

INSTITUTO FEDERAL
Goiás
Câmpus Anápolis

Mateus Amelio Inacio

DESENVOLVIMENTO DE APLICAÇÃO MÓVEL PARA A INFORMATIZAÇÃO DA RESENHA DE EQUINOS E MUARES

Anápolis-GO

2025

Mateus Amelio Inacio

DESENVOLVIMENTO DE APLICAÇÃO MÓVEL PARA A INFORMATIZAÇÃO DA RESENHA DE EQUINOS E MUARES

Projeto de desenvolvimento apresentado ao curso de Bacharelado em Ciência da Computação, como requisito para obtenção do grau final na disciplina de Trabalho de conclusão de curso.

Instituto Federal de Goiás - Câmpus Anápolis

Orientador: Me. Alexandre Bellezi José

Anápolis-GO

2025

Dados Internacionais de Catalogação na Publicação (CIP)

I35d Inacio, Mateus Amelio

Desenvolvimento de aplicação móvel para a informatização da resenha de equinos e muares. / Mateus Amelio Inacio. – Anápolis: IFG, 2026.

67 f. il. color.

Orientador: Prof. Me. Alexandre Bellezi José.

Trabalho de Conclusão de Curso – Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Campus Anápolis – Bacharelado em Ciência da Computação, 2026.

1. Resenha de equinos e muares. 2. Aplicação móvel. 3. Engenharia de software.. 4. Identificação animal. 5. Computação móvel.

I. José, Alexandre Bellezi. (orient.).

II. Título.

CDD 005.36

Ficha catalográfica elaborada pelo bibliotecário Matheus Rocha Piacenti CRB1/02992

IFG Campus Anápolis

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[x] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor:
Matrícula:
Título do Trabalho:

Autorização - Marque uma das opções

1. (x) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ____/____/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

gov.br Documento assinado digitalmente
MATEUS AMELIO INACIO
Data: 02/04/2026 18:34:25-0300
Verifique em https://validar.iti.gov.br

_____, ____/____/____.
Local Data

_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

ATA DE DEFESA DE MONOGRAFIA

Na presente data realizou-se a sessão pública de defesa da Monografia intitulada **DESENVOLVIMENTO DE APLICAÇÃO MÓVEL PARA A INFORMATIZAÇÃO DA RESENHA DE EQUINOS E MUARES**, sob orientação de Alexandre Bellezi Jose , apresentada pelo aluno **Mateus Amelio Inacio (20181060140048)** do Curso **Bacharelado em Ciência da Computação (Câmpus Anápolis)**. Os trabalhos foram iniciados às **10:30** do dia **03/03/2026** pelo Professor presidente da banca examinadora, constituída pelos seguintes membros:

- **Alexandre Bellezi Jose** (Presidente)
- **Hugo Vinicius Leao e Silva** (Examinador Interno)
- **Sergio Daniel Carvalho Canuto** (Examinador Interno)

A banca examinadora, tendo terminado a apresentação do conteúdo da Monografia, passou à arguição do candidato. Em seguida, os examinadores reuniram-se para avaliação e deram o parecer final sobre o trabalho apresentado pelo aluno, tendo sido atribuído o seguinte resultado:

[X] Aprovado [ ] Reprovado Nota : 7,8

Observação / Apreciações:

Proclamados os resultados pelo presidente da banca examinadora, foram encerrados os trabalhos e, para constar, eu **Alexandre Bellezi Jose** lavrei a presente ata que assino juntamente com os demais membros da banca examinadora.

Documento assinado eletronicamente por:

- **Alexandre Bellezi Jose**, em **03/03/2026 12:12:39** com chave **6b318d72171311f184eb005056a537a4**.
- **Sergio Daniel Carvalho Canuto**, em **03/03/2026 13:27:25** com chave **dcdddbec171d11f1926c005056a537a4**.
- **Hugo Vinicius Leao e Silva**, em **03/03/2026 14:24:46** com chave **dfd0f3c2172511f1a705005056a537a4**.

Este documento foi emitido pelo SUAP. Para comprovar sua autenticidade, faça a leitura do QrCode ao lado ou acesse https://suap.ifg.edu.br/comum/autenticar_documento/ e informe os dados a seguir.

Tipo de Documento: Ata de Projeto Final

Data da Emissão: 03/03/2026

Código de Autenticação: ceb235

Lista de ilustrações

Figura 1 – Formulário para marcação de sinais e marcas do animal. 12
Figura 2 – Resenha descritiva. 12
Figura 3 – Exemplo de resenha preenchida. 13
Figura 4 – Albinóide: a e b. 17
Figura 5 – Variações da pelagem alazã: a) tostada; b) cereja; c) sobre baia; d) amarilha. 18
Figura 6 – a) Castanha clara; b) Castanha escura. 19
Figura 7 – Pelagem zaina. 19
Figura 8 – Pelagem baia. 20
Figura 9 – Pelo de rato. 20
Figura 10 – Pelagens tordilha: a) negra; b) apatacada; c) clara; d) ruça. 21
Figura 11 – Pelagem tordilha: e) cardã; f) pedrez. 21
Figura 12 – Variações: ruão, lobuna e rosilha, respectivamente. 22
Figura 13 – a) Baia de pampa; b) Preto de pampa; c) Pampa de preto; d) Pampa de alazão. 22
Figura 14 – Pelagem preta. 23
Figura 16 – Cavalo crinalvo. 24
Figura 15 – Particularidades especiais dos equinos. 25
Figura 17 – Redemoinho na linha dos olhos e espiga, respectivamente. 26
Figura 18 – Particularidades nos membros. 27
Figura 19 – Exemplo *Kanban* com tarefas. 28
Figura 20 – Coleções de animais e proprietários no *Firestore*. 31
Figura 21 – Visão geral da arquitetura do Cordova. 32
Figura 22 – Comparação entre o comportamento de navegação com rota interna em aplicação React empacotada com Cordova. 33
Figura 23 – Diagrama de alto nível: interação do usuário, marcação por regiões e persistência no *Firestore*. 43
Figura 24 – Exemplo de documento no *Firestore* para uma marca (campos principais). 44
Figura 25 – Tela de login do aplicativo. 45
Figura 26 – Formulário de cadastro de proprietários. 46
Figura 27 – Formulário de cadastro de animais. 47
Figura 28 – Tela de controle da resenha. 48
Figura 29 – Descrição da resenha. 50
Figura 30 – Tela de visão geral de regiões zootécnicas. 51
Figura 31 – Lado direito com marca. 52

Lista de tabelas

Tabela 1 – Classificação das pelagens. 17
Tabela 2 – Marcas da cabeça. 24
Tabela 3 – Análise de requisitos funcionais do sistema de resenha (consolidada a partir do documento-base). 39

Lista de abreviaturas e siglas

HTML	HyperText Markup Language
AIE	Anemia Infecciosa Equina
AGRODEFESA	Agência Goiana de Defesa Agropecuária
API	Application Programming Interface
APK	Android Package Kit
CLI	Command-Line Interface
GIMP	GNU Image Manipulation Program
HTTP	Hypertext Transfer Protocol
IBGE	Instituto Brasileiro de Geografia e Estatística
LTS	Long-Term Support
PNSE	Programa Nacional de Sanidade de Equídeos
MAPA	Ministério da Agricultura, Pecuária e Abastecimento
NPM	Node Package Manager
NoSQL	Not Only SQL
PESE	Programa Estadual de Sanidade do Cavalo
RF	Requisito Funcional
RNA	Ácido Ribonucleico
TO-DO	To Do
WIP	Work in Progress
GPS	Global Positioning System
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
CSS	Cascading Style Sheets
SDK	Software Development Kit

SQL	Structured Query Language
iOS	iPhone Operating System
SPA	Single Page Application
URL	Uniform Resource Locator
SGBD	Sistema Gerenciador de Banco de Dados
CRMV	Conselho Regional de Medicina Veterinária
UML	Unified Modeling Language
CRUD	Create, Read, Update and Delete
CSP	Content Security Policy
PDF	Portable Document Format
README	Readme (arquivo de instruções do repositório)
W3C	World Wide Web Consortium

Sumário

1 INTRODUÇÃO 10

Introdução 10
1.1 Justificativa 13
1.2 Objetivos 14
1.2.1 Objetivo geral 14
1.2.2 Objetivos específicos 14
1.3 Estrutura do trabalho 14

2 REFERENCIAL TEÓRICO 16
2.1 Identificação de equinos e muares 16
2.1.1 Tipos de pelagens e suas tonalidades 16
2.1.2 Tipos de pelagens e suas descrições 16
2.1.2.1 Pelagens simples e uniformes 16
2.1.2.2 Simples e uniformes com crina, cauda e extremidades pretas: 19
2.1.2.3 Pelagens compostas 20
2.1.2.4 Pelagens Conjugadas 21
2.1.3 Tipos de particularidade 23
2.1.4 Engenharia de Software 27
2.1.4.1 Desenvolvimento de aplicações móveis 28
2.1.4.2 Banco de dados Firestore 29
2.1.4.3 React 30
2.1.4.4 Apache Cordova 31
2.1.4.5 Navegação com React Router (HashRouter) em ambiente híbrido 32
2.1.4.6 Mapa de imagens HTML 34
2.1.4.6.1 Responsividade e Acessibilidade. 34

3 METODOLOGIA 35
3.1 Etapas do desenvolvimento 35
3.2 Ferramentas e métodos utilizados 36

4 DESENVOLVIMENTO DO SOFTWARE 38
4.1 Levantamento de requisitos 38
4.2 Modelagem (diagramas UML, casos de uso) 40
4.3 Arquitetura do sistema 41
4.4 Implementação 44

4.4.1 Autenticação e acesso ao sistema . . . 45
4.4.2 Cadastro e gestão de proprietários . . . 46
4.4.3 Cadastro e gestão de animais . . . 46
4.4.4 Fluxo de resenha e controle de preenchimento . . . 48
4.4.5 Mapeamento visual por regiões zootécnicas . . . 49
4.4.6 Interação com áreas clicáveis e registro de marcas . . . 51
4.4.7 Responsividade dos mapas e consistência do clique . . . 52
4.4.8 Navegação entre telas em ambiente híbrido . . . 53
4.4.9 Persistência e continuidade do preenchimento . . . 53
4.5 Desafios . . . 53
4.5.1 Acessibilidade das áreas clicáveis . . . 53
4.5.2 Empacotamento do React como aplicação Cordova . . . 54
4.5.3 Responsividade dos mapas de imagem e recálculo de coordenadas . . . 55

5 RESULTADOS . . . 59
5.1 Retomada dos objetivos e atendimento alcançado . . . 59
5.1.1 Objetivo geral . . . 59
5.1.2 Objetivos específicos . . . 59
5.1.3 Comparação entre objetivos e resultados . . . 60
5.2 Resultado principal entregue . . . 60
5.3 Evidências de funcionamento e validação do protótipo . . . 61
5.4 Benefícios gerados . . . 62
5.4.1 Benefícios operacionais (processo de resenha) . . . 62
5.4.2 Benefícios de padronização e qualidade de dados . . . 62
5.4.3 Benefícios de organização, consulta e rastreabilidade . . . 63
5.4.4 Benefícios técnicos (portabilidade/manutenibilidade) . . . 63
5.5 Conclusão e trabalhos futuros . . . 64

REFERÊNCIAS . . . 66

1 Introdução

No Brasil, existe um número significativo de equinos e muares. De acordo com o Ministério da Agricultura (BARBOSA, 2011), essa quantidade de animais coloca o país como detentor do terceiro maior rebanho de equídeos da América Latina, com um total de 7.793.259 animais (IBGE, 2010), em um mercado que movimenta aproximadamente R$ 7,3 bilhões apenas com a produção de cavalos. Em função da expressividade desse rebanho e da relevância econômica do setor, foi implantado, em 1981, pelo Governo Federal, o Programa Nacional de Sanidade de Equídeos (PNSE), com o objetivo de propor estratégias de vigilância epidemiológica e sanitária das principais doenças que acometem esses animais, visando à profilaxia, ao controle e à erradicação dessas enfermidades em todas as Unidades Federativas (Brasil. Ministério da Agricultura e Pecuária, 2022).

Entre as doenças de maior importância sanitária destacam-se a Anemia Infecciosa Equina (AIE), o Mormo e a Influenza Equina. Legalmente, os proprietários devem possuir, especialmente para o transporte dos animais, documentos que comprovem que estes são livres dessas enfermidades ou, no caso da Influenza Equina, que foram devidamente vacinados. A responsabilidade pela realização dos exames e pela correta identificação do animal, por meio de uma resenha completa, recai sobre o médico veterinário.

Assim como apontado por Batista e Carrijo (2016), a AIE é a única doença para a qual é necessário um exame negativo do animal em todo o território nacional, quer o transporte se efetue no estado de origem ou fora dele, salvo nos casos de envio para abate. A anemia infecciosa equina constitui um grande obstáculo ao desenvolvimento da criação de equinos, pois é uma doença infecciosa e incurável que causa danos e prejuízos aos proprietários que dependem do trabalho desses animais e aos criadores envolvidos na melhoria genética do rebanho, além de impedir o acesso ao mercado internacional (ALMEIDA, 2006). Trata-se de uma doença infecciosa crônica de etiologia viral, limitada a equinos, asininos e muares, caracterizada por episódios periódicos de febre, anemia hemolítica, icterícia, depressão, edema e emagrecimento crônico (CARLSON, 2006).

Outra doença que exige exame específico e adequada identificação do animal é o Mormo, definida como uma enfermidade infecciosa causada pela bactéria *Burkholderia mallei*, mais frequente em equinos (cavalos, asnos e mulas), mas que pode também ser contraída por outros mamíferos, como cachorros, gatos, bodes e inclusive o ser humano, caracterizando-se como zoonose. A doença é adquirida, sobretudo, pela ingestão de água ou comida contaminada e é encontrada na América Latina, África e Ásia (MORMO, 2022).

Ainda conforme mostrado por Batista e Carrijo (2016), no caso da gripe (Influenza) equina, os animais utilizados para exposições, leilões e esportes devem portar certificado

de vacinação ou atestado emitido por médico veterinário que comprove a ausência da doença no estabelecimento de origem (BRASIL, 2013). A Influenza equina é uma doença infecciosa do sistema respiratório, causada por um vírus de ácido ribonucleico (RNA) que sofre mutações contínuas, sendo que altas concentrações de animais favorecem o aparecimento de epizootias. É altamente contagiosa e raramente fatal, acometendo cavalos, burros, mulas e outros animais domésticos (BRASIL, 2013). Até o momento, a doença não foi detectada em humanos, embora existam evidências sorológicas de infecção (BRASIL, 2013). A vacinação contra a gripe equina pode reduzir a gravidade e a disseminação da infecção, motivo pelo qual os registros de vacinação do animal são verificados pelos órgãos de fiscalização.

Os órgãos estaduais de defesa agropecuária possuem legislações específicas que complementam a fiscalização federal. No Estado de Goiás, destacam-se o Decreto Estadual nº 5.652, de 6 de setembro de 2002, a Lei Estadual de Saneamento para Animais (Lei nº 13.998, de 13 de dezembro de 2001) e o Programa Estadual de Sanidade do Cavalo (PESE), os quais disciplinam e reforçam as ações de controle sanitário sobre o rebanho equino.

As medidas de controle e prevenção listadas acima são baseadas no princípio de que os equinos devem ser identificados de forma única, de modo a garantir a precisão das medidas sanitárias adotadas. Em alguns casos, como na confirmação de AIE, os animais devem ser sacrificados ou colocados em quarentena, o que torna imprescindível a associação inequívoca entre o laudo laboratorial e o indivíduo ao qual se refere. Além disso, existem outros documentos que exigem identificação única dos animais, como o passaporte do cavalo, considerado um importante documento de identificação.

Para registrar o animal em uma associação de criadores, é realizada uma resenha no momento do pré-registro e outra no momento do registro oficial. O Manual Legislativo - Sanidade Animal, do Ministério da Agricultura, Pecuária e Abastecimento (MAPA), ressalta em seu texto sobre resenha equina: “Procure fazer a resenha o mais fielmente possível” (Ministério da Agricultura, 2016), exemplificando a importância desse instrumento e demonstrando preocupação com os aspectos humanos relevantes para sua elaboração, uma vez que os profissionais envolvidos podem não ter a experiência necessária para identificar com segurança o animal em questão.

De acordo com o manual de identificação de equinos para uso em países da União Europeia, “por identificação entende-se o conjunto único de características que distinguem um indivíduo no universo do seu semelhante e que permite que um indivíduo possa ser reconhecido entre um conjunto da mesma espécie, raça ou família”; quanto mais completa for essa identificação, maior será a possibilidade de o indivíduo ser reconhecido inequivocamente (BARBOSA, 2011, p.8). Tal definição sugere que quanto mais completa for a descrição de um animal, maior será a garantia de que ele seja identificado de forma única. Para a elaboração adequada da resenha, recomenda-se que o animal esteja limpo,

principalmente em seus membros locomotores, devendo ser conduzido à mão em terreno firme (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

Hoje, é tarefa do médico veterinário preencher a ficha de resenha dos animais, não sendo permitido o preenchimento ou a modificação da resenha por parte do proprietário, funcionário ou treinador. A resenha é composta por uma descrição gráfica e uma descrição textual, nas quais são descritas todas as marcas dos animais, conforme ilustrado na Figura 1 e na Figura 2. A Figura 3 apresenta um exemplo de resenha preenchida à mão.

