários.

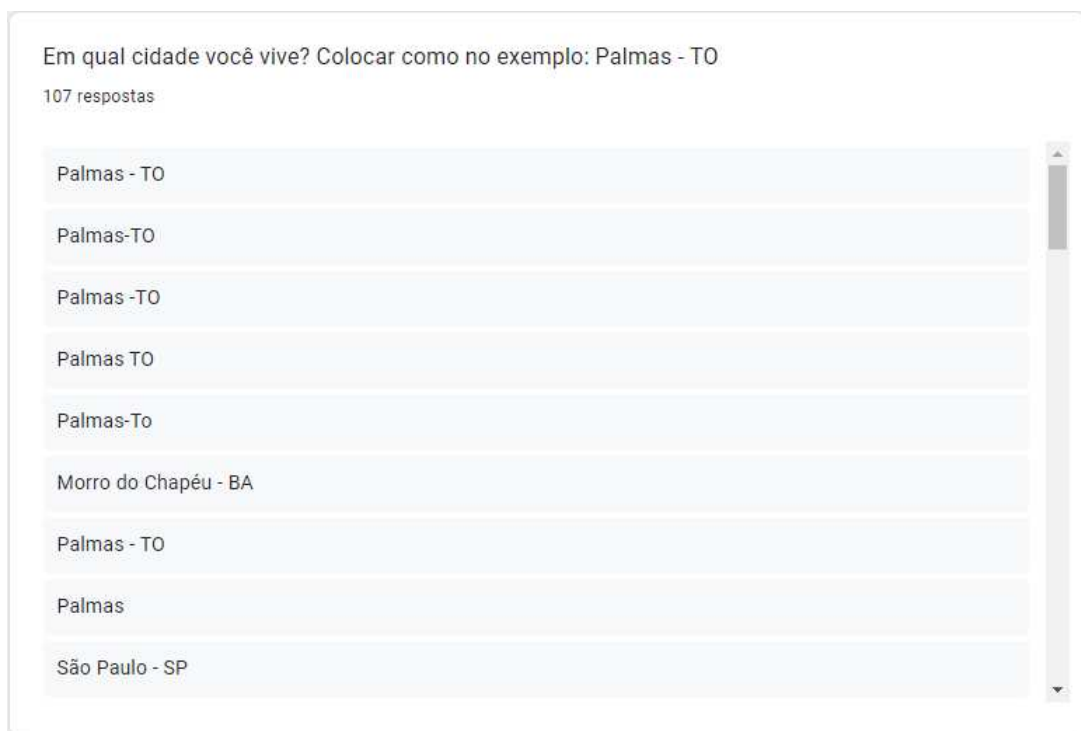


Figura 73 – Resposta 3.

**Fonte:** Google Formulários.

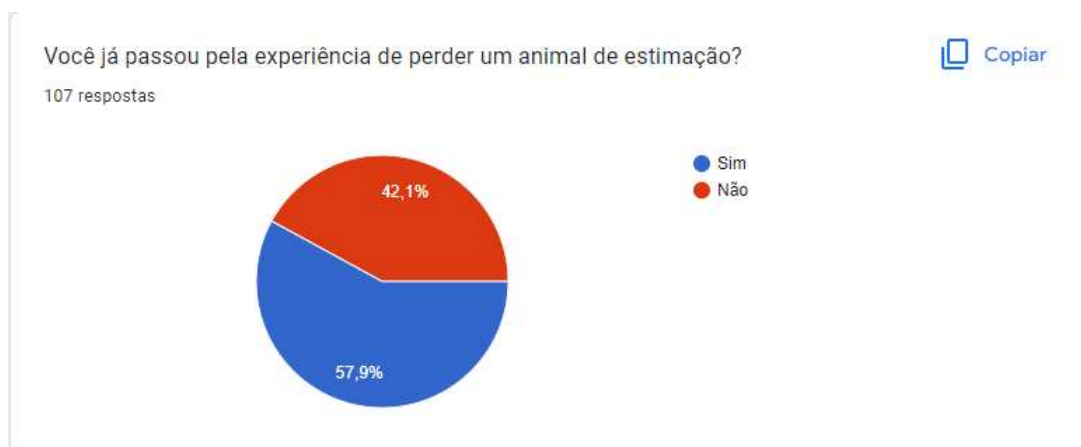


Figura 74 – Resposta 4.