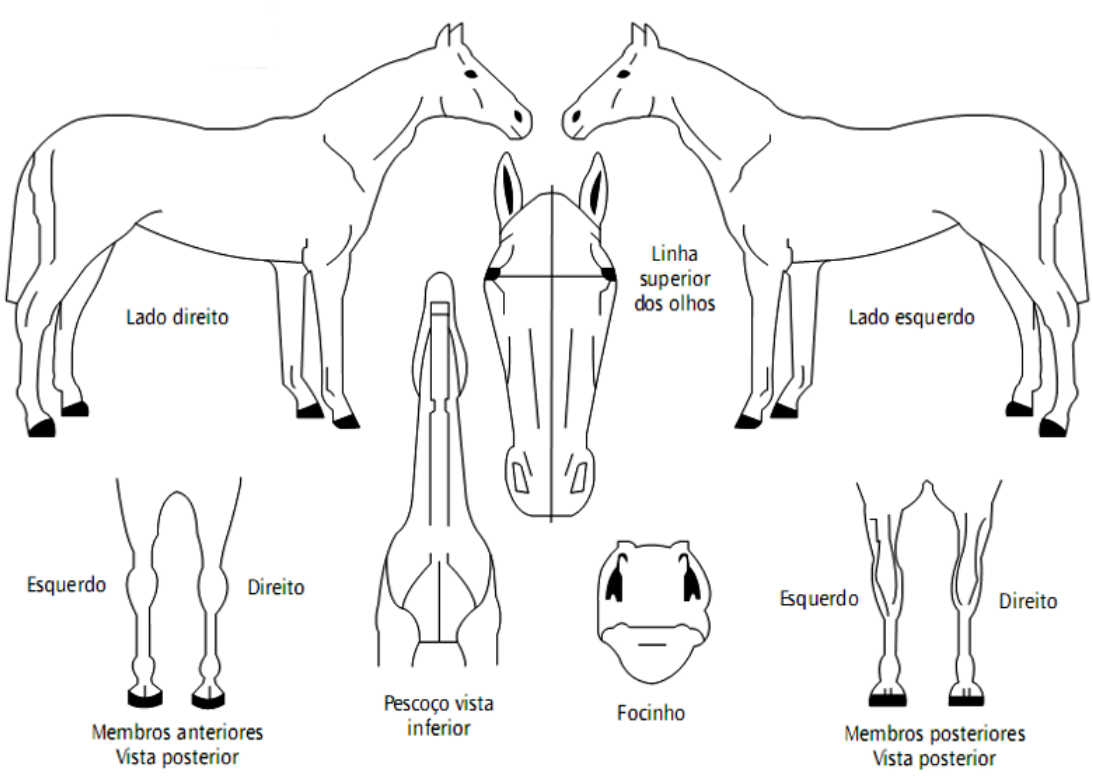


Figura 1 – Formulário para marcação de sinais e marcas do animal.

Fonte: Batista e Carrijo (2016)

RESENHA DESCRITIVA

Descrição do animal:

Figura 2 – Resenha descritiva.

Fonte: Batista e Carrijo (2016)

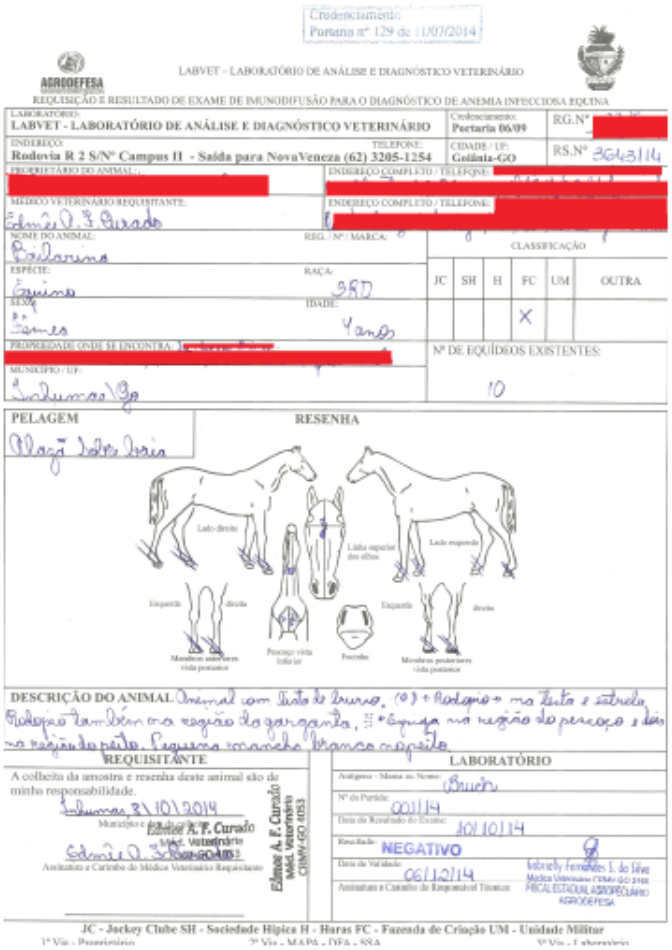


Figura 3 – Exemplo de resenha preenchida.

Fonte: Batista e Carrijo (2016)

1.1 Justificativa

Apesar de sua importância para a sanidade animal e para a rastreabilidade dos equinos e muares, o preenchimento da resenha ainda é realizado, majoritariamente, de forma manual, em formulários em papel. Tal procedimento apresenta pouca praticidade, é mais suscetível a falhas de preenchimento e conservação dos documentos e dificulta a organização, o compartilhamento e a recuperação das informações ao longo do tempo.

Ademais, o Manual Legislativo - Sanidade Animal enfatiza a necessidade de que a resenha seja elaborada "o mais fielmente possível" (Ministério da Agricultura, 2016), evidenciando que a qualidade da identificação depende, em grande medida, da experiência e da habilidade do profissional responsável. Em muitos casos, contudo, esses profissionais podem não possuir treinamento específico para representar, de maneira padronizada e precisa, as particularidades morfológicas e as marcas naturais dos animais, o que aumenta a subjetividade do processo (BATISTA; CARRIJO, 2016).

Devido à pouca praticidade no preenchimento da resenha de equinos e muares de

maneira estritamente manual, este trabalho se justifica pela proposição de uma aplicação móvel voltada a facilitar tal preenchimento e a apoiar o médico veterinário e o público interessado na identificação dos animais por meio de resenha. Com o uso de uma aplicação móvel, é garantida a persistência da resenha em banco de dados, o que contribui para que esses documentos possam ser mantidos, organizados e consultados de forma mais eficiente.

Nesse contexto, a escolha do tema fundamenta-se na necessidade de modernizar o processo de identificação de equinos e muares, reduzindo a subjetividade na elaboração da resenha, aumentando a precisão na identificação e promovendo a integração entre os registros sanitários e os sistemas de informação utilizados por proprietários, criadores, associações de raça e órgãos de defesa sanitária.

1.2 Objetivos

À luz do contexto apresentado, este trabalho estabelece um objetivo geral e alguns objetivos específicos, descritos a seguir.

1.2.1 Objetivo geral

O objetivo geral deste trabalho é desenvolver um aplicativo informatizado para dispositivos móveis, inspirado no trabalho exploratório de engenharia de *software* proposto por Batista e Carrijo (2016), capaz de auxiliar o veterinário na elaboração da resenha de equinos e muares por meio de *tablets* ou *smartphones*, integrando-se ao serviço em nuvem *Google Firebase* para armazenamento e gerenciamento dos dados. Espera-se, com isso, aprimorar a precisão e a confiabilidade da identificação desses animais em forma de resenha.

1.2.2 Objetivos específicos

Os objetivos específicos deste trabalho são:

- Estudar as particularidades da identificação de equinos e muares;
- Criar um banco de dados de imagens de marcas naturais;
- Projetar e implementar um aplicativo Android que se comunique com o banco de dados em nuvem *Firestore*, componente do *Google Firebase*.

1.3 Estrutura do trabalho

Além desta introdução, este trabalho está organizado em capítulos que tratam, de forma articulada, dos aspectos conceituais, metodológicos e práticos relacionados ao

desenvolvimento da solução proposta.

O Capítulo 2 apresenta o referencial teórico que fundamenta o trabalho. São discutidos os conceitos relacionados à identificação de equinos e muares, incluindo pelagens, particularidades morfológicas e procedimentos de resenha, bem como aspectos de engenharia de *software* pertinentes ao desenvolvimento da aplicação, como aplicações móveis, banco de dados *Firestore*, React, Apache Cordova, navegação em ambiente híbrido e uso de mapas de imagem HTML.

O Capítulo 3 descreve a metodologia adotada no desenvolvimento do projeto, caracterizando-o como um trabalho de natureza aplicada. São detalhadas as etapas de desenvolvimento, a organização das atividades (com uso de *Kanban*) e as ferramentas e tecnologias empregadas na implementação da solução.

O Capítulo 4 aborda o desenvolvimento do *software* propriamente dito. Nesse capítulo são apresentados o levantamento de requisitos, a modelagem do sistema, a definição da arquitetura, bem como a implementação das principais funcionalidades, incluindo autenticação, cadastro de proprietários e animais, fluxo de resenha, mapeamento por regiões zootécnicas, persistência em banco de dados e os desafios técnicos enfrentados, como o empacotamento com Cordova e o recálculo de coordenadas dos mapas de imagem.

O Capítulo 5 apresenta os resultados obtidos com a implementação da aplicação, destacando os benefícios operacionais, técnicos e de padronização de dados proporcionados pela informatização da resenha. Ao final, são expostas as considerações finais e conclusões do trabalho.

2 Referencial Teórico

2.1 Identificação de equinos e muares

De acordo com Ministério da Agricultura (2016): "Entende-se por equídeos todos os solípedes domésticos da família Equidae, abrangendo equinos (cavalos, pôneis), asininos (jumentos) e muares (burros e mulas)."

Para Batista e Carrijo (2016), a identificação correta desses animais torna-se complicada até mesmo para especialistas da área, uma vez que não é incomum que pessoas confundam cores, tipos de pelagem e raças desses animais. Por exemplo, a cor é essencial para identificar certas raças e, em diferentes raças, a mesma cor pode receber nomes distintos.

2.1.1 Tipos de pelagens e suas tonalidades

De acordo com o manual da Agência Goiana de Defesa Agropecuária (AGRODEFESA) para Elaboração de Resenha de Equídeos com a Finalidade de Solicitação de Diagnóstico para Anemia Infecciosa Equina e Mormo GOIÁS. Agência Goiana de Defesa Agropecuária (2012), a pelagem é o conjunto de pelos, de uma ou de diversas cores, espalhados pela superfície do corpo e extremidades, em distribuição e disposição variadas, cujo todo determina a cor do animal. O manual indica que as pelagens existentes são classificadas em quatro categorias, sendo que cada categoria possui vários tipos de pelagem, com coloração do revestimento externo distinto e cada tipo tem diversas variedades que são identificadas pelas diferentes tonalidades da coloração do pelo. A Tabela 1 cita as diferentes classes de pelagens.

Ainda sobre as diferentes categorias:

> A pelagem composta possui pelos de duas ou três cores diferentes, distribuídos no corpo do animal. Pelagens simples com crina, cauda e extremidades pretas apresentam coloração uniforme na cabeça, no pescoço e no tronco, mas possuem crina, cauda e extremidades pretas. Já as pelagens conjugadas possuem duas cores diferentes, porém a cor branca está sempre presente Batista e Carrijo (2016).

2.1.2 Tipos de pelagens e suas descrições

2.1.2.1 Pelagens simples e uniformes

- **Branca**: Composta exclusivamente de pelos brancos e pele pigmentada, é uma pelagem rara. Porém existe a variedade branca pseudo-albina, conhecida como gázeo

Tabela 1 – Classificação das pelagens.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

Categoria	Tipo	Variedades
Simples e uniformes	Branca	Maltinta e Azeviche
	Alazã	pseudo-albina
	Preta	diversas
Simples e uniformes com crina, cauda e extremidades pretas	Castanha	diversas
	Baia	diversas
	Pelo de rato	claro e escuro
Compostas	Tordilha	diversas
	Rosilha	Diversas
	Lobuna	Clara e escura
	Ruão	Clara e escura
Conjugadas	Pampa	Diversas
	Leopardo	Diversas
	Mantado	Diversas
	Nevado	Diversas

ou pombo, que se caracteriza pela presença de pelos brancos em pele quase que totalmente despigmentada, como ilustrado na Figura 4 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

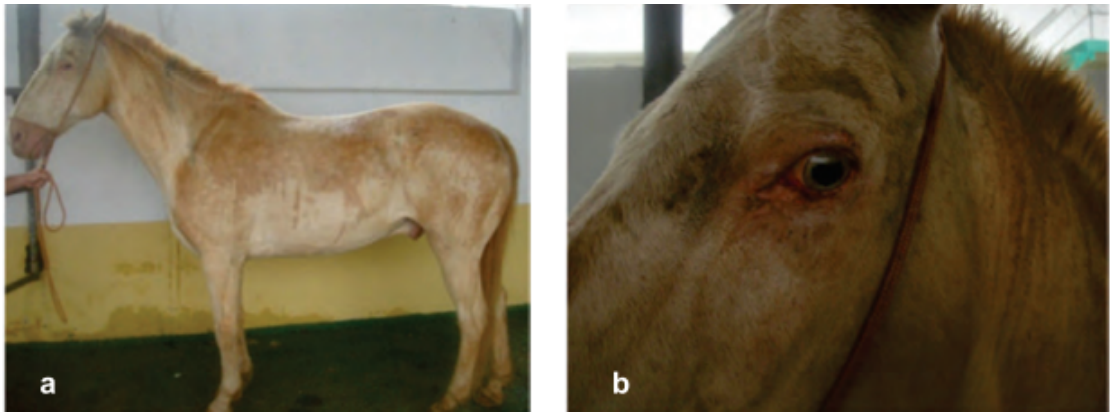


Figura 4 – Albinóide: a e b.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

- **Preta**: O corpo todo do animal é recoberto por pelos de coloração preta, de tonalidades que vão de preto brilhante ao opaco, que ao reflexo da luz exibem uma tonalidade azulada, como ilustrado na Figura 14 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Preta maltina**: É a pelagem preta com reflexos avermelhados principalmente nas regiões do flanco e axilas, mas a cabeça tem sempre predomínio de pelos pretos

(GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Preta azeviche**: É a pelagem com pelos pretos e de tonalidade bem forte, que apresenta reflexos azulados, sendo também uma pelagem rara (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Alazã**: Pelos da cabeça, pescoço, tronco, membros, crina e cauda de coloração vermelha (canela), que pode variar do escuro ao amarelado. Crina e/ou cauda podem ser de tonalidade mais clara, como ilustrado na Figura 5 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

Variedades comuns:

- **Alazã tostada:** Pelos do corpo, crina e cauda de tonalidade vermelha escura, lembrando a "cor do café torrado", como ilustrado na Figura 5 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Alazã cereja:** Pelos de tonalidade vermelha, lembrando a cor da cereja, como ilustrado na Figura 5 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Alazã sobre baia (acima de baia):** Cabeça, pescoço e tronco amarelos, com crina, cauda e extremidades avermelhadas, como ilustrado na Figura 5 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Alazã Amarilha:** Pelos de tonalidade amarela, que podem variar do claro ao escuro, com crina e cauda branca ou creme, como ilustrado na Figura 5 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 5 – Variações da pelagem alazã: a) tostada; b) cereja; c) sobre baia; d) amarilha.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

2.1.2.2 Simples e uniformes com crina, cauda e extremidades pretas:

São caracterizados por apresentarem coloração uniforme na cabeça, pescoço e tronco, porém, com crina, cauda e extremidades pretas.

- **Castanho claro:** O vermelho da pelagem é de tonalidade clara (amarelada) com crina, cauda e membros pretos, como ilustrado na Figura 6 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Castanho escuro:** O vermelho da pelagem é de tonalidade escura com crina, cauda e membros pretos, como ilustrado na Figura 6 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 6 – a) Castanha clara; b) Castanha escura.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

- **Zaina:** Pelagem em que pelos pretos e castanhos se entremeiam, dando uma tonalidade geral escura, com regiões como ganachas, axilas, flancos e virilhas com tonalidade amarelada, bem mais clara que as demais partes do corpo, como ilustrado na Figura 7 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 7 – Pelagem zaina.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

- **Baio:** Caracteriza-se pela presença de pelos amarelos que variam do claro ao escuro na cabeça, pescoço e tronco, com crina, cauda e extremidades pretas. Geralmente apresenta listra de burro, faixa crucial e zebruras, como ilustrado na Figura 8 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Pelo de rato:** Pelos cinza na cabeça, pescoço e tronco, com crina, cauda e extremidades pretas, com faixas cruciais e zebruras. Essa pelagem não é encontrada em



Figura 8 – Pelagem baia.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

equinos, sendo exclusiva de asininos e muares, como ilustrado na Figura 9 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 9 – Pelo de rato.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

2.1.2.3 Pelagens compostas

- **Tordilha:** Interpolação de pelos brancos em todo o corpo do animal, como ilustrado na Figura 10 e na Figura 11 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Tordilha negro:** Apresenta pelagem preta com poucos pelos brancos, como ilustrado na Figura 10 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Tordilha apatacada:** Interpolação de pelos pretos e brancos esboçando a forma de patacas (moedas antigas) na superfície da pelagem, como ilustrado na Figura 10 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Tordilha claro:** Predomínio de pelos brancos na pelagem tordilha, como ilustrado na Figura 10 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Tordilha ruça:** Quando não mais se observar no tordilho os pelos pretos da pelagem de origem. O animal terá o corpo recoberto por pelos brancos, como ilustrado na Figura 10 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Tordilha cardã:** Pelagem tordilha que apresenta reflexos avermelhados ou amarelados. Comum naqueles animais que nasceram castanhos, alazões ou baios, como ilustrado na Figura 11 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 10 – Pelagens tordilha: a) negra; b) apatacada; c) clara; d) ruça.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)



Figura 11 – Pelagem tordilha: e) cardã; f) pedrez.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

- **Rosilha:** Pelagem que mescla, de maneira uniforme, pelos brancos e vermelhos, dando ao conjunto uma tonalidade rósea; crina, cauda e extremidades dos membros da cor do corpo ou, raramente, mais escuras, como ilustrado na Figura 12 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Lobuna:** Caracteriza-se pela interpolação de pelos amarelos e pretos, também caracterizada pelo predomínio de pelos pretos na cabeça, como ilustrado na Figura 12 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

- **Ruão:** Caracteriza-se pela interpolação de pelos vermelhos, pretos e brancos. Nas extremidades, geralmente, os pelos são pretos, e a cauda e a crina são mais claras, como ilustrado na Figura 12 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).

2.1.2.4 Pelagens Conjugadas

- **Pampa de preto**: Predomina o branco sobre o preto, como ilustrado na Figura 13 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 12 – Variações: ruão, lobuna e rosilha, respectivamente.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

- **Preto de pampa**: Predomina o preto sobre o branco, como ilustrado na Figura 13 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Baio de pampa**: Predomina o baio sobre o pampa, como ilustrado na Figura 13 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).
- **Pampa de alazão**: Predomina o pampa sobre o alazão, como ilustrado na Figura 13 (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012).



Figura 13 – a) Baia de pampa; b) Preto de pampa; c) Pampa de preto; d) Pampa de alazão.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)



Figura 14 – Pelagem preta.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

2.1.3 Tipos de particularidade

As particularidades dos equídeos podem aparecer em quatro localidades: cabeça, tronco, pescoço e membros. Elas podem ser divididas em particularidades especiais, delimitadas por pelos brancos, pretos e/ou escuros contrastantes com a pelagem dominante, ocorrendo nas áreas da cabeça, pescoço, tronco e membros (BATISTA; CARRIJO, 2016); e particularidades gerais, que não são delimitadas por uma coloração contrastante da pelagem, mas por um padrão diferente de crescimento dos pelos, formando uma marca no animal GOIÁS. Agência Goiana de Defesa Agropecuária (2012).

Segundo Batista e Carrijo (2016), quando os sinais brancos estiverem localizados na cabeça, sobre a pele despigmentada, dependendo da forma, região e tamanho, recebem nomes como os apresentados na Tabela 2. A Figura 15 apresenta algumas dessas marcas.

Local Marca	**Marca**	**Descrição**
Cabeça	Redemoinhos	Forma Arredondada.
Cabeça	Espada Romana	Quando tiver uma espiga localizada na tábua do pescoço.
Cabeça	Estrela	Formada por uma mancha branca na fronte, podendo ter várias formas: em coração, em losango, em meia lua, em U e "escorrida".
Cabeça	Luzeiro	Formado por uma malha na fronte, podendo ser também "escorrido".
Cabeça	Cordão	Determinado por uma fina mancha branca (mais largo que o filete), que se estende da fronte ao chanfro, e até as narinas às vezes, podendo ser interrompido ou desviado.
Cabeça	Filete	Determinado por um estreito fio de pelos brancos que escorre pela fronte ou chanfro.
Cabeça	Beta	Pinta branca que corre entre as narinas.
Cabeça	Ladre	Mancha branca entre as narinas que se apresenta ligada ao cordão ou filete.

Local Marca	**Marca**	**Descrição**
Cabeça	Bebe com Branco	Quando um dos lábios ou ambos são brancos.
Cabeça	Malacara	Malha branca despigmentada que recobre toda a fronte, todo o chanfro e atinge a região do focinho e bochecha.
Cabeça	Frente Aberta	Malha branca despigmentada que recobre toda a fronte e chanfro.
Cabeça	Pelos Brancos Superficiais (Vestígios)	Pelos brancos esparsos localizados na região da fronte.

Tabela 2 – Marcas da cabeça.

Fonte: Batista e Carrijo (2016)

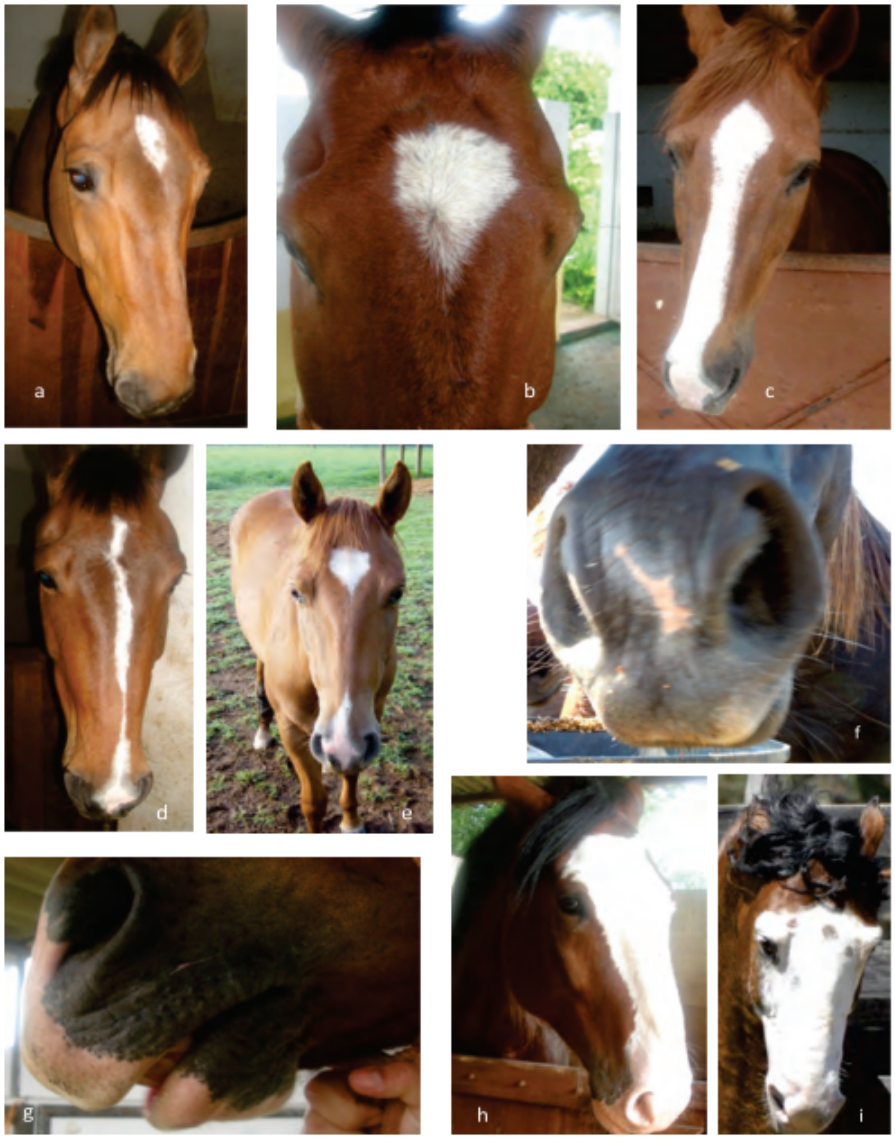


Figura 16 – Cavalo crinalvo.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

Na região do pescoço, existe uma particularidade chamada crinalvo, que ocorre quando as crinas são brancas e o restante da pelagem é colorido (BATISTA; CARRIJO, 2016), como apresentado na Figura 16. Também podem ocorrer redemoinhos, que consistem em um padrão diferente de crescimento dos pelos; quando possuem forma mais alongada,

Figura 15 – Particularidades especiais dos equinos.



A) Estrela; B) Luzeiro; C) Luzeiro com cordão desviado para a narina direita e ladre; D) Estrela com filete e ladre; E) Luzeiro com filete a partir do final do chanfro e ladre; F) Beta; G) Bebe em branco; H) Frente aberta; I) Malacara.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

são chamados de espiga. Se uma espiga estiver localizada na tábua do pescoço, ela é chamada de espada romana.

No tronco, de acordo com o manual da AGRODEFESA para elaboração de resenha de equídeos com a finalidade de solicitação de diagnóstico para anemia infecciosa equina (AIE) e Mormo (GOIÁS. Agência Goiana de Defesa Agropecuária, 2012), podem ser encontradas:

- **Listra de burro**: Listra estreita, mais escura que a pelagem, estendendo-se na região dorsal, da cernelha à base da cauda.

- **Redemoinhos**: Forma arredondada.
- **Seta (espiga)**: Forma mais alongada.
- **Faixa crucial**: Faixa escura que corta transversalmente a cernelha, geralmente de pelagem vermelha, alcançando as espáduas.
- **Bragado**: Quando o animal apresenta malhas brancas no terço posterior do ventre e nas partes internas das coxas.

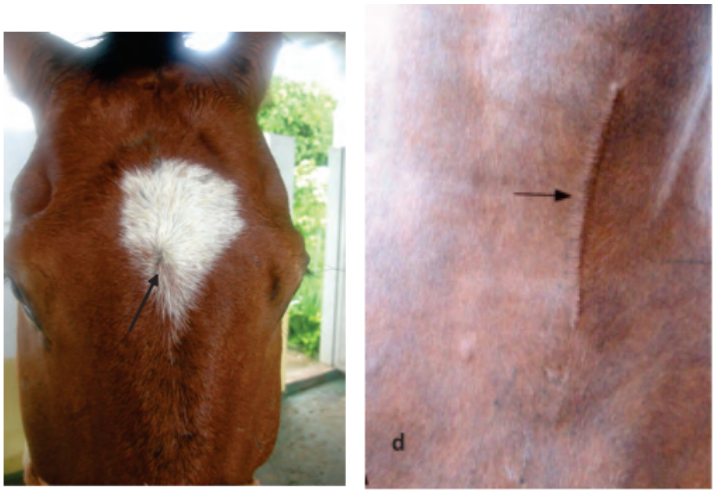


Figura 17 – Redemoinho na linha dos olhos e espiga, respectivamente.

Fonte: GOIÁS. Agência Goiana de Defesa Agropecuária (2012)

Nos membros, marcas brancas bem delineadas em contraste com a cor da pelagem dominante caracterizam os diferentes calçamentos, sendo eles:

- **Calçado sobre a coroa**: Quando o branco está situado apenas na circunferência da coroa do casco.
- **Baixo calçado**: Quando o branco vai até o boleto (quartela).
- **Médio Calçado**: Quando o branco abrange qualquer parte da canela.
- **Alto Calçado**: Quando o branco alcança os joelhos e os jarretes.
- **Zebruras**: Quando o animal apresenta malhas brancas no terço posterior do ventre e nas partes internas das coxas.

A Figura 18 apresenta os diferentes tipos de calçados que um equídeo pode vir a possuir.

Foi apresentada acima uma grande gama de marcas que podem ocorrer no animal. Durante o processo de resenha, cada marca presente deve ser anotada no local exato da resenha gráfica (BATISTA; CARRIJO, 2016). A Figura 1 é um exemplo do espaço onde devem ser feitas as marcações.



Figura 18 – Particularidades nos membros.

Fonte: Batista e Carrijo (2016)

> Por exemplo, os remoinhos são marcados puxando – se uma linha para fora do desenho, simbolizados por um "X" ou um "R", quando simples e por "X-" (xis e traço) quando são espigas. Batista e Carrijo (2016)

Além dos tipos de marcas e particularidades descritos aqui, existe também um grande número de variações de cores de pelagens nesses animais. Com base no banco de dados relacional proposto por Batista e Carrijo (2016), este trabalho visa criar um banco de dados não relacional para classificar tais marcas e nomenclaturas de cores, de modo que ele possa ser usado na implementação da aplicação móvel desenvolvida neste trabalho, com base nos requisitos coletados por Batista e Carrijo (2016).

2.1.4 Engenharia de Software

No desenvolvimento da aplicação, utilizou-se o método *Kanban*, conforme descrito por Kanban (2022), por se tratar de uma abordagem simples e que não pressupõe papéis rigidamente definidos, como ocorre no *Scrum* (Jeff Stuherland, 2014), tornando-se mais compatível com um projeto desenvolvido individualmente.

O *Kanban* sugere que o desenvolvimento de um projeto seja dividido em tarefas dispostas em cartões em um quadro organizado por colunas, em que cada coluna representa

um estágio da resolução da tarefa. No projeto implementado neste trabalho, os requisitos coletados por Batista e Carrijo (2016) transformaram-se em tarefas para o quadro. Na Figura 19, apresenta-se um exemplo de quadro *Kanban* simplificado, utilizado com o objetivo de ilustrar a metodologia descrita. O quadro é composto por três colunas principais: *TO-DO*, *Work in Progress* (WIP) e *Done*. A coluna *TO-DO* concentra as tarefas que ainda não foram iniciadas; a coluna *Work in Progress* reúne as atividades que se encontram em fase de desenvolvimento ou execução; por fim, a coluna *Done* contém as tarefas já concluídas.

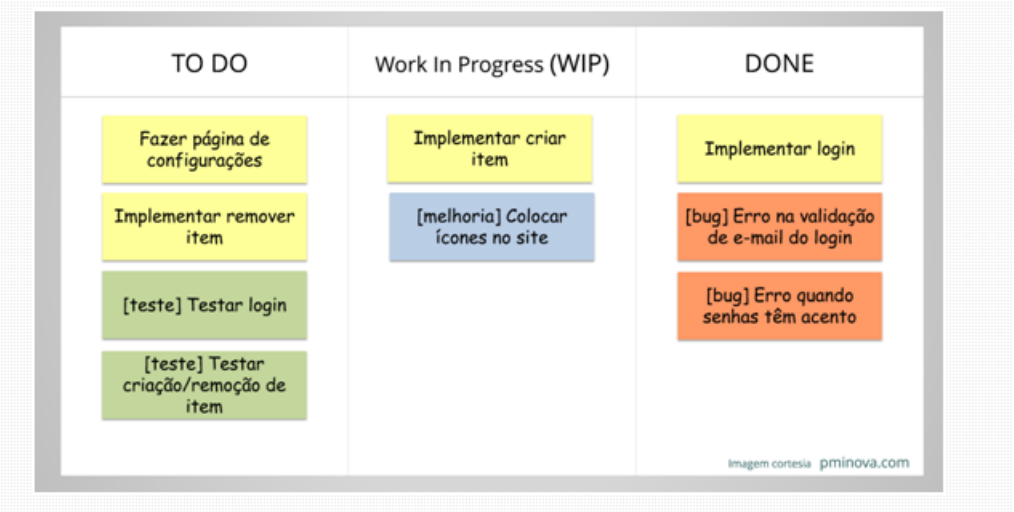


Figura 19 – Exemplo *Kanban* com tarefas.

Fonte: <https://bityli.com/UVtLta> **Acesso em**: 06/06/2020

2.1.4.1 Desenvolvimento de aplicações móveis

O desenvolvimento de aplicações móveis pode ser entendido como o processo de engenharia de *software* voltado à concepção, construção, teste e manutenção de sistemas que executam em dispositivos portáteis, como smartphones e tablets, normalmente conectados a redes sem fio e dotados de sensores diversos. De acordo com Sommerville (2016), tais sistemas devem ser tratados como produtos de *software* completos, sujeitos às mesmas atividades sistemáticas de especificação, projeto, implementação e validação previstas para outras categorias de aplicação, ainda que apresentem restrições particulares de ambiente e de plataforma.

Na literatura de engenharia de *software*, autores como Pressman e Maxim (2014) destacam que aplicações móveis se diferenciam de aplicações tradicionais de *desktop* por uma combinação de fatores: diversidade de plataformas e lojas de aplicativos, ciclos de desenvolvimento mais curtos, interfaces de usuário limitadas pelo tamanho da tela e pela interação por toque, além de forte dependência de conectividade de rede e de integração com serviços externos e sensores do dispositivo (câmera, *Global Positioning System* (GPS),

acelerômetro, entre outros). Essas características impõem desafios adicionais de projeto relacionados a desempenho, consumo de bateria, segurança e privacidade dos dados.

Do ponto de vista de qualidade de produto, Maia (2019) argumenta que aplicações móveis constituem um contexto específico dentro dos modelos de qualidade de *software*, pois são fortemente afetadas por variações de contexto de uso, pelo emprego de sensores do dispositivo e por requisitos de usabilidade elevados, o que demanda a particularização de modelos de qualidade gerais (como a norma ISO/IEC 25010) para esse tipo de *software*. No contexto desta aplicação, métricas como tempo médio para conclusão da resenha, taxa de sucesso na execução das tarefas, quantidade de erros de seleção das regiões zootécnicas, número de tentativas até o clique correto e grau de satisfação do usuário poderiam ser empregadas para avaliar a usabilidade do sistema em dispositivos móveis. Tais métricas seriam particularmente adequadas, pois o *software* depende de interação por toque, navegação entre telas e marcação precisa em mapas de imagem. Entretanto, elas não foram aplicadas neste trabalho porque o foco da pesquisa esteve na implementação e validação técnica da solução proposta, e não na condução de um estudo empírico formal de usabilidade com médicos-veterinários em contexto real de campo. Assim, a avaliação quantitativa da usabilidade fica caracterizada como oportunidade para trabalhos futuros.

Ainda na perspectiva tecnológica, o desenvolvimento móvel pode ser realizado por meio de abordagens nativas, híbridas ou multiplataforma. Obras introdutórias, como Alves (2017), descrevem que, enquanto o desenvolvimento nativo utiliza diretamente as linguagens e os *Software Development Kits* (SDKs) de cada sistema operacional (por exemplo, Java/Kotlin para Android e Swift/Objective-C para iOS), abordagens híbridas e multiplataforma permitem construir a interface em tecnologias *web* (HTML, CSS e *JavaScript*) ou em *frameworks* especializados, gerando executáveis para diferentes plataformas a partir de uma base de código única. No contexto deste trabalho, a aplicação proposta enquadra-se como uma aplicação móvel híbrida, voltada ao sistema Android, que combina o uso de *web technologies* (via React) com o empacotamento fornecido pelo Apache Cordova, de modo a atender aos requisitos de informatização da resenha de equídeos em um cenário de uso predominantemente no campo.

2.1.4.2 Banco de dados Firestore

Uma das etapas do desenvolvimento da aplicação foi a implementação do banco de dados conceituado por Batista e Carrijo (2016). O trabalho citado modela o banco de dados da aplicação tendo em vista o uso do PostgreSQL; porém, neste trabalho, mostrou-se válida a adoção de um banco de dados não relacional Firestore (2022).

> O Cloud Firestore é um banco de dados NoSQL orientado a documentos. Ao contrário de um banco de dados SQL, não há tabelas nem linhas. Em vez disso, os dados são armazenados em documentos, que são organizados em coleções. Firestore (2022)

Com o uso do *Firestore* em vez de um banco de dados *Structured Query Language* (SQL), é possível tirar proveito do fato de ele funcionar na nuvem, eliminando a necessidade de um servidor de banco de dados dedicado. O funcionamento em nuvem elimina a necessidade de manter uma infraestrutura própria para esse fim, mas não elimina custos. Esses custos passam a estar vinculados ao consumo do serviço gerenciado. No *Cloud Firestore*, existe uma franquia gratuita limitada, aplicável a um banco por projeto, que inclui 1 GiB de armazenamento, 50.000 leituras diárias, 20.000 escritas diárias, 20.000 exclusões diárias e 10 GiB mensais de transferência de saída; ultrapassados esses limites, torna-se necessário habilitar cobrança no projeto. Como os valores excedentes variam conforme a localidade da infraestrutura, pode-se afirmar, de modo geral, que a adoção do *Firestore* reduz o ônus de administração de infraestrutura própria, mas transfere o custo para um modelo recorrente de consumo de serviço. Esse ponto, aliado ao fato de um banco de dados não relacional como o *Firestore* possuir uma estrutura e um modelo de dados mais simples, reduz o tempo necessário para a implementação do projeto proposto. Devido à natureza não relacional do *Firestore*, que adota um modelo de dados orientado a documentos, a estrutura do banco apresenta menor rigidez em relação a esquemas fixos, quando comparada a bancos de dados relacionais tradicionais.