**Fonte:** Google Formulários.

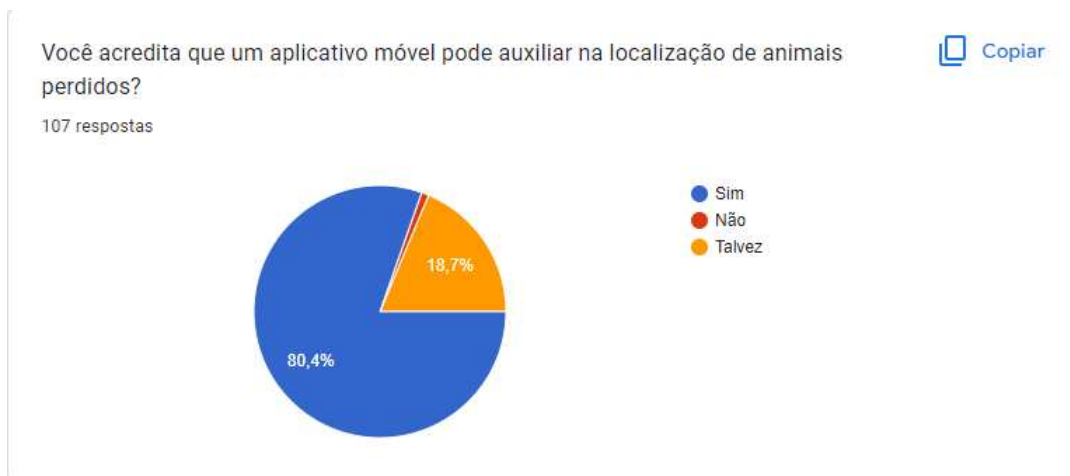


Figura 75 – Resposta 5.

**Fonte:** Google Formulários.

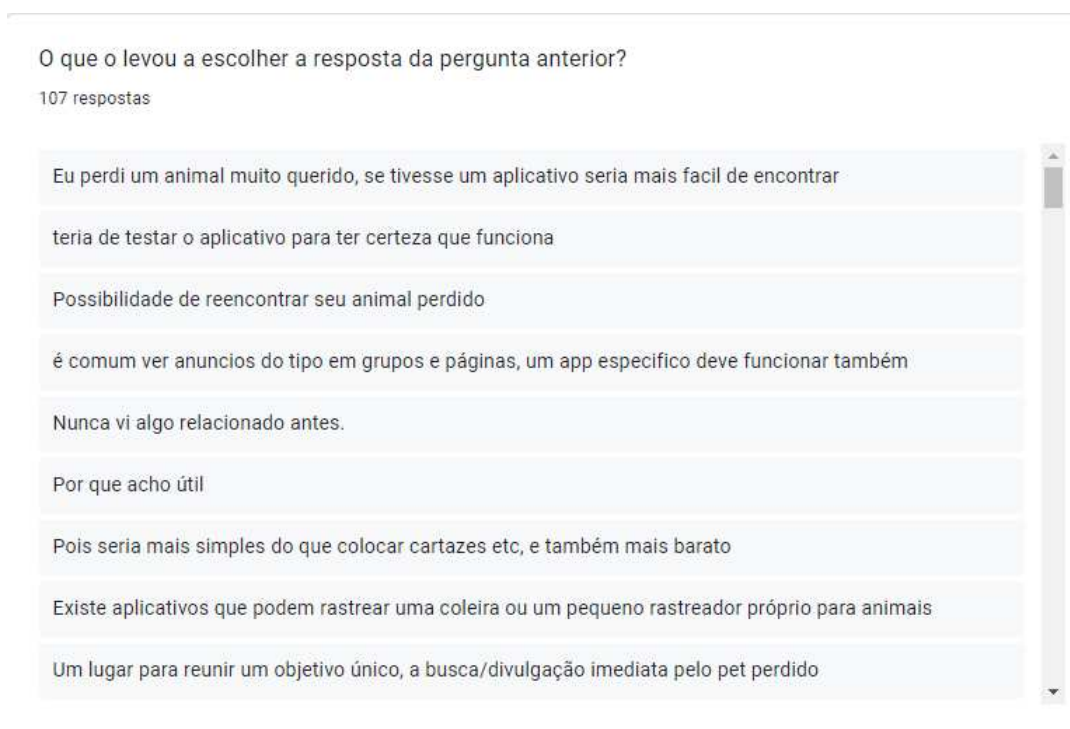


Figura 76 – Resposta 6.