Em sistemas relacionais, a modelagem passa obrigatoriamente pelas etapas conceitual, lógica e física, com definição prévia de tabelas, colunas, tipos de dados e restrições de integridade, as quais devem ser rigorosamente seguidas durante a inserção e atualização dos dados. Em contrapartida, no *Firestore* os dados são organizados em coleções e documentos, não sendo exigida a definição prévia de um esquema fixo. Cada documento pode possuir campos diferentes, permitindo maior flexibilidade na evolução da estrutura de dados ao longo do desenvolvimento do sistema. Do ponto de vista da arquitetura de sistemas, ao empregar bancos de dados não relacionais como o *Firestore*, a responsabilidade pela garantia do formato, da tipagem e da integridade dos dados passa a ser do cliente da base de dados, neste caso o *software* desenvolvido neste trabalho.

A Figura 20 apresenta as coleções de dados de proprietários e animais cadastrados no sistema, sendo possível observar que cada proprietário é representado como um documento dentro da coleção proprietários, contendo seus respectivos atributos e referências aos animais associados.

2.1.4.3 React

O React é uma biblioteca *JavaScript* utilizada para a construção de interfaces de usuário para a *web*, amplamente empregada no desenvolvimento de aplicações baseadas em componentes reutilizáveis e na atualização eficiente da interface. Além de seu uso em aplicações *web*, o ecossistema do React também abrange soluções voltadas ao desenvolvimento móvel, como o React Native, que permite a construção de aplicações para Android

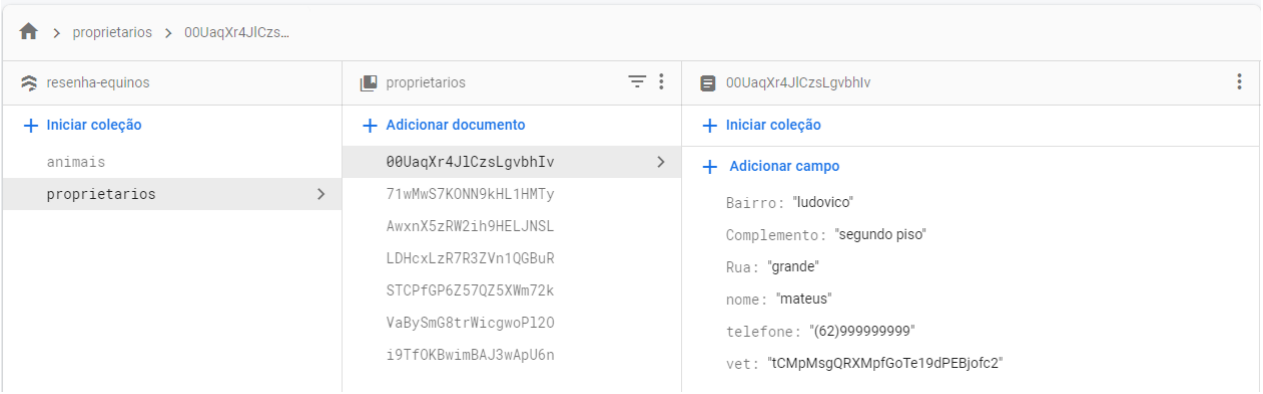


Figura 20 – Coleções de animais e proprietários no *Firestore*.

Fonte: Próprio autor

e iOS a partir de uma única base de código. Dessa forma, o React se destaca por favorecer a reutilização de código, a organização modular da interface e a manutenção de aplicações com maior complexidade.

2.1.4.4 Apache Cordova

Como explicado na subseção anterior, o React produz interfaces voltadas para a *web*. Com o objetivo de fazer com que essas interfaces funcionem em um aplicativo móvel, foi utilizado no projeto o Apache Cordova, ferramenta multiplataforma que permite a construção de aplicações híbridas (DESHPANDE, 2021), ou seja, aplicações construídas com ferramentas para *web*, mas que se comportam como aplicativos nativos.

O Cordova permitiu que a aplicação construída em React fosse executada dentro de um componente *WebView* do sistema operacional (isto é, um componente nativo responsável por renderizar conteúdo *web* dentro do aplicativo), além de possibilitar o acesso a recursos do dispositivo, como câmera, sensores e armazenamento.

A Figura 21 apresenta um diagrama da arquitetura de uma aplicação desenvolvida com Apache Cordova. Nessa arquitetura, a camada *Web* (bloco chamado *Web App* na figura em questão), composta por HTML, CSS e *JavaScript*, além de recursos como imagens e mapas, é executada dentro de uma *WebView* do sistema operacional. Já o arquivo `config.xml` é utilizado pelo Cordova para definir configurações globais da aplicação, como identificador, versão, página inicial e preferências de plataforma. A *WebView* atua como um navegador *web* embutido, responsável por renderizar a interface e executar o código *JavaScript* do aplicativo.

Para acessar funcionalidades do dispositivo que não existem no ambiente puramente *web*, o Cordova fornece uma ponte (descrita como *Cordova Native APIs* na Figura 21) entre o *JavaScript* e o código nativo. Essa ponte é utilizada por *plugins* Cordova (por exemplo, geolocalização e câmera), que expõem APIs em *JavaScript* e, internamente,

chamam as APIs do sistema operacional. Por fim, essas chamadas chegam aos serviços do sistema operacional móvel, como sensores, entrada (*input*), serviços e gráficos, permitindo que o *app web* se comporte como um aplicativo nativo ao acessar recursos de hardware e do sistema.

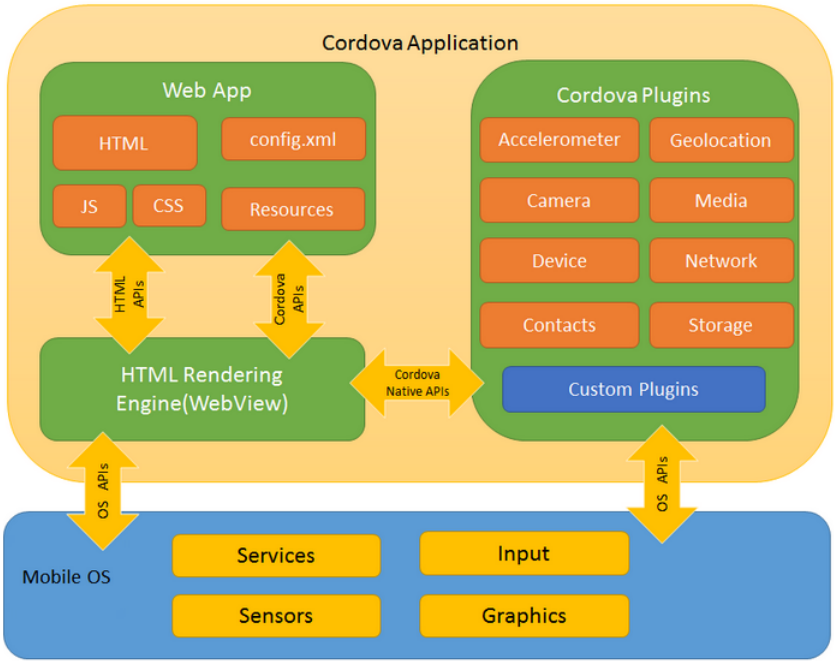


Figura 21 – Visão geral da arquitetura do Cordova.

Fonte: Deshpande (2021)

2.1.4.5 Navegação com React Router (HashRouter) em ambiente híbrido

Em aplicações híbridas (isto é, aplicações móveis desenvolvidas com tecnologias *web*, como HTML, CSS e *JavaScript*, mas empacotadas para execução como aplicativos instaláveis no sistema operacional) executadas em *WebView* (como as empacotadas com Apache Cordova), a aplicação React funciona como uma *Single Page Application* (SPA): existe um único ponto de entrada (o arquivo `index.html`, presente no diretório `www/` do pacote do *app*) e as diferentes "telas" são componentes React renderizados conforme a rota.

Em aplicações *web* hospedadas em um servidor, é comum utilizar o BrowserRouter[1], que se baseia na *History API*[2], com rotas como `/cabeca` ou `/membros-post`. Nessa abordagem, quando o usuário recarrega a página ou acessa uma rota diretamente (por

[1] BrowserRouter é um componente da biblioteca React Router, responsável por realizar a navegação em aplicações React a partir da URL exibida no navegador, utilizando caminhos convencionais, como `/cabeca` ou `/membros-post`, sem a necessidade do caractere `#`.

[2] *History API* é uma interface disponibilizada pelos navegadores que permite manipular o histórico de navegação e alterar a URL exibida sem recarregar completamente a página, viabilizando o funcionamento de aplicações de página única (*Single Page Applications* – SPA).

exemplo, digitando a URL, abrindo um favorito ou um link), o navegador/*WebView* tenta carregar aquele caminho como se fosse um endereço de verdade, isto é, como se existisse uma página ou arquivo correspondente a ele. Em um site com servidor configurado para SPA, esse problema é resolvido por uma regra de *fallback*: independentemente do caminho solicitado, o servidor responde com o mesmo `index.html`, e então o React Router interpreta a rota e renderiza o componente correto.

No Cordova, entretanto, não há um servidor *web* configurado para realizar esse *fallback*. Como a aplicação é carregada localmente no *WebView*, o recarregamento de uma rota interna pode levar o ambiente a tentar resolvê-la como se fosse um caminho físico dentro do pacote da aplicação. A Figura 22 ilustra esse comportamento. À esquerda, observa-se o fluxo problemático, em que a navegação para uma rota interna faz o *WebView* tentar localizar um recurso correspondente no pacote local, resultando em falha de carregamento. À direita, apresenta-se a alternativa adotada neste trabalho com o uso do HashRouter[3], na qual o carregamento continua apontando para o mesmo arquivo principal da aplicação, enquanto o fragmento da rota é interpretado apenas pelo *JavaScript* do React.

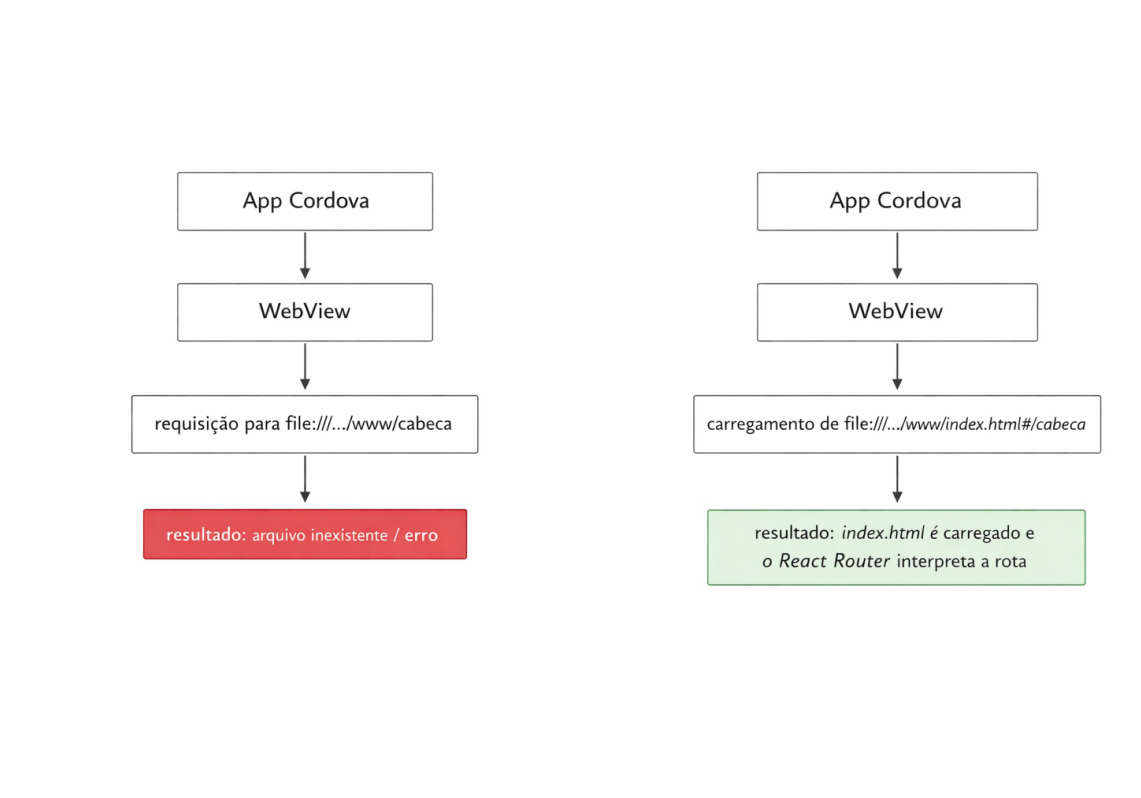


Figura 22 – Comparação entre o comportamento de navegação com rota interna em aplicação React empacotada com Cordova.

Fonte: Próprio autor

[3] HashRouter é um componente da biblioteca React Router que realiza a navegação a partir do fragmento da URL, isto é, da parte localizada após o caractere `#`. Dessa forma, a rota é interpretada pelo *JavaScript* da aplicação sem ser tratada pelo navegador ou *WebView* como um caminho físico de arquivo.

Com essa estratégia, evita-se a dependência de reescritas de rota no contêiner híbrido, tornando a navegação mais robusta em cenários de recarregamento da tela e de abertura direta de rotas internas.

No contexto deste trabalho, essa decisão garantiu:

- **Compatibilidade** com o empacotamento Android via Cordova;
- ***Deep-link*** funcional dentro do contêiner híbrido;
- **Propagação** do identificador do animal (`animalId`) tanto por `state` na navegação quanto por `?animalId=` na *querystring*[4], preservando telas legadas.

2.1.4.6 Mapa de imagens HTML

O elemento HTML *map* pode ser usado para criar áreas clicáveis dentro de imagens, utilizando o elemento *area*. Ou seja, dentro de uma mesma imagem, é possível ter diferentes efeitos para cliques realizados em diferentes regiões da imagem. Por exemplo, ao clicar na cabeça do equídeo, o usuário será redirecionado para a página `cabeça.html` (BATISTA; CARRIJO, 2016).

Com o uso de *map*, é possível capturar a região zootécnica exata em que o usuário realizou um clique, permitindo, na aplicação, que ele toque nas regiões do animal em que deseja adicionar uma marcação para a resenha.

Os mapas HTML usados no desenvolvimento deste trabalho estão disponíveis no Apêndice 5 de Batista e Carrijo (2016).

2.1.4.6.1 Responsividade e Acessibilidade.

Para garantir a correção do clique em diferentes resoluções e densidades de tela, as coordenadas do *image map* são **recalculadas proporcionalmente** com base nas dimensões originais da imagem (largura e altura) sempre que a imagem é carregada ou redimensionada.

[4] *Querystring* é a parte da URL composta por pares `chave=valor` após o caractere `?`, opcionalmente separados por `&`. Por exemplo: `https://exemplo.com/#/rota?animalId=123`.

3 Metodologia

Este trabalho de conclusão de curso caracteriza-se como um trabalho de implementação de *software*, de natureza aplicada, voltado ao desenvolvimento de um sistema informatizado para a resenha de equinos e muares. A implementação aqui descrita toma como base o estudo exploratório de engenharia de *software* conduzido por Batista e Carrijo (2016), que realizou o levantamento de requisitos, a modelagem do domínio e a definição preliminar da solução. A partir desses artefatos e da compreensão do ambiente de aplicação, foram selecionadas as tecnologias mais adequadas e iniciou-se o desenvolvimento da aplicação.

O projeto reutiliza e amplia a base de pelagens (nomes das cores) e marcas naturais coletadas por Batista e Carrijo (2016), integrando-as a um sistema de gerenciamento de banco de dados na nuvem. Para isso, adotou-se o *Firestore Database*, do *Google Firebase*, como Sistema Gerenciador de Banco de Dados (SGBD), de forma que, futuramente, tais dados possam ser consultados em uma arquitetura cliente-servidor. Nesta aplicação, todas as necessidades de *backend* são atendidas por serviços do *Firebase*, o que acelera o desenvolvimento — pela dispensa da implantação manual de servidores de aplicação e de banco de dados — e oferece uma solução mais simples para escalabilidade da solução proposta.

3.1 Etapas do desenvolvimento

O desenvolvimento do sistema foi conduzido de forma incremental e iterativa, inspirado em práticas de desenvolvimento ágil, com adoção do método *Kanban* (*embora Kanban possa ser combinado a Scrum em alguns contextos, neste trabalho ele foi utilizado como principal estratégia de organização do fluxo de atividades*). Considerando que uma parte significativa da análise de requisitos já havia sido realizada por Batista e Carrijo (2016) — incluindo o levantamento do domínio, identificação das principais funcionalidades e modelagem inicial da solução — este trabalho concentrou-se em refinar esses artefatos e transformá-los em um produto funcional.

De maneira geral, as etapas do desenvolvimento podem ser resumidas nos seguintes passos:

1. **Revisão e complementação de requisitos**: inicialmente, foram analisados os artefatos produzidos por Batista e Carrijo (2016), tais como descrição do problema, funcionalidades desejadas, modelos de dados (lógico e físico), diagrama de contexto e requisitos de interface e de *layout* sugeridos. A partir dessa base, foram identificados

os requisitos diretamente relacionados à implementação da aplicação móvel híbrida e realizados pequenos ajustes e complementações. Um desses ajustes decorre do fato de que o trabalho original modelou o banco de dados com foco em uma abordagem relacional, enquanto, no presente trabalho, foi utilizado o *Firestore*.

2. **Planejamento e organização do *backlog***: com os requisitos revisados, foi elaborado um *backlog* de produto contendo as funcionalidades a serem implementadas (por exemplo: cadastro de proprietário, cadastro de equídeos, resenha com marcações sobre a imagem, persistência em nuvem etc.). Esse *backlog* foi priorizado de forma a viabilizar uma versão inicial funcional do sistema (*versão mínima viável*).

3. **Implementação incremental**: a partir do *backlog* priorizado, as funcionalidades foram desenvolvidas em pequenos incrementos. Em cada ciclo de desenvolvimento, eram selecionados itens do *backlog*, implementados com o uso de React e Cordova para a camada de interface e de integração com o dispositivo móvel, e de serviços do *Firebase* (como *Firestore Database* e autenticação) para o *backend*. Ao final de cada incremento, o sistema era executado em dispositivos Android para validação do comportamento.

4. **Testes e ajustes**: após cada incremento, foram adicionados testes unitários, com foco em verificar o correto cadastramento dos dados, a navegação entre as telas e a interação com as imagens de resenha. Erros identificados nessa etapa eram registrados, priorizados e corrigidos em ciclos subsequentes, seguindo a lógica de melhoria contínua proposta pelas metodologias ágeis.

Essa abordagem em etapas permitiu que o desenvolvimento acompanhasse a lógica de inspeção e adaptação típica do desenvolvimento ágil, ainda que em um contexto de equipe reduzida (autor e orientador), garantindo flexibilidade para ajustes ao longo do processo sem perder de vista o escopo definido a partir do trabalho exploratório anterior.

3.2 Ferramentas e métodos utilizados

O desenvolvimento do sistema foi apoiado por um conjunto de ferramentas de *software* e por práticas inspiradas em metodologias ágeis, de forma a organizar o trabalho, facilitar a iteração sobre o código e possibilitar ajustes rápidos ao longo do processo.

No que diz respeito à gestão do desenvolvimento, adotou-se uma abordagem baseada em práticas ágeis, com organização visual inspirada em *Kanban*, adaptada ao contexto de um projeto individual. Foram definidos pequenos ciclos de trabalho, com objetivos claros, priorizando-se as funcionalidades essenciais para a versão inicial do sistema. A cada ciclo, eram planejadas as tarefas, implementadas as funcionalidades selecionadas e, ao final, realizados testes e ajustes.

Para organização das atividades, utilizou-se o *Trello*, um serviço *web* gratuito que oferece quadros no formato *Kanban* para o gerenciamento visual de tarefas. Nesse quadro, as atividades eram distribuídas em listas como "A Fazer", "Em Andamento" e "Concluído", permitindo acompanhar o progresso do desenvolvimento, priorizar demandas e registrar pendências de correção.

A implementação do sistema foi realizada utilizando o editor de código *Visual Studio Code*, que oferece recursos como realce de sintaxe, auto-completação, integração com controle de versão e extensões específicas para desenvolvimento em *JavaScript* e *React*. O gerenciamento de dependências e *scripts* de construção foi feito com o ecossistema *Node.js*, por meio do *Node Package Manager* (NPM), utilizando a versão **v24** do *Node.js*, em conjunto com as ferramentas da linha de comando do *Apache Cordova*, na versão **13.0.0**[1].

Para controle de versão do código-fonte, utilizou-se o sistema *Git*, com repositório hospedado em uma plataforma de *hosting* de código (*GitHub*). O uso do *Git* permite registrar o histórico de alterações, facilitar a recuperação de versões anteriores e garantir maior segurança no armazenamento do projeto.

Para os testes, foram combinadas abordagens automatizadas e manuais. Como suporte aos testes unitários em *JavaScript*, adotou-se a biblioteca *Jest*, amplamente utilizada na comunidade para validação de funções e componentes em projetos baseados em *React*. Além disso, foram executados testes funcionais e exploratórios diretamente em dispositivos Android e em emuladores, com foco em verificar o comportamento da interface, a integração com o *Firestore Database* e a consistência dos dados de resenha armazenados na nuvem.

A combinação dessas ferramentas e métodos contribuiu para um processo de desenvolvimento mais estruturado, iterativo e alinhado às boas práticas de engenharia de *software*, uma vez que testes de *software* (unitários, de integração e funcionais) são considerados parte integrante da verificação e validação de um sistema, com o objetivo de demonstrar que ele atende aos requisitos e de descobrir defeitos antes da implantação, conforme discutido por Sommerville no capítulo dedicado a testes de *software* (SOMMERVILLE, 2016, Cap-8). Nesse contexto, aproveitou-se o trabalho exploratório já realizado por Batista e Carrijo (2016), avançando-se na concretização de uma solução implementada.

[1] Até março de 2026, a linha **v24** é a LTS ativa mais recente do *Node.js*. Já o *Apache Cordova CLI* não adota, em sua documentação oficial, uma classificação explícita de versões *LTS*; por isso, optou-se por citar sua versão estável mais recente, **13.0.0**.

4 Desenvolvimento do Software

Este capítulo descreve o processo de construção do sistema informatizado para resenha de equinos e muares, desde o reaproveitamento da análise de requisitos e da modelagem proposta por Batista e Carrijo (2016) até a implementação da solução com *web* híbrida (*React* + *Apache Cordova*) e banco de dados *NoSQL* (*Cloud Firestore*). São apresentados, de forma sintética, o levantamento de requisitos, a modelagem conceitual, a arquitetura do sistema, as principais funcionalidades implementadas, os testes realizados e os desafios encontrados ao longo do desenvolvimento.

4.1 Levantamento de requisitos

O levantamento de requisitos deste trabalho se apoia diretamente no trabalho-base de Batista e Carrijo (2016), que já havia conduzido uma etapa detalhada de análise do domínio, identificação de atores e definição das funcionalidades essenciais para um sistema de apoio à resenha de equinos e muares. Nesse sentido, a modelagem definida pelo trabalho inicial foi tomada como ponto de partida.

Código	**Requisito funcional**
RF-01	Autenticar usuário (médico veterinário) por número do veterinário CRMV (Conselho Regional de Medicina Veterinária) e senha, liberando o acesso às funcionalidades do controle de resenhas.
RF-02	Encerrar aplicação / sair: permitir finalizar o aplicativo via opção *Sair*, retornando à tela de *login* (sem permanecer em segundo plano).
RF-03	Exibir menu e navegar entre módulos: disponibilizar opções para Menu principal (Controle de Resenhas), Cadastro Proprietário, Busca Proprietário, Cadastro Equino, Configurações, Ajuda e Sair.
RF-04	Ajuda / resolução de problemas: disponibilizar tela de Ajuda com lista de problemas comuns e orientações de solução.
RF-05	Listar e buscar resenhas: exibir o Controle de Resenha e permitir busca/filtragem por critérios (ex.: proprietário, nome do animal, cor e subcor), incluindo opção de seleção “todos” quando aplicável.
RF-06	Visualizar detalhes a partir do controle: permitir abrir a visualização detalhada de dados do proprietário e do equídeo associados à resenha selecionada.
RF-07	Editar resenha existente: permitir alterar dados de uma resenha previamente cadastrada, a partir do controle de resenhas.

Código	Requisito funcional
RF-08	Excluir resenha com confirmação: permitir excluir resenha existente exibindo confirmação antes da remoção e informando sucesso após a operação.
RF-09	Cadastrar proprietário: permitir cadastro de proprietário com dados de identificação e endereço (ex.: nome, telefone, rua, bairro, complemento), persistindo as informações no banco.
RF-10	Cancelar cadastro com confirmação de descarte: ao cancelar/descartar alterações, solicitar confirmação; "Sim" retorna ao controle e "Não" mantém o usuário no cadastro.
RF-11	Buscar proprietário: permitir pesquisa por múltiplos atributos (ex.: proprietário, telefone, rua, bairro, complemento), com seleção por valores existentes (e opção "todos" quando aplicável).
RF-12	Editar e excluir cadastro do proprietário: permitir edição e exclusão do proprietário selecionado, com comandos explícitos para salvar alterações e remover o registro.
RF-13	Cadastrar dados do equídeo (Parte 1): permitir inserir dados do equídeo (ex.: nome, sexo, idade, espécie, cidade/estado, cor/subcor etc.), com validações e ações de salvar/limpar.
RF-14	Selecionar marcações em regiões zootécnicas por imagem clicável (Parte 2): permitir seleção de regiões em imagem interativa, com inclusão de múltiplas marcações e possibilidade de remoção.
RF-15	Registrar detalhes de cada marcação: para cada marca selecionada, registrar *Local*, *Marca* e *Região zootécnica*, com opções provenientes do banco e coerentes com a região clicada.
RF-16	Usar mapeamento técnico das imagens (HTML/*areas*): utilizar o mapeamento de coordenadas/áreas para suportar o clique e identificação das regiões zootécnicas na imagem do animal.
RF-17	Confirmar resenha (Parte 3): exibir resumo das regiões selecionadas e permitir confirmação/finalização do registro da resenha.

Tabela 3 – Análise de requisitos funcionais do sistema de resenha (consolidada a partir do documento-base).

Fonte: Próprio autor, inspirado na modelagem de Batista e Carrijo (2016)

A partir desse conjunto prévio, os requisitos foram revisitados à luz do novo objetivo tecnológico deste trabalho: disponibilizar a solução como um aplicativo híbrido, executado em dispositivos móveis Android, com interface baseada em navegador e integração direta com um serviço de banco de dados em nuvem. Esse contexto de uso em campo traz preocupações adicionais de usabilidade e fluxo de interação, como:

- Simplicidade das telas para uso em dispositivos com telas sensíveis ao toque, sob condições de luminosidade variáveis;
- Navegação clara entre as etapas de cadastro (proprietário, animal, resenha) e as regiões zootécnicas;
- Tempo de resposta aceitável mesmo com conexão instável à internet;
- Possibilidade de retomar resenhas já iniciadas, minimizando retrabalho em caso de falhas de rede.

Os requisitos funcionais centrais mantêm a mesma essência descrita em Batista e Carrijo (2016): cadastrar proprietários e animais, registrar a resenha zootécnica com marcações por região, armazenar e recuperar descrições textuais completas e permitir a consulta posterior das resenhas realizadas.

4.2 Modelagem (diagramas UML, casos de uso)

A modelagem conceitual do domínio também reutilizou os modelos do trabalho de Batista e Carrijo (2016), que apresenta um conjunto de diagramas UML (*Unified Modeling Language*) abrangendo casos de uso, classes e, em alguns pontos, interações entre componentes do sistema. Esses modelos capturam os principais atores envolvidos no processo de resenha (por exemplo, médico-veterinário responsável, proprietário, sistema de resenha) e os casos de uso fundamentais, tais como:

- Cadastrar proprietário;
- Cadastrar equídeo;
- Registrar resenha zootécnica;
- Consultar resenha previamente registrada;
- Emitir relatório ou descrição completa da resenha.

No nível de classes, o modelo apresentado por Batista e Carrijo (2016) organiza entidades como *Equídeo*, *Proprietário*, *Médico-veterinário* e *taxonomias de identificação do animal* (raça, pelagem/cor, subcor, particularidades). Tal modelagem relacional foi originalmente pensada para um SGBD PostgreSQL, mas sua estrutura conceitual permaneceu válida para orientar a implementação neste trabalho, mesmo com a adoção de um banco de dados NoSQL.

Assim, a estratégia adotada neste trabalho foi utilizar os diagramas UML do trabalho base como referência e guia conceitual, preservando a semântica zootécnica e o

fluxo de casos de uso, enquanto as adaptações tecnológicas (como o uso de listas embarcadas e documentos do *Firestore*) foram tratadas em nível de arquitetura e implementação. Dessa forma, evita-se retrabalho na fase de modelagem, concentrando o esforço deste trabalho na adaptação tecnológica e na construção do protótipo funcional.

4.3 Arquitetura do sistema

A arquitetura do sistema desenvolvido neste trabalho combina uma *aplicação web* SPA escrita em **React** com empacotamento híbrido via **Apache Cordova** para Android (DESHPANDE, 2021) e uso do ***Cloud Firestore*** como banco de dados *NoSQL* em nuvem (FIRESTORE, 2022). Essa escolha visa reduzir a necessidade de infraestrutura própria (servidor de aplicação dedicado e SGBD relacional), ao mesmo tempo em que permite a entrega de um aplicativo instalável em dispositivos móveis utilizados em campo.

No lado cliente, a aplicação React é responsável por toda a interação com o usuário: cadastro de proprietários e animais, navegação entre as regiões zootécnicas, marcação de particularidades sobre mapas de imagem e geração da descrição textual da resenha. Essa aplicação é empacotada pelo Cordova em um contêiner Android, o que viabiliza a distribuição como *app* e o acesso a funcionalidades do dispositivo (armazenamento local, integração com a câmera em versões futuras, etc.), preservando o modelo de desenvolvimento *web* com HTML, CSS e *JavaScript* (DESHPANDE, 2021).

A comunicação com o *backend* é realizada diretamente entre o aplicativo Cordova e o serviço gerenciado do ***Firestore***, por meio do *SDK* oficial do *Firebase*. Não há um servidor de aplicação intermediário: as operações de criação, leitura, atualização e remoção são invocados pelo cliente, respeitando as regras de segurança configuradas no próprio *Firestore*. A Figura 23 ilustra, em alto nível, o fluxo de interação entre o usuário, a aplicação de resenha e a persistência dos dados na nuvem. Inicialmente, o usuário realiza o acesso ao sistema por meio da tela de login, sendo autenticado via serviço do *Firebase*. Após a autenticação, a aplicação navega para a tela de visão geral, carregando o contexto do animal selecionado, identificado pelo parâmetro *animalId*.

A partir dessa tela, o usuário pode selecionar uma região zootécnica específica do animal (por exemplo, cabeça, membros ou pescoço), sendo então redirecionado para a página correspondente à região escolhida. Nessa etapa, a aplicação carrega a imagem associada ao mapa da região, juntamente com suas dimensões originais, que são utilizadas como base para o cálculo de escalonamento das áreas clicáveis.

Para garantir a precisão da interação em diferentes dispositivos, é realizado o recálculo das coordenadas das áreas do mapa (*scaleCoords*) sempre que a imagem é carregada ou redimensionada. Esse processo assegura que os pontos de clique permaneçam consistentes independentemente da resolução da tela.

Quando o usuário realiza um clique em uma área do mapa, a aplicação exibe um modal para seleção do tipo de marca a ser registrada (por exemplo, estrela, redemoinho, entre outras). Após a confirmação, os dados da marca — incluindo identificador do animal, região, área selecionada, tipo da marca e coordenadas relativas — são enviados ao serviço de aplicação responsável pela lógica de persistência.

Esse serviço, por sua vez, realiza a gravação das informações no banco de dados *Firestore*, armazenando a marca como um documento associado ao animal. Após a confirmação da escrita, a aplicação recebe o retorno com os dados da marca criada e realiza a renderização do respectivo ícone na interface, posicionando-o corretamente com base nas coordenadas escaladas.

Adicionalmente, o sistema utiliza um mecanismo de escuta em tempo real (*onSnapshot*) para acompanhar alterações na coleção de marcas do animal. Dessa forma, quaisquer atualizações são automaticamente refletidas na interface, mantendo a lista de marcas sincronizada com o estado persistido no banco de dados.

Por fim, o usuário pode retornar à tela de visão geral, preservando o contexto do animal selecionado, permitindo a continuidade do processo de preenchimento da resenha em outras regiões. Esse fluxo evidencia a integração entre interface interativa, lógica de aplicação e persistência em nuvem, garantindo consistência, responsividade e atualização em tempo real dos dados.

Do ponto de vista dos dados, a arquitetura proposta neste trabalho adapta o modelo relacional definido no trabalho base de Batista e Carrijo (2016) para o contexto de um banco de dados *NoSQL* em nuvem. Como já citado, o autor propõe em seu trabalho um esquema em *PostgreSQL* para raças, pelagens/cores, subcores e particularidades; porém, neste trabalho, optou-se por manter essas taxonomias como **listas embarcadas no código** (por serem relativamente estáveis e de cardinalidade limitada) e utilizar o *Firestore* apenas para as entidades dinâmicas: animais, proprietários e resenhas. Essa decisão simplifica o modelo físico no *Firestore*, evitando a necessidade de coleções específicas para cada taxonomia e reduzindo a complexidade de sincronismo, sem perder a semântica zootécnica construída no trabalho anterior (BATISTA; CARRIJO, 2016).

As resenhas são representadas como documentos que agregam informações do animal, do proprietário e das marcações realizadas em cada região zootécnica. Para cada marca, são armazenados campos como `tipo`, `regiao`, `mapName`, `areaId`/`areaLabel` e, principalmente, as **coordenadas relativas** da marcação (normalizadas em relação à largura e altura originais da imagem). A Figura 24 apresenta um exemplo de documento no *Firestore* para uma marca, evidenciando os campos principais usados pela aplicação. À esquerda, é possível ver o documento do animal dentro da coleção `animais`, contendo atributos gerais como nome, sexo, idade, espécie, cor, subcor, raça e identificador do proprietário. No centro, observa-se a subcoleção `marcas`, associada a esse animal, na qual

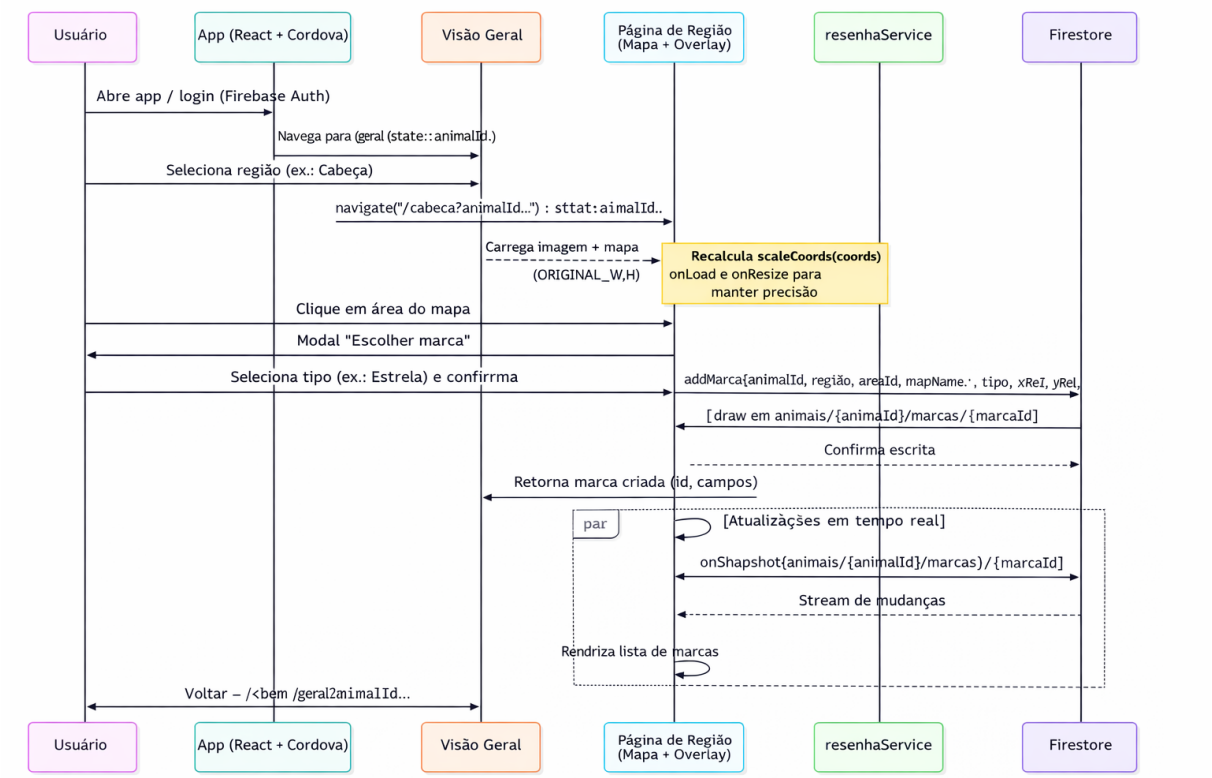


Figura 23 – Diagrama de alto nível: interação do usuário, marcação por regiões e persistência no *Firestore*.