**Fonte:** Google Formulários.

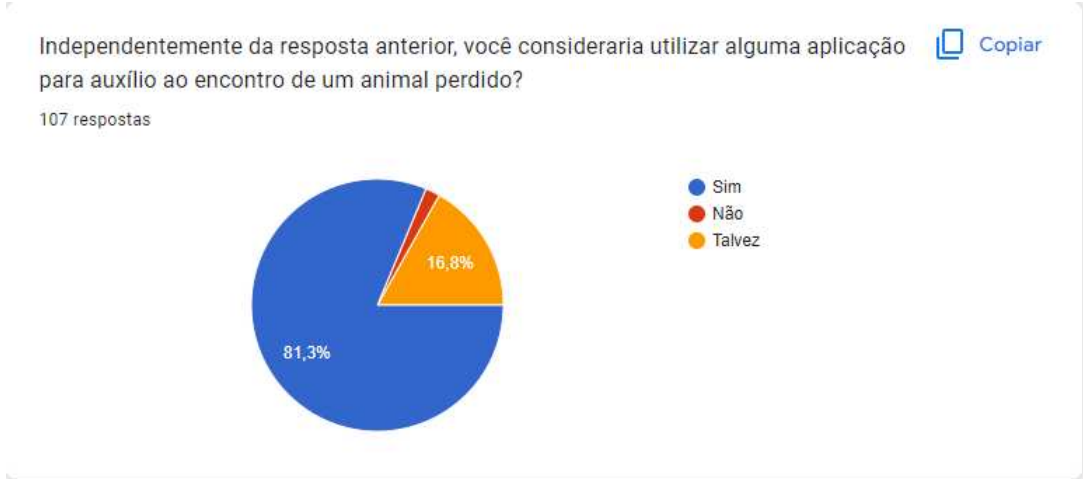


Figura 77 – Resposta 7.

Fonte: Google Formulários.

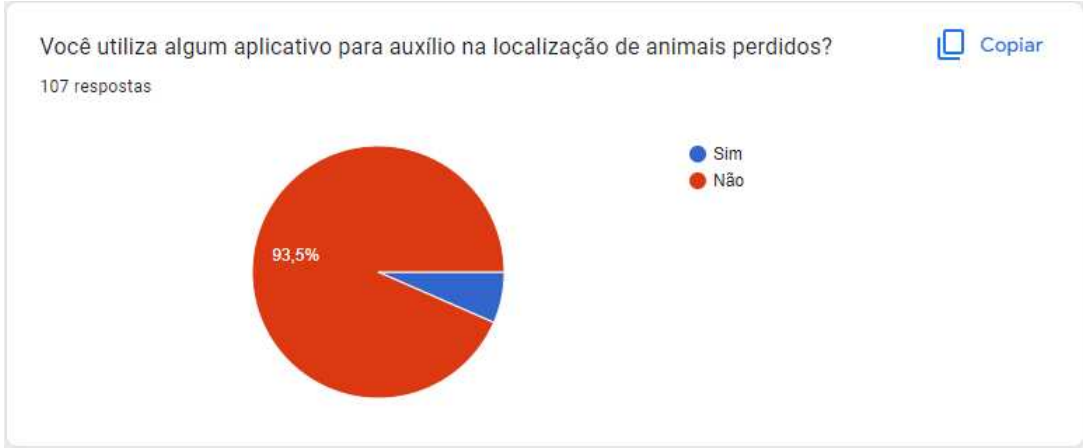


Figura 78 – Resposta 8.

Fonte: Google Formulários.



Figura 79 – Resposta 9.

Fonte: Google Formulários.