Fonte: Próprio autor

cada documento representa uma marca registrada durante o preenchimento da resenha. À direita, são exibidos os campos de um desses documentos de marca, incluindo o identificador da área selecionada no mapa (`areaId`), o rótulo textual da área (`areaLabel`), a data de criação (`createdAt`), o identificador do usuário responsável pelo registro (`createdBy`), o ícone utilizado para representar graficamente a marca (`icon`), o nome do mapa de origem (`mapName`), a região zootécnica correspondente (`regiao`), o tipo da marca registrada (`tipo`) e as coordenadas relativas `x` e `y`, empregadas para reposicionar a marca de forma responsiva na interface. Essa organização evidencia que os dados gerais do animal permanecem no documento principal, enquanto as ocorrências visuais da resenha são armazenadas de forma separada, porém vinculada, na subcoleção de marcas, facilitando tanto a persistência quanto a recuperação estruturada dessas informações.

A escolha por coordenadas relativas faz parte da própria arquitetura de apresentação: na reexibição das resenhas, a aplicação recalcula a posição de cada marca com base na dimensão atual da imagem renderizada no dispositivo, garantindo consistência visual em diferentes tamanhos e resoluções de tela. Assim, a camada de interface permanece responsiva e independente de um tamanho fixo de mapa, característica essencial para uso em variados modelos de smartphones e tablets.

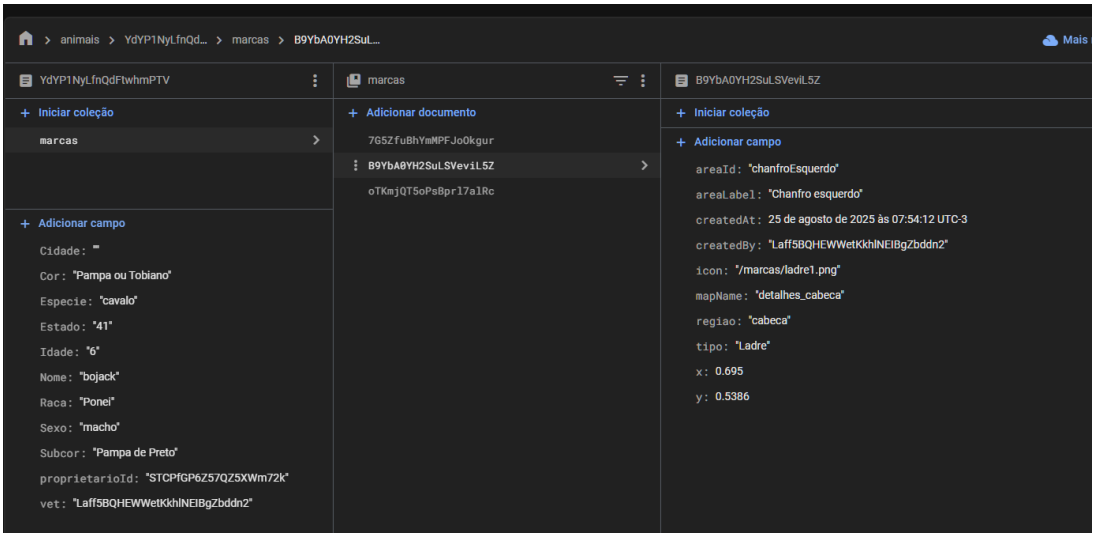


Figura 24 – Exemplo de documento no *Firestore* para uma marca (campos principais).

Fonte: Próprio autor

A navegação dentro da aplicação React utiliza `HashRouter`, o que se mostrou mais compatível com o contêiner do Cordova. O identificador do animal (`animalId`) é propagado de forma **redundante**: (i) no `state` repassado ao navegar entre as telas; e (ii) como parâmetro `?animalId=` na URL. Essa estratégia assegura compatibilidade com telas mais antigas que dependiam da *querystring*, ao mesmo tempo em que permite que as telas novas utilizem o `state` de maneira mais idiomática em React. Na prática, essa solução se mostrou robusta no contexto do aplicativo híbrido, mesmo em cenários de recarregamento da *WebView* ou retomada da aplicação em segundo plano.

Em síntese, a arquitetura do sistema equilibra três eixos principais: (i) preservação da modelagem conceitual e das regras de negócio propostas por Batista e Carrijo (2016); (ii) uso de uma pilha tecnológica leve e amplamente difundida (React, Cordova, *Firestore*) para viabilizar a entrega em campo; e (iii) decisões específicas de representação de dados (como coordenadas relativas e listas embarcadas de taxonomias) que simplificam a implementação e favorecem a portabilidade da solução.

4.4 Implementação

Esta seção descreve as principais funcionalidades implementadas no aplicativo híbrido (React + Cordova) voltado à informatização da **resenha** de equídeos (equinos e muares). A implementação foi organizada para permitir um fluxo de uso direto em campo: autenticar, cadastrar (ou selecionar) proprietário e animal, preencher a resenha por regiões zootécnicas e, ao final, revisar os dados antes do encerramento.

4.4.1 Autenticação e acesso ao sistema

O acesso ao aplicativo inicia pela tela de *login* (Figura 25), responsável por autenticar o usuário antes de liberar as rotas internas do sistema. Conforme observado na imagem, essa tela apresenta um formulário objetivo, composto por dois campos de entrada — *CRMV* e *senha* —, além de um botão para confirmar o *login* e um *link* de apoio *"Esqueceu sua senha?"* para recuperação de acesso. Essa etapa restringe o cadastro e a edição de dados de proprietários/animais a usuários autorizados e também serve como ponto de partida para recuperar o contexto de trabalho, como a continuidade de uma resenha previamente iniciada.

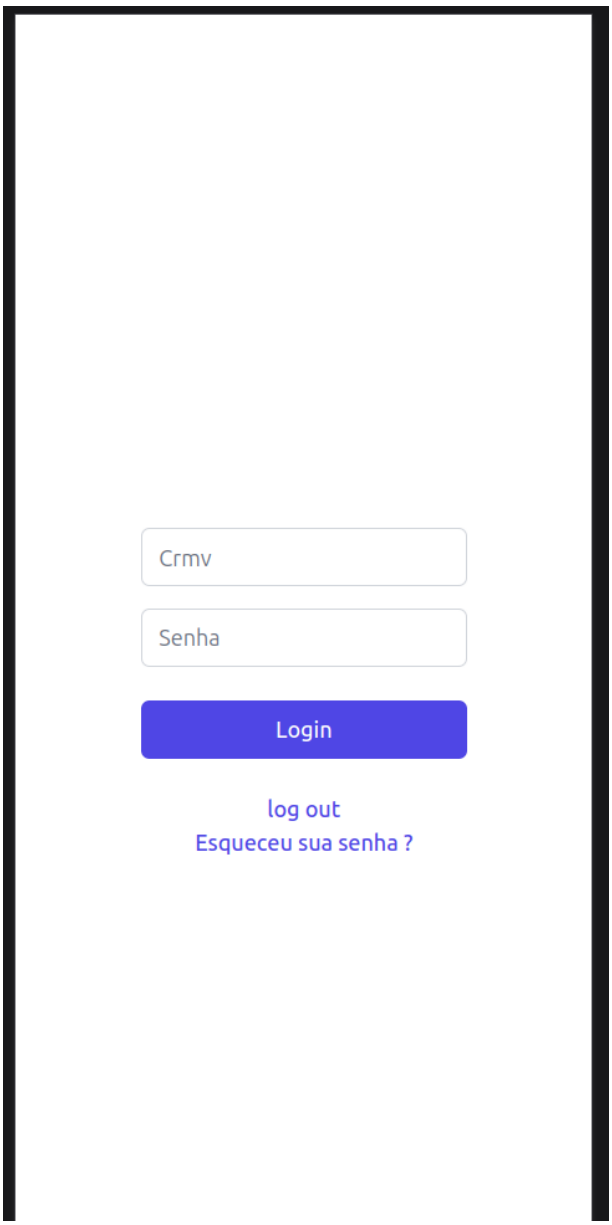


Figura 25 – Tela de login do aplicativo.

Fonte: Próprio autor

4.4.2 Cadastro e gestão de proprietários

O módulo de **proprietário** permite registrar e manter os dados do responsável pelo animal. Conforme apresentado na Figura 26, a tela exibe o formulário *Dados do Proprietário*, com campos para *Nome do Proprietário*, *Telefone* (com exemplo de formatação), *Rua*, *Bairro* e *Complemento (opcional)*, além de um botão *"Cadastrar proprietário"* para confirmar o cadastro e um ícone de retorno no topo para voltar à etapa anterior. Do ponto de vista de persistência, cada proprietário é armazenado como um documento na coleção correspondente do *Firestore*, contendo seus atributos e referências para os animais associados. Essa organização facilita a busca e a reutilização do cadastro do proprietário em múltiplas resenhas.

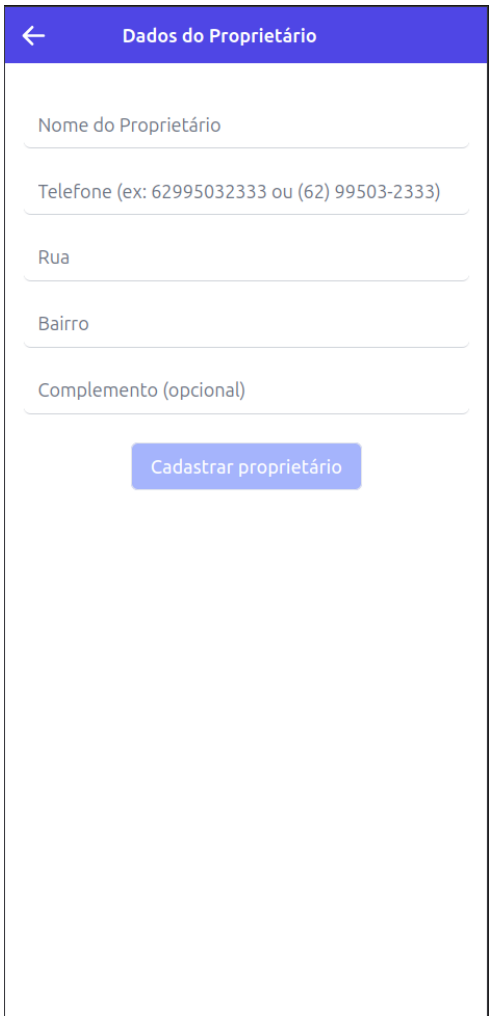


Figura 26 – Formulário de cadastro de proprietários.

Fonte: Próprio autor

4.4.3 Cadastro e gestão de animais

O módulo de **animal/equídeo** permite cadastrar as informações gerais do indivíduo (identificação e atributos descritivos necessários ao processo). Conforme ilustrado na

Figura 27, a tela *Dados do equídeo* apresenta campos para *Nome do animal*, seleção de *Sexo* (*Macho* ou *Fêmea*), *Idade do animal* e *Espécie*, além de seletores em formato de lista para *Raça* e *Proprietário*. Também são disponibilizados campos para *Estado* e *Cidade*, bem como seletores para *Cor* e *Sub-cor*, finalizando com o botão *"Cadastrar animal"* para confirmar o registro e um ícone de retorno no topo da tela. Cada animal é persistido como documento, mantendo vínculo com o proprietário. Além desses dados básicos (como *nome*, *idade* e *sexo*), o modelo também está preparado para guardar as marcações feitas nas telas de mapa durante a resenha, usando categorias padronizadas. Isso ajuda a manter os registros organizados e consistentes, além de facilitar a busca e a comparação dessas informações em resenhas futuras, simplificando a implementação e favorecendo a portabilidade da solução.



Figura 27 – Formulário de cadastro de animais.

Fonte: Próprio autor

4.4.4 Fluxo de resenha e controle de preenchimento

O preenchimento da resenha é conduzido por um fluxo de navegação que preserva o **identificador do animal** ao longo das telas, garantindo que cada etapa atualize o mesmo registro em edição. Para apoiar o andamento do processo, foi implementada uma tela de **controle da resenha** (Figura 28), que centraliza a seleção do animal e o acesso às etapas de preenchimento. Como se observa na imagem, a tela exibe o usuário autenticado (com opção de *log out*), filtros para *Proprietário*, *Nome do animal*, *Cor* e *Sub-cor*, e uma seção de *Lista de animais* com os resultados da busca, apresentando informações como proprietário, nome do animal, cor e sub-cor. A tabela de resultados também inclui as colunas *Editar* e *Excluir*: a opção *Editar* permite acessar a edição das marcas (particularidades) do animal, enquanto *Excluir* remove o animal da base de dados juntamente com sua resenha associada. Além disso, é apresentada uma dica para refinamento por filtros. A partir dessa listagem, o usuário consegue localizar rapidamente o indivíduo desejado, retomar o preenchimento de uma resenha já iniciada e revisar o cadastro antes de concluir. Essa estratégia reduz inconsistências e melhora a usabilidade, especialmente quando a coleta ocorre em etapas, conforme as condições de manejo.

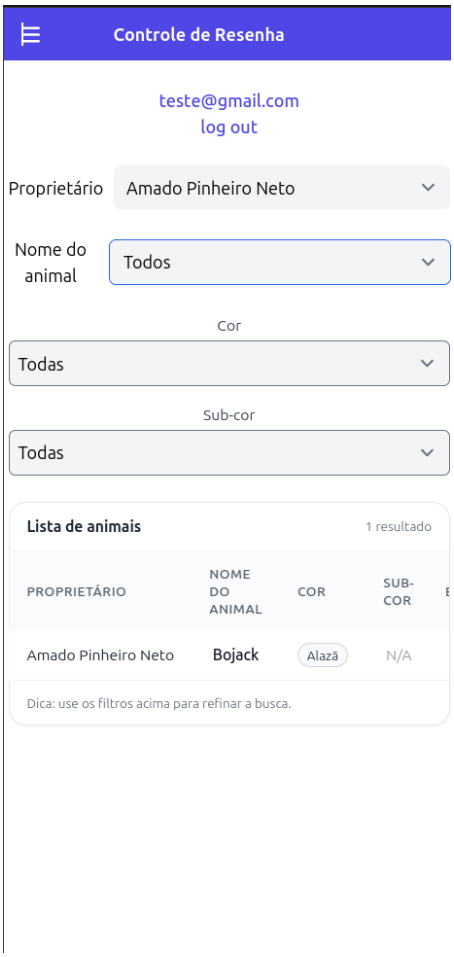


Figura 28 – Tela de controle da resenha.

Fonte: Próprio autor

4.4.5 Mapeamento visual por regiões zootécnicas

A funcionalidade central do aplicativo é a coleta visual de características por meio de *mapas de imagem* (*image maps*), permitindo selecionar áreas clicáveis diretamente sobre ilustrações do corpo do animal. Para refletir a prática da resenha e organizar a interface, o sistema foi dividido em telas por região, contemplando **lado esquerdo** e **lado direito**, **cabeça** e **focinho**, **pescoço (inferior)**, **membros anteriores** e **membros posteriores**.

Além disso, a **visão geral** passou a atuar como uma página de navegação. Conforme visto na Figura 30, a tela *Visão geral do equídeo (interativa)* apresenta *previews* das regiões zootécnicas, organizadas em cartões com rótulos, como *Membros anteriores — vista posterior*, *Lado direito*, *Pescoço — vista inferior* e *Cabeça — detalhes frontais*. Ao serem clicados, esses cartões direcionam o usuário para a página específica em que é possível realizar as marcações daquela região. A Figura 30 ilustra uma resenha cujo processo de preenchimento já foi iniciado. Tanto essa figura quanto a Figura 29 representam a resenha do mesmo animal, ainda em andamento. Observando-se o campo de descrição textual em conjunto com a visualização das imagens das regiões zootécnicas, nota-se que, nesse exemplo, existem particularidades registradas no focinho, no pescoço e no tronco.

Ao rolar a página para baixo, a mesma tela de visão geral também consolida o resultado do preenchimento em formato textual, exibindo a **descrição do animal** gerada a partir das marcações registradas em cada região (Figura 29). Essa síntese corresponde à componente descritiva tradicional da resenha e pode ser confrontada com o modelo de *resenha preenchida* apresentado na Figura 3, reforçando a equivalência entre o registro produzido no aplicativo e o padrão adotado no procedimento manual.

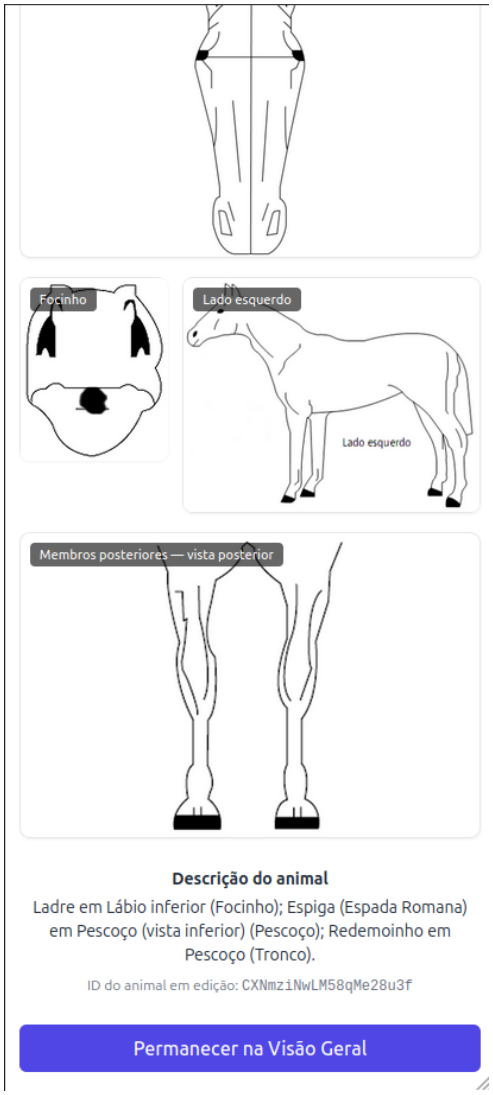


Figura 29 – Descrição da resenha.

Fonte: Próprio autor

Em cada uma dessas páginas de marcação, a imagem da respectiva região é associada a um arquivo de coordenadas, com extensão `.map`, que define as áreas selecionáveis, inclusive por meio de polígonos. Isso permite um mapeamento mais fiel ao contorno anatômico do animal do que abordagens baseadas apenas em formas simples.

Como alternativa tecnológica para a implementação dessas áreas clicáveis, considerou-se o uso de ferramentas do ecossistema *React Native*. Contudo, essa abordagem não foi adotada neste trabalho, pois, entre as bibliotecas analisadas, não se identificou uma solução que oferecesse, de forma direta, funcionalidade equivalente ao atributo `shape="poly"` da tag HTML `<area>`. Esse recurso mostrou-se especialmente adequado ao problema tratado, por permitir a definição de áreas clicáveis com contornos arbitrários, possibilitando um mapeamento mais preciso das regiões zootécnicas do animal.

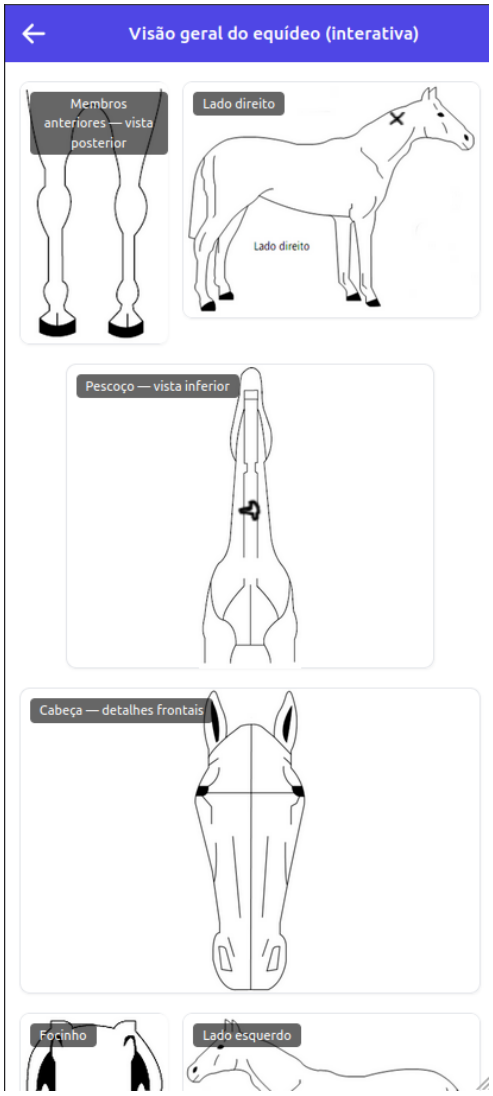


Figura 30 – Tela de visão geral de regiões zootécnicas.

Fonte: Próprio autor

4.4.6 Interação com áreas clicáveis e registro de marcas

Ao selecionar uma área do mapa, o aplicativo registra a ocorrência vinculada ao animal e à região correspondente. Conforme ilustrado na Figura 31, a tela do *Lado direito* apresenta a ilustração da região com o ponto marcado sobre o corpo do animal e, na parte superior, uma seção de *Marcas* que lista os traços disponíveis e/ou já selecionados (por exemplo, *Redemoinho*, *Espiga (Seta)*, *Lista de burro*, *Faixa crucial* e *Bragado*). Quando aplicável, o sistema permite associar **marcas/traços** pré-definidos (símbolos e padrões utilizados em resenhas), armazenando as seleções de forma estruturada para posterior revisão e consolidação. Além do registro, a navegação foi planejada para manter o usuário no contexto da resenha corrente, incluindo um botão *"Voltar à Visão Geral"* para retornar rapidamente à tela de navegação sem perder o vínculo com o animal em edição, reduzindo a chance de preenchimento em animal incorreto.



Figura 31 – Lado direito com marca.

Fonte: Próprio autor

4.4.7 Responsividade dos mapas e consistência do clique

Como o aplicativo é executado em dispositivos Android com diferentes tamanhos e densidades de tela, foi implementado um mecanismo de **reescala proporcional** das coordenadas do *image map*. As coordenadas originais, definidas para a imagem-base, são recalculadas de acordo com as dimensões efetivamente renderizadas no dispositivo, assegurando que o toque do usuário corresponda à área anatômica pretendida. Esse processo de recálculo é apresentado de forma mais completa, com explicação e ilustração do algoritmo, na seção 4.5.3.

Na prática, a renderização dos mapas foi encapsulada em um componente reutilizável, responsável por aplicar o conjunto de coordenadas de cada região, recalcular os pontos ao carregar a imagem e em eventos de redimensionamento, e encaminhar a seleção para o estado da resenha.

4.4.8 Navegação entre telas em ambiente híbrido

A navegação do aplicativo utiliza **React Router** com o componente `HashRouter`, abordagem adequada ao empacotamento via Cordova (DESHPANDE, 2021) por evitar dependência de reescritas de rota em servidor e manter o controle de navegação no fragmento da URL. Também foi adotada uma estrutura de navegação consistente (por exemplo, uma barra lateral), facilitando o acesso às telas de cadastro e às páginas de regiões zootécnicas durante a execução da resenha.

4.4.9 Persistência e continuidade do preenchimento

Os dados coletados em todas as etapas (cadastros e marcações da resenha) são persistidos no ***Firestore***, de modo que o preenchimento não dependa de uma única sessão de uso. Essa persistência favorece continuidade e rastreabilidade: o usuário pode interromper e retomar a resenha mantendo o histórico de informações registradas de forma consistente com a estrutura orientada a documentos adotada no banco de dados. Na prática, a edição pode ser continuada a qualquer momento ao acessar a tela de **visão geral** e selecionar a região zootécnica desejada, retomando diretamente a página de marcação correspondente (Figura 30).

4.5 Desafios

O desenvolvimento da aplicação envolveu desafios que vão além da implementação das funcionalidades de negócio. Por se tratar de um aplicativo móvel híbrido, foi necessário conciliar um *stack* voltado ao desenvolvimento *web* (React) com um ambiente de empacotamento e execução em *WebView* (Apache Cordova), garantindo que o mesmo código-fonte pudesse ser mantido e evoluído de forma organizada, ao mesmo tempo em que resultasse em um *APK* (*Android Package Kit*) executável em dispositivos Android.

4.5.1 Acessibilidade das áreas clicáveis

Entretanto, a implementação plena de boas práticas de acessibilidade *web* no uso de `<map>` e `<area>` mostrou-se desafiadora. Isso ocorre porque o propósito central do *software* é oferecer uma **interação visual** para representar e registrar, com precisão, as regiões zootécnicas do animal. Nessa abordagem, a compreensão do conteúdo e a seleção das áreas dependem fortemente da percepção espacial e do reconhecimento da anatomia ilustrada, o que dificulta a oferta de alternativas textuais equivalentes sem perda relevante de significado ou aumento de complexidade.

Ainda assim, diretrizes de acessibilidade do W3C (World Wide Web Consortium) recomendam que conteúdos não textuais e controles baseados em imagem disponham de

alternativas textuais equivalentes, de modo que seu propósito possa ser compreendido por tecnologias assistivas (World Wide Web Consortium (W3C), 2024; World Wide Web Consortium (W3C), 2025). Assim, apesar de terem sido consideradas alternativas como o uso de descrições textuais e rótulos associados às áreas clicáveis, sua aplicação integral não foi priorizada nesta versão, permanecendo como ponto de melhoria para trabalhos futuros.

4.5.2 Empacotamento do React como aplicação Cordova

Do ponto de vista do ciclo de construção, um primeiro desafio foi estruturar o projeto para que a base de código em React se integrasse ao modelo de diretórios e ao *pipeline* de compilação do Cordova. Enquanto o React (via Create React App) trabalha com um diretório de desenvolvimento (`src/` e `public/`) e produz uma saída otimizada em `build/`, o Cordova espera consumir os arquivos finais do aplicativo a partir do diretório `www/`, que é empacotado dentro do aplicativo nativo.

Para resolver essa divergência, foi necessário (i) consolidar a estrutura do projeto em um único repositório, preservando a ergonomia do desenvolvimento em React; (ii) automatizar a geração do *bundle* de produção e sua disponibilização no diretório `www/`; e (iii) adaptar pontos específicos para o contexto de execução em *WebView*.

Em termos práticos, o fluxo de construção adotado pode ser descrito em quatro etapas:

1. **Unificação da base de código:** os diretórios `src/` e `public/` passam a fazer parte do projeto Cordova, e as dependências e *scripts* do `package.json` são consolidados para que o projeto mantenha a experiência de desenvolvimento do React.

2. **Ajustes no `public/index.html`:** como o aplicativo roda em ambiente *mobile*, foram adicionadas configurações típicas de aplicativos híbridos, como *meta tags* de *viewport*/detecção de telefone e uma política de segurança de conteúdo (*Content Security Policy* — CSP). Além disso, o carregamento do `cordova.js` foi incluído (via `<script src="cordova.js">`), pois o arquivo é injetado pelo Cordova durante a execução e dá acesso às APIs e *plugins* nativos.

3. **Sincronização com o evento `deviceready`:** ao contrário de uma aplicação *web* tradicional, as APIs do Cordova só estão disponíveis após o disparo do evento `deviceready`. Por isso, o *bootstrap* do React (montagem da árvore de componentes) foi condicionado a esse evento, reduzindo erros de inicialização e garantindo que, ao acessar funcionalidades nativas, o ambiente já esteja pronto.

4. **Automação do *build* e cópia para `www/`:** como o Cordova empacota os arquivos de `www/`, tornou-se necessário automatizar a etapa em que o React gera o *build* e o resultado é disponibilizado no local esperado pelo Cordova. Para

isso, foi configurado um *hook* do Cordova do tipo `before_prepare` (por exemplo, `<hook type="before_prepare" src="scripts/prebuild.js" />`), responsável por executar o *build* do React e, em seguida, substituir o conteúdo de `www/` pelo conteúdo gerado.

Um ponto crítico nesse processo foi o tratamento de caminhos de arquivos estáticos. Por padrão, o *build* do React gera referências absolutas (por exemplo, iniciando em `/static/...`), o que funciona bem quando há um servidor HTTP, mas tende a falhar quando a aplicação é carregada diretamente no contexto de *WebView* (*e.g.*, via `file://`). Assim, foi necessário ajustar a propriedade `"homepage": "./"` no `package.json`, de modo que os *assets* fossem referenciados por caminhos relativos, compatíveis com o empacotamento do Cordova.

Além do empacotamento em si, esse arranjo introduz desafios adicionais de manutenção: sempre que o aplicativo é executado ou gerado para distribuição, o *build* precisa ser refeito e sincronizado com `www/`; eventuais mudanças em dependências do React impactam o *bundle* final; e o processo de depuração em dispositivo tende a ser menos direto do que em um navegador tradicional, exigindo atenção a *logs* e ferramentas de inspeção.

Por fim, no contexto de navegação entre telas, também foi necessário considerar limitações de roteamento típicas de aplicações empacotadas, como a preferência por `<HashRouter>` em vez de `<BrowserRouter>`, evitando problemas de resolução de rotas quando não há servidor para reescrita de URLs, como foi previamente explicado na seção 4.4.8.

4.5.3 Responsividade dos mapas de imagem e recálculo de coordenadas

Um desafio adicional esteve relacionado ao uso de *image maps* em HTML para seleção das regiões zootécnicas. No trabalho original, os mapas foram gerados no GIMP a partir das imagens recortadas de cada região, o que faz com que as coordenadas das áreas (`<area shape="poly">`) fiquem definidas em pixels absolutos na resolução nativa de cada imagem. Entretanto, no contexto do aplicativo móvel, a mesma tela pode ser renderizada em diferentes tamanhos e densidades, exigindo redimensionamento da imagem (por exemplo, com `width: 100%`) para manter boa usabilidade. Como consequência, o mapeamento desloca e o toque do usuário passa a não corresponder à região correta.

Para preservar a precisão do mapeamento em múltiplas resoluções, foi necessário implementar um algoritmo de reescala das coordenadas. A ideia é tratar cada vértice do polígono como um ponto (x, y) definido no mapeamento original feito via GIMP, considerando as dimensões originais da imagem, $W_{orig} \times H_{orig}$, em que W_{orig} representa a largura original do arquivo de imagem e H_{orig} representa sua altura original. Esses pontos são então projetados para o sistema de coordenadas da imagem no instante em que ela

é exibida ao usuário, isto é, para $W_{atual} \times H_{atual}$, em que W_{atual} corresponde à largura efetivamente renderizada na tela e H_{atual} à altura efetivamente renderizada na tela, após o carregamento da página e eventuais adaptações ao tamanho do dispositivo. Desse modo, aplicam-se fatores de escala independentes em cada eixo, conforme a Equação 1:

$$x' = x \cdot \left(\frac{W_{atual}}{W_{orig}}\right) \qquad y' = y \cdot \left(\frac{H_{atual}}{H_{orig}}\right) \tag{1}$$

Equação 1 – Reescala independente das coordenadas nos eixos x e y.

Em outras palavras, a razão $\frac{W_{atual}}{W_{orig}}$ indica o quanto a imagem foi ampliada ou reduzida horizontalmente em relação ao arquivo original, enquanto $\frac{H_{atual}}{H_{orig}}$ indica essa mesma variação no eixo vertical.

O recálculo é disparado no carregamento da imagem e em eventos de redimensionamento/orientação, utilizando `naturalWidth`/`naturalHeight` para obter as dimensões originais e `clientWidth`/`clientHeight` para as dimensões atuais, atualizando dinamicamente o atributo `coords` de cada `<area>`.

A seguir, apresenta-se um exemplo simplificado do padrão adotado para o recálculo das coordenadas, descrito em pseudocódigo nos Algoritmos 1 e 2.

Algoritmo 1: Reescala de coordenadas para *image map* (cálculo pontual).

```
1  Function SCALE-COORDS(coordsStr, scaleW, scaleH):
2      nums ← PARSE-COORDS(coordsStr)
3      scaled ← lista vazia
4      for i ← 0 to |nums| − 1 do
5          if i é par then
6              scaled[i] ← nums[i] × scaleW
7          else
8              scaled[i] ← nums[i] × scaleH
9          scaled[i] ← ROUND(scaled[i])
10     retorne JOIN-WITH-COMMA(scaled)
```

Algoritmo 2: Recomputação do conjunto de áreas escaladas (atualização do conjunto).

```
1  Procedure
     RECOMPUTE(imgRef, hasValidBase, originalWidth, originalHeight, areas):
2      img ← imgRef.current
3      if img = NIL ou hasValidBase = falso then
4          retorne NIL
5      scaleW ← img.clientWidth/originalWidth
6      scaleH ← img.clientHeight/originalHeight
7      next ← lista vazia
8      foreach a em areas do
9          a2 ← COPY(a)
10         a2.scaledCoords ← Scale-Coords(a.coords, scaleW, scaleH)
11         adicionar a2 em next
12     retorne next
```

A implementação do recálculo de coordenadas adotada no projeto segue a mesma lógica de proporcionalidade, porém organizada em duas rotinas, conforme os Algoritmos 1 e 2.

No Algoritmo 1, as coordenadas originais são recebidas como uma *string* no formato "x1,y1,x2,y2,..." e transformadas em uma lista de valores numéricos na etapa PARSE-COORDS. Em seguida, aplica-se a escala conforme a posição do índice: valores em índices pares correspondem ao eixo x e recebem o fator $scaleW$, enquanto valores em índices ímpares correspondem ao eixo y e recebem o fator $scaleH$. Após esse processamento, cada valor é arredondado por meio da operação ROUND. Por fim, a função JOIN-WITH-COMMA recompõe a sequência de coordenadas em uma única *string*, separando os valores por vírgula, de modo a reconstruir exatamente o formato exigido pelo atributo `coords` da tag HTML `<area>`.

O Algoritmo 2 centraliza o momento de atualização. Primeiro, obtém-se a referência da imagem renderizada (`imgRef.current`) e valida-se se há uma base consistente de dimensões originais (`hasValidBase`), prevenindo cálculos com valores indefinidos. Em seguida, os fatores de escala são calculados pela razão entre o tamanho atual renderizado e o tamanho original conhecido. Por fim, o conjunto de áreas é reconstruído, preservando as demais propriedades e substituindo apenas o campo `scaledCoords` pelo resultado de SCALE-COORDS. Essa lista atualizada é armazenada no estado (`setScaledAreas(next)`), fazendo com que as regiões clicáveis permaneçam alinhadas à imagem em qualquer tamanho de tela, sem acumular erro de arredondamento entre recomputações.