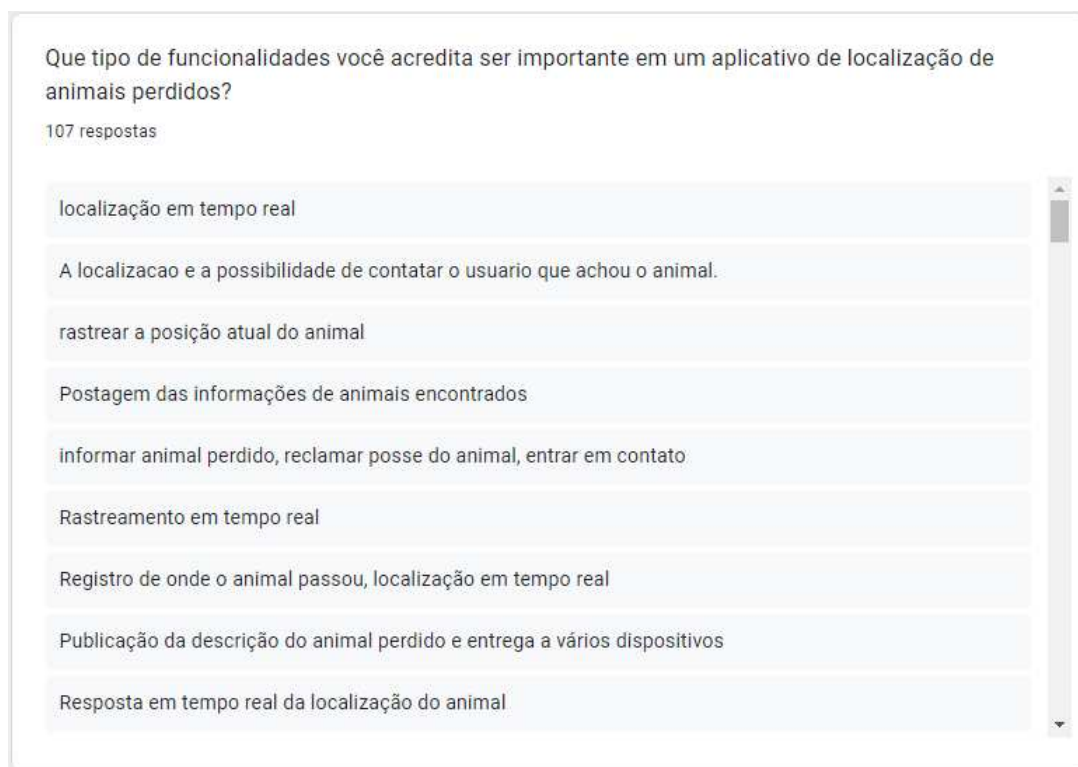


Figura 80 – Resposta 10.

**Fonte:** Google Formulários.

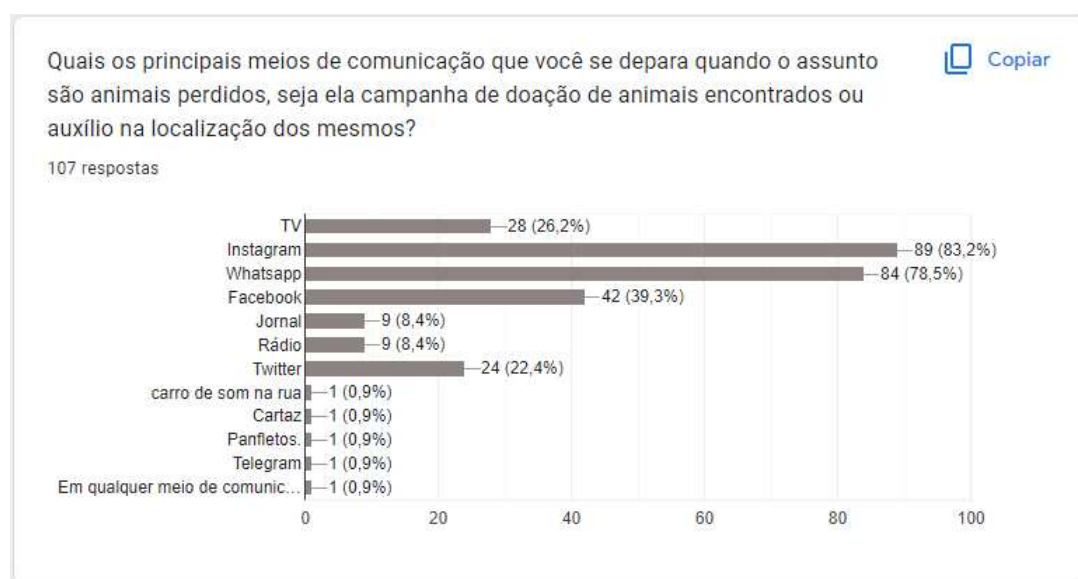


Figura 81 – Resposta 11.

**Fonte:** Google Formulários.



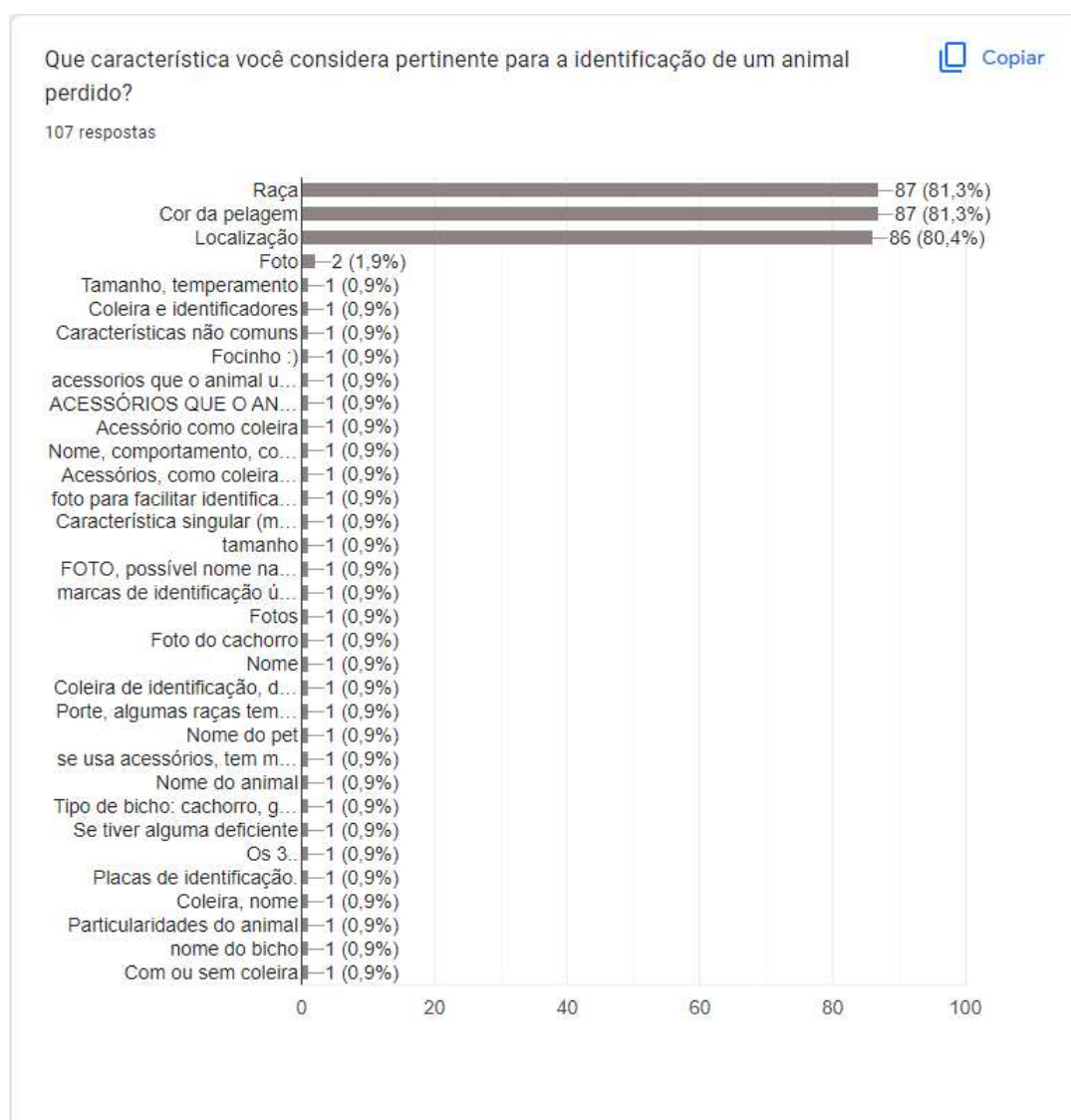


Figura 82 – Resposta 12.

Fonte: Google Formulários.