Os desafios descritos nesta subseção fundamentam decisões técnicas do projeto

(organização do repositório, automação do *build* e ajustes de execução em *WebView*) e fornecem o contexto para outras dificuldades enfrentadas ao longo do desenvolvimento, como integração com *plugins* nativos, permissões no Android, e a necessidade de garantir responsividade e usabilidade em telas de diferentes tamanhos, incluindo o recálculo dinâmico das coordenadas de *image maps* para manter o alinhamento das regiões zootécnicas em diferentes resoluções.

5 Resultados

Este capítulo apresenta os resultados obtidos com a implementação do protótipo de aplicativo móvel para informatização da resenha zootécnica de equinos e muares. Os resultados são descritos a partir do que foi efetivamente construído e validado durante o desenvolvimento (implementação incremental, testes unitários e testes funcionais/exploratórios em ambiente Android), buscando relacionar as entregas com os objetivos estabelecidos no início do trabalho.

5.1 Retomada dos objetivos e atendimento alcançado

A seção de objetivos, apresentada no Capítulo 1, orientou as decisões de arquitetura, a priorização de requisitos e a forma de validação do protótipo. Nesta seção, retomam-se o objetivo geral e os objetivos específicos, comparando-os com os resultados obtidos.

5.1.1 Objetivo geral

O objetivo geral definido foi desenvolver um aplicativo móvel, inspirado no trabalho de Batista e Carrijo (2016), integrando-o ao ecossistema do *Firebase* (em especial o *Cloud Firestore*), de modo a apoiar o médico-veterinário na elaboração da resenha de identificação e contribuir para maior precisão e confiabilidade do registro.

5.1.2 Objetivos específicos

Os objetivos específicos estabelecidos foram:

1. Estudar as particularidades e seus significados no contexto de identificação de equinos e muares, visando embasar a estrutura de cadastro e os campos da resenha.
2. Criar uma base de imagens e referências de marcas naturais/particularidades para uso no processo de resenha, aproveitando e ampliando os materiais mapeados no trabalho anterior.
3. Projetar e implementar um aplicativo executável em dispositivos Android, com persistência em *Cloud Firestore* e funcionalidades essenciais do controle de resenhas (cadastro de proprietário, cadastro de animal, registro de marcações por regiões zootécnicas e consulta das resenhas).

5.1.3 Comparação entre objetivos e resultados

A partir do protótipo desenvolvido, observa-se que o objetivo geral foi atendido no escopo de uma *versão mínima viável*: a aplicação foi implementada como aplicativo móvel híbrido (React empacotado via Apache Cordova para Android), com autenticação e persistência de dados no *Cloud Firestore*. Esse conjunto viabiliza o uso em campo por meio de um fluxo guiado (cadastros e resenha) e mantém o histórico das informações registradas, favorecendo precisão e consistência.

Quanto aos objetivos específicos, o atendimento ocorreu da seguinte forma:

Estudo das particularidades: o referencial teórico consolidou as categorias de identificação (pelagens, cores/subcores e particularidades) e suas descrições, que foram utilizadas como base para estruturar campos do cadastro e as opções de seleção durante o preenchimento.

Base de imagens e referências: o projeto incorporou o mapeamento de regiões e particularidades sobre imagens (*image maps* HTML), além de manter um conjunto de recursos visuais (imagens das regiões e de marcas naturais) como apoio ao registro guiado.

Aplicativo Android com *Firestore*: foi entregue um protótipo funcional com telas de autenticação, cadastro/consulta de proprietários, cadastro/consulta de animais e fluxo de resenha por regiões zootécnicas, incluindo persistência em nuvem e possibilidade de continuidade do preenchimento.

5.2 Resultado principal entregue

O resultado principal deste trabalho é um aplicativo móvel que informatiza o fluxo de resenha, substituindo o preenchimento manual em papel por um processo guiado em dispositivo Android. Na prática, a aplicação centraliza, em uma mesma solução, as etapas de cadastro (*proprietário* e *animal*) e o registro das marcações por regiões zootécnicas, com persistência em nuvem.

Em termos funcionais, o protótipo contempla:

- autenticação do usuário (médico-veterinário), controlando o acesso ao módulo de resenhas;
- cadastro e gestão de proprietários e de animais, com possibilidade de consulta e edição;
- registro da resenha zootécnica em etapas, com seleção de regiões em mapas clicáveis, inclusão/remoção de marcações e registro de detalhes por marca;

- armazenamento e recuperação das informações no *Cloud Firestore*, permitindo rastreabilidade e consulta posterior;
- tratamento de responsividade dos mapas de imagem, preservando consistência de clique em diferentes tamanhos de tela.

5.3 Evidências de funcionamento e validação do protótipo

Foram empregados testes unitários em *JavaScript* com a biblioteca *Jest*, nos quais foram criados casos de teste automatizados para validar funções responsáveis pelo processamento dos dados de resenha, como manipulação de coordenadas, criação de objetos de marca e lógica de persistência. Esses testes foram estruturados a partir da definição de entradas controladas e da verificação dos resultados esperados, garantindo que as funções retornassem valores corretos e consistentes para diferentes cenários de uso. Além disso, componentes críticos da interface foram testados de forma isolada, assegurando o correto comportamento diante de interações do usuário

O foco dos testes manuais esteve em:

- navegação entre telas e etapas do fluxo (cadastro de proprietário, cadastro de animal e resenha);
- comportamento das áreas clicáveis nos mapas de imagem (seleção, remoção e consistência da marcação);
- integração com o *Cloud Firestore* (criação, leitura e atualização de registros) e consistência dos dados persistidos;
- retomada de preenchimentos e manutenção do estado da resenha durante a execução.

Embora os testes realizados não substituam uma avaliação formal com usuários em campo, acompanhada de métricas sistemáticas de desempenho, erros e satisfação, eles permitiram verificar empiricamente o funcionamento dos elementos centrais da solução proposta. Observou-se, nos cenários testados, que o protótipo foi capaz de executar adequadamente a navegação entre etapas, o registro de marcações nas regiões zootécnicas, a persistência dos dados e a retomada de resenhas em andamento. Assim, os testes fornecem evidências de que a aplicação é tecnicamente executável e funcionalmente coerente com os requisitos definidos, sustentando, nesse nível, a conclusão de viabilidade do processo informatizado proposto.

5.4 Benefícios gerados

Os benefícios descritos a seguir são derivados da comparação entre o fluxo manual de resenha e o fluxo informatizado entregue pelo protótipo. Eles se organizam em quatro dimensões: operacional, padronização/qualidade de dados, organização/rastreabilidade e benefícios técnicos.

5.4.1 Benefícios operacionais (processo de resenha)

A informatização da resenha impacta diretamente o processo operacional de identificação, ao substituir o preenchimento manual em papel por um fluxo guiado em dispositivo móvel. No sistema desenvolvido, os dados cadastrados no aplicativo são persistidos em um banco de dados centralizado em nuvem, por meio do *Cloud Firestore*, o que permite concentrar, em um mesmo ambiente, as informações de proprietários, animais e marcações realizadas durante a resenha. Dessa forma, reduz-se a fragmentação da informação e amplia-se a possibilidade de recuperação posterior dos registros.

Essa centralização mostra-se especialmente vantajosa quando comparada aos riscos inerentes ao uso de resenhas em papel. Fichas físicas podem ser extraviadas, danificadas por umidade, desgaste ou manejo em campo, além de permanecerem em posse de diferentes pessoas, o que dificulta a manutenção do histórico documental do animal. No processo informatizado, por sua vez, os registros ficam associados ao animal e armazenados em nuvem, facilitando sua recuperação e reduzindo a probabilidade de perda desse histórico.

Entre os ganhos operacionais, destacam-se:

- **Agilidade em campo:** o fluxo por etapas orienta o preenchimento e reduz a necessidade de transcrição posterior.
- **Continuidade do preenchimento:** a persistência em banco centralizado permite interromper e retomar a resenha posteriormente, sem perda dos dados já informados.
- **Clareza do fluxo:** a separação em módulos (proprietário, animal e resenha) facilita a execução e o treinamento de usuários.

5.4.2 Benefícios de padronização e qualidade de dados

A padronização é favorecida pelo uso de campos estruturados, validações e listas de seleção para informações zootécnicas. Ao reduzir a dependência de descrições livres e de interpretações variáveis entre profissionais, o sistema contribui para maior consistência do registro.

Os principais benefícios nessa dimensão incluem:

- **Redução de ambiguidade:** opções padronizadas para pelagens, cores/subcores e particularidades auxiliam na uniformidade das descrições.
- **Validações e coerência:** o fluxo guiado permite restringir escolhas e exigir campos essenciais, reduzindo omissões.
- **Melhor legibilidade:** registros digitais evitam problemas comuns de caligrafia e deterioração de documentos físicos.

5.4.3 Benefícios de organização, consulta e rastreabilidade

Ao persistir os dados em banco de dados (*Cloud Firestore*) e manter uma estrutura de entidades (proprietários, animais, resenhas e marcações), o sistema melhora a capacidade de organização, busca e rastreamento do histórico.

Os benefícios incluem:

- **Consulta facilitada:** localizar registros de um proprietário ou animal torna-se mais rápido do que buscar documentos físicos em arquivos.
- **Histórico centralizado:** as resenhas ficam associadas ao animal e armazenadas no *Cloud Firestore*, favorecendo a rastreabilidade ao longo do tempo.
- **Reuso de informação:** dados cadastrais podem ser reaproveitados em novas resenhas, evitando duplicidade de preenchimento.

5.4.4 Benefícios técnicos (portabilidade/manutenibilidade)

Do ponto de vista técnico, a solução adotou uma pilha baseada em *JavaScript* e React, empacotada para Android via Apache Cordova, com persistência em nuvem por meio do *Cloud Firestore*. Essa composição gera benefícios relevantes de portabilidade e manutenção.

Os principais benefícios técnicos incluem:

- **Portabilidade para Android com base *web*:** o empacotamento híbrido permite distribuir e executar a aplicação em dispositivos Android mantendo o modelo de desenvolvimento *web*, o que favorece ciclos rápidos de melhoria.
- **Redução de infraestrutura própria:** ao utilizar serviços gerenciados (por exemplo, *Firestore*), evita-se a necessidade de implantar e administrar servidor dedicado e banco de dados relacional.

- **Separação entre dados dinâmicos e taxonomias estáveis:** manter taxonomias como listas embarcadas no código, enquanto entidades dinâmicas (proprietários, animais e resenhas) são persistidas no banco, reduz a complexidade do modelo e facilita evoluções incrementais.
- **Robustez em diferentes telas:** o uso de coordenadas relativas e o recálculo de posicionamento das marcações conforme o tamanho renderizado da imagem contribuem para consistência visual em diferentes resoluções e densidades.
- **Evolução incremental e manutenção orientada a componentes:** a organização típica de aplicações React favorece manutenção do código e aprimoramento por iterações, permitindo incorporar futuras funcionalidades sem reescrita completa.

5.5 Conclusão e trabalhos futuros

Os resultados apresentados indicam que a proposta de informatização da resenha é viável e pode reduzir fragilidades do processo manual, especialmente quanto à perda de registros, à dificuldade de consulta e à variação na padronização das informações. No escopo do protótipo entregue, o objetivo geral foi alcançado ao disponibilizar um aplicativo Android com fluxo guiado e persistência em nuvem, além de incorporar o mapeamento por regiões zootécnicas e o registro de marcações de forma consistente.

Como limitações, destaca-se que a validação ocorreu majoritariamente sem um estudo formal com usuários em campo e sem métricas quantitativas (por exemplo, tempo médio de preenchimento, taxa de erros e satisfação). Adicionalmente, por depender principalmente de persistência em nuvem, o protótipo pode ser impactado por conectividade instável.

Como trabalhos futuros, recomendam-se:

- **Modo *offline* e sincronização:** na versão atual, o aplicativo depende da comunicação com o *Cloud Firestore* para registrar e recuperar os dados da resenha. Como evolução, recomenda-se a implementação de *cache* local no dispositivo Android, de modo a permitir a execução do fluxo de resenha mesmo sem conexão com a internet, realizando a sincronização dos registros quando o acesso à rede for restabelecido.
- **Relatórios e exportação:** geração de relatórios (por exemplo, em formato *PDF* (*Portable Document Format*)) com a descrição textual consolidada e anexos das marcações, facilitando compartilhamento e arquivamento.
- **Captura e anexação de imagens:** integração com câmera/galeria para anexar evidências fotográficas às marcações, reforçando rastreabilidade e permitindo a

criação de uma base de dados para treinamento de modelos para identificar cores e marcas nos animais.

- **Avaliação com usuários:** condução de estudo de uso com médicos-veterinários, comparando o processo informatizado com o processo manual em condições reais.
- **Evolução com apoio de visão computacional:** investigação de técnicas de reconhecimento de imagens como apoio à identificação de raça/pelagem e detecção de particularidades, como uma camada complementar ao preenchimento manual. Caso exista um módulo pronto para identificação automática de raça, pelagem e particularidades a partir de imagens, o *software* precisaria ser adaptado para incorporar uma etapa adicional de captura e envio de fotografias do animal, bem como uma interface para apresentação e validação, pelo usuário, das sugestões retornadas por esse módulo. Nesse cenário, a arquitetura da aplicação passaria a contemplar um fluxo complementar ao preenchimento manual, no qual os resultados da análise automática seriam utilizados como apoio à decisão, e não como substituição integral da conferência realizada pelo médico veterinário. Também seriam necessárias adequações na persistência dos dados, de modo a armazenar tanto a imagem utilizada quanto os resultados inferidos e sua eventual confirmação ou correção pelo usuário.

Por fim, o código-fonte do protótipo está disponível no repositório do *GitHub* *Projeto Equinos*.[1] No repositório constam instruções de execução e compilação do projeto (arquivo `README.md`), incluindo dependências e passos para rodar em ambiente de desenvolvimento e gerar o aplicativo para Android.

[1] Disponível em: <https://github.com/minacio00/projetoEquinos>. Acesso em: 2 de abril de 2026.

Referências

ALMEIDA. *Anemia infecciosa equina: prevalência equídeos de serviço em Minas Gerais*. [S.l.]: Arquivo Brasileiro de Medicina Veterinária e Zootecnia, 2006. Citado na página 10.

ALVES, W. P. *Desenvolvendo aplicações com Xamarin*. [S.l.]: Novatec, 2017. Citado na página 29.

BARBOSA, M. IdentificaÇÃo equina implementação do regulamento (ce) nº504/2008 de 6 de junho. 2011. Citado 2 vezes nas páginas 10 e 11.

BATISTA, L. M. N.; CARRIJO, W. C. *Informatização da resenha de equinos e muares*. Monografia (Trabalho de Conclusão de Curso (Bacharelado em Informática)) — Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Inhumas, 2016. Citado 20 vezes nas páginas 10, 12, 13, 14, 16, 23, 24, 26, 27, 28, 29, 34, 35, 37, 38, 39, 40, 42, 44 e 59.

BRASIL, M. da Agricultura do. *Manual de Legislação: Saúde Animal*. 2013. Citado na página 11.

Brasil. Ministério da Agricultura e Pecuária. *Programa Nacional de Sanidade dos Equídeos*. 2022. Disponível em: <https://www.gov.br/agricultura/pt-br/assuntos/sanidade-animal-e-vegetal/saude-animal/programas-de-saude-animal/sanidade-de-equideos>. Citado na página 10.

CARLSON. *Doenças dos sistemas hematopoiético e hemolinfático*. 2006. Citado na página 10.

DESHPANDE, R. *Build A Hybrid App Using Cordova + Angular*. 2021. <https://bityli.com/zmTnRn>. Acesso em: 10 de Mai. 2022. Citado 4 vezes nas páginas 31, 32, 41 e 53.

FIRESTORE, G. *Modelo de dados do Cloud Firestore*. 2022. <https://firebase.google.com/docs/firestore/data-model?hl=pt-br>. Acesso em: 06 de Jun. 2022. Citado 2 vezes nas páginas 29 e 41.

GOIÁS. Agência Goiana de Defesa Agropecuária. *Manual para Elaboração de Resenha de Equídeos com a Finalidade de Solicitação de Diagnóstico para Anemia Infecciosa Equina (AIE) e Mormo*. Goiânia, 2012. 41 p. Citado 12 vezes nas páginas 12, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25 e 26.

Jeff Stuherland. *Scrum : a arte de fazer o dobro do trabalho na metade do tempo*. [S.l.]: LeYa, 2014. Citado na página 27.

KANBAN. 2022. <https://en.wikipedia.org/wiki/Kanban_(development)#Managing_workflow>. Acesso em: 06 de Jun. 2022. Citado na página 27.

MAIA, V. C. *Identification of Quality Characteristics for Mobile Applications*. Dissertação (Dissertação (Mestrado em Engenharia de Sistemas e Computação)) — Universidade Federal do Rio de Janeiro (UFRJ), COPPE/Programa de Engenharia de Sistemas e Computação, Rio de Janeiro, 2019. Acesso em: 05 de Dez. 2025. Citado na página 29.

Ministério da Agricultura, P. S. Ministério da agricultura. manual de preenchimento para emissÃo de guia de trÂnsito animal de equÍdeos versão 19.0. 2016. Citado 3 vezes nas páginas 11, 13 e 16.

MORMO. 2022. <https://pt.wikipedia.org/wiki/Mormo>. Acesso em: 06 de Jun. 2022. Citado na página 10.

PRESSMAN, R. S.; MAXIM, B. R. *Software Engineering: A Practitioner's Approach*. 8. ed. [S.l.]: McGraw-Hill Education, 2014. Citado na página 28.

SOMMERVILLE, I. *Software Engineering*. 10. ed. [S.l.]: Pearson, 2016. Citado 2 vezes nas páginas 28 e 37.

World Wide Web Consortium (W3C). *Web Content Accessibility Guidelines (WCAG) 2.2*. 2024. <https://www.w3.org/TR/WCAG22/>. Acesso em: 31 de Mar. 2026. Citado na página 54.

World Wide Web Consortium (W3C). *Understanding Success Criterion 1.1.1: Non-text Content*. 2025. <https://www.w3.org/WAI/WCAG21/Understanding/non-text-content.html>. Acesso em: 31 de Mar. 2026. Citado na página 54.