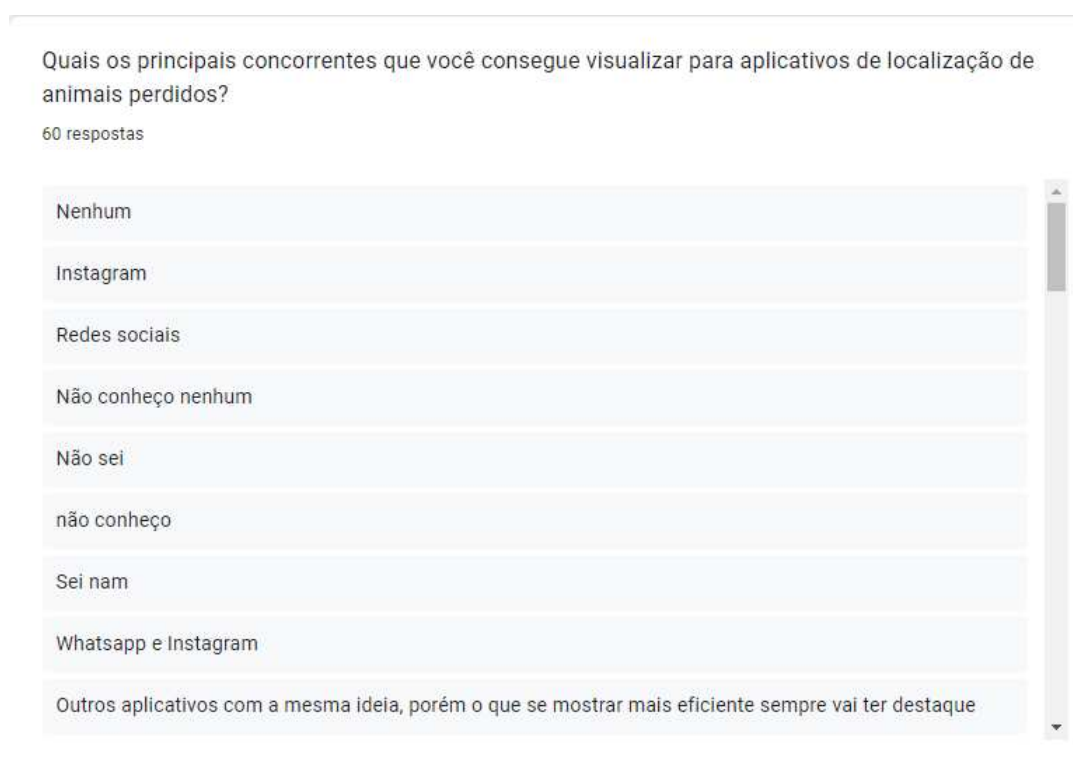


Figura 83 – Resposta 13.

**Fonte:** Google Formulários.

Para melhorar a visualização das respostas obtidas pelos usuários do formulário, foi utilizado o método *Nuvem de Palavras*, cujo propósito é destacar os termos mais utilizados em textos ou resultados de pesquisa. Este método é utilizado por diversas instituições com o propósito de apresentação de resultados ou recomendações de especialistas, como pode ser visto no trabalho de "Robert Attenstaedt" pela Livraria Nacional de Medicina do Reino Unido, onde o mesmo gera uma *Nuvem de Palavras* para visualização de uma pesquisa realizada em um Jornal Britânico que descreve as principais práticas físicas de pessoas. (ATENSTAEDT, 2017)

- Que característica você considera pertinente para a identificação de um animal perdido?
- Que tipo de funcionalidades você acredita ser importante em um aplicativo de localização de animais perdidos?

[illegible]

**Fonte:** Autoria Própria.

Já em relação às funcionalidades desejadas, a maioria das respostas dos usuários destacam principalmente funcionalidades voltadas a comunicação e localização do animal, sendo estas propostas principais do projeto.



Figura 85 – Nuvem de palavras 2.

Fonte: Autoria Própria.

## Apêndices

# APÊNDICE A – Requisitos do Sistema

Um requisito pode ser definido como uma condição ou potencialidade de que o usuário necessita para resolver ou atingir um objetivo, ou uma condição que um sistema, componente ou produto deve possuir para satisfazer qual seja a exigência para resolução de dado problema (PÁDUA, 2003).

## A.0.1 Requisitos Funcionais

Requisitos funcionais descrevem as funções que o produto ou sistema deverá realizar em benefícios dos usuários. Onde essas funções são detalhadas para elucidação de seu funcionamento, como atores envolvidos, referências e prioridades (PÁDUA, 2003).

Abaixo segue os requisitos funcionais do sistema.

<b>RF001</b>	<b>Registrar usuário</b>
<b>Descrição:</b> O sistema deve permitir o cadastro de um novo usuário, recebendo as seguintes informações: e-mail, senha, foto de perfil e nome	
<b>Atores:</b> Usuário	
<b>Entradas e Pré-Condições:</b> Nenhuma	
<b>Saídas e Pós-Condições:</b> O usuário é cadastrado no sistema	
<b>Prioridade:</b> Essencial	

Figura 86 – Requisito Funcional 01

**Fonte:** Autoria Própria (2022).

RF002	<a href="#">Logar com o Google</a>
<p><b>Descrição:</b> O sistema deve permitir que o usuário faça 'login' com o Google</p> <p><b>Atores:</b> Usuário</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no (Google)</p> <p><b>Saídas e Pós-Condições:</b> O usuário faz 'login' no sistema</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 87 – Requisito Funcional 02

**Fonte:** Autoria Própria (2022).

RF003	Cadastrar animais de estimação
<p><b>Descrição:</b> O sistema deve permitir que o cadastro de um animal de estimação, passadas as seguintes informações: nome, foto e descrição</p> <p><b>Atores:</b> Usuário</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema</p> <p><b>Saídas e Pós-Condições:</b> Um animal de estimação é associado ao usuário</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 88 – Requisito Funcional 03

**Fonte:** Autoria Própria (2022).

<b>RF004</b>	Confirmar raça do animal
<p><b>Descrição:</b> O sistema deve identificar a raça do animal cadastrado consoante a foto realizada no cadastro do <b>RF003</b>, e deve permitir que o usuário confirme a raça selecionando ofertando a opção dele selecionar outra raça de animal caso não corresponda com o que foi previamente selecionado pelo sistema, ou retirar outra foto, de modo a realizara identificação novamente.</p> <p><b>Atores:</b> Usuário</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema e ter realizado o cadastro de um animal de estimação</p> <p><b>Saídas e Pós-Condições:</b> A raça do animal é confirmada pelo usuário.</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 89 – Requisito Funcional 04

**Fonte:** Autoria Própria (2022).

<b>RF005</b>	Identificar animal perdido
<p><b>Descrição:</b> O sistema deve permitir a identificação de um animal pelo aplicativo, através da foto tirada pelo usuário, seja ele estando logado ou não junto a localização do aparelho do mesmo, baseados no <b>RF001</b>, <b>RF006</b> .</p> <p><b>Atores:</b> Usuário</p> <p><b>Entradas e Pré-Condições:</b> O usuário ter o aplicativo baixado</p> <p><b>Saídas e Pós-Condições:</b> A raça do animal é identificada.</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 90 – Requisito Funcional 05

**Fonte:** Autoria Própria (2022).



<b>RF006</b>	<b>Avisar usuários próximos com animais perdidos</b>
<p><b>Descrição:</b> O sistema deve oferecer notificar usuários próximos caso seja identificado um animal de estimação com as características do animal do dono que foi perdido.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema e ter pelo menos um animal de estimação cadastrado.</p> <p><b>Saídas e Pós-Condições:</b> Os usuários próximos são notificados</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 91 – Requisito Funcional 06

**Fonte:** Autoria Própria (2022).

<b>RF007</b>	<b>Solicitar localização</b>
<p><b>Descrição:</b> O sistema deve solicitar o uso localização do usuário.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema.</p> <p><b>Saídas e Pós-Condições:</b> A localização do usuário é recebida</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 92 – Requisito Funcional 07

**Fonte:** Autoria Própria (2022).

<b>RF008</b>	<b>Alterar descrição do usuário</b>
<p><b>Descrição:</b> O sistema deve permitir que o usuário altere sua descrição.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema.</p> <p><b>Saídas e Pós-Condições:</b> A localização do usuário é recebida</p> <p><b>Prioridade:</b> Desejável</p>	

Figura 93 – Requisito Funcional 08

**Fonte:** Autoria Própria (2022).

<b>RF009</b>	<b>Iniciar Conversa</b>
<p><b>Descrição:</b> O sistema deve permitir que o usuário inicie conversas com outros usuários, baseado no <b>RF005</b>, <b>RF006</b> e <b>RF0010</b>.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema.</p> <p><b>Saídas e Pós-Condições:</b> O usuário inicia uma conversa com outro usuário</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 94 – Requisito Funcional 09

**Fonte:** Autoria Própria (2022).

<b>RF0010</b>	<b>Ficar Offline</b>
<p><b>Descrição:</b> O sistema deve permitir que o usuário fique inativo para receber mensagens de outros usuários caso seja encontrado um animal perdido por outro usuário.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema.</p> <p><b>Saídas e Pós-Condições:</b> O usuário bloqueia a comunicação</p> <p><b>Prioridade:</b> Desejável</p>	

Figura 95 – Requisito Funcional 10

**Fonte:** Autoria Própria (2022).

<b>RF0011</b>	<b>Definir animal perdido</b>
<p><b>Descrição:</b> O sistema deve permitir que o usuário defina se um animal está perdido para a realização da busca posteriormente pelo sistema.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir um animal cadastrado no sistema.</p> <p><b>Saídas e Pós-Condições:</b> O usuário define um animal como perdido</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 96 – Requisito Funcional 11

**Fonte:** Autoria Própria (2022).

RF0012	Confirmar animal perdido
<p><b>Descrição:</b> O sistema deve permitir que o usuário confirma que o animal a qual foi notificado nos <b>RF005, RF006 e RF009</b>, seja ou não o animal a qual foi perdido.</p> <p><b>Atores:</b> Sistema</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir um animal cadastrado no sistema.</p> <p><b>Saídas e Pós-Condições:</b> O usuário define um animal como achado</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 97 – Requisito Funcional 12

**Fonte:** Autoria Própria (2022).

## A.0.2 Requisitos Não Funcionais

Requisitos não-funcionais são os aspectos necessários para que o produto atinja a qualidade desejada, incluindo requisitos de qualidade como requisitos de desempenho, requisitos de persistência, e requisitos técnicos (PÁDUA, 2003).

Abaixo segue os requisitos não-funcionais do sistema.

<b>RNF001</b>	Linguagem Programação mobile
<b>Descrição:</b> O sistema deve ser desenvolvido utilizando a linguagem de programação Dart junto ao seu framework Flutter	
<b>Atores:</b> Usuário	
<b>Prioridade:</b> Essencial	

Figura 98 – Requisito Não Funcional 01

**Fonte:** Autoria Própria (2022).

<b>RNF002</b>	Requisito de disponibilidade
<b>Descrição:</b> O sistema deve garantir que o usuário esteja com a localização ativada e esteja conectado a uma rede de 'internet' estável	
<b>Atores:</b> Usuário	
<b>Prioridade:</b> Essencial	

Figura 99 – Requisito Não Funcional 02

**Fonte:** Autoria Própria (2022).

RNF003	Requisito de usabilidade
<p><b>Descrição:</b> O sistema deve ter uma 'interface' intuitiva para diferentes categorias de usuários.</p> <p><b>Atores:</b> Usuário</p> <p><b>Entradas e Pré-Condições:</b> O usuário possuir uma conta no sistema</p> <p><b>Saídas e Pós-Condições:</b> O usuário faz 'login' no sistema</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 100 – Requisito Não Funcional 03

**Fonte:** Autoria Própria (2022).

RNF004	Requisito de disponibilidade
<p><b>Descrição:</b> O sistema deve estar disponível para os usuários nas plataformas android e IOS</p> <p><b>Atores:</b> Usuário</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 101 – Requisito Não Funcional 04

**Fonte:** Autoria Própria (2022).

RNF005	Requisito de armazenamento
<p><b>Descrição:</b> O sistema deve armazenar as informações no serviço cloud do Firebase, utilizando seus serviços de Storage e cloud firestore.</p> <p><b>Atores:</b> Usuário</p> <p><b>Prioridade:</b> Essencial</p>	

Figura 102 – Requisito Não Funcional 05

**Fonte:** Autoria Própria (2022).

<b>RNF006</b>	Requisito de autenticação
<b>Descrição:</b> O sistema deve exigir autenticação para acessar suas funcionalidades.	
<b>Atores:</b> Usuário	
<b>Prioridade:</b> Essencial	

Figura 103 – Requisito Não Funcional 06

**Fonte:** Autoria Própria (2022).

<b>RNF007</b>	Requisito de identificação do animal
<b>Descrição:</b> O sistema deve utilizar a linguagem python para treinar a rede neural para identificar a raça dos animais de estimação.	
<b>Atores:</b> Usuário	
<b>Prioridade:</b> Essencial	

Figura 104 – Requisito Não Funcional 07

**Fonte:** Autoria Própria (2022).