INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
GOIANO - CAMPUS URUTAÍ
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

ADAMO GONÇALVES MAIA
WANDERSON FELIPE GONÇALVES PONTES

APLICATIVO PARA GERENCIAMENTO DE PRODUÇÃO DE LEITE

Urutaí
2023

ADAMO GONÇALVES MAIA
WANDERSON FELIPE GONÇALVES PONTES

APLICATIVO PARA GERENCIAMENTO DE PRODUÇÃO DE LEITE

Trabalho de Curso apresentado ao curso de Bacharelado em Sistemas de Informação do Instituto Federal Goiano – Campus Urutaí, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Dr. Júnio César de Lima

Urutaí
2023

AGRADECIMENTOS

Em primeiro lugar, agradecemos a Deus, que fez com que nossos objetivos fossem alcançados, durante todos os anos de estudos no instituto. Aos amigos/familiares, e ao professor Júnio César de Lima por sua orientação e ajuda, que muito contribuíram para a realização deste trabalho.

RESUMO

É inegável que a tecnologia está a cada dia tornando-se parte da rotina das pessoas. Além das pessoas, as empresas estão também aceitando que a tecnologia pode ajudar a ganhar mais e perder menos, integrando-a no ambiente de trabalho. Um desses setores é o de leite e derivados, onde usam ordenhadeiras para realizar a ordenha, simulando um bezerro se amamentando, fazendo a vaca aceitar a ordenha mais facilmente. Além da ordenhadeira, algo que grandes produtores de leite também usam são aplicativos para *smartphones* de controle dos dados do produto, fazendo estatísticas e ajudando o produtor nas tomadas de decisões. Mas ainda há pequenos produtores que não utilizam esses aplicativos, por conta da complexidade e diversas funcionalidades que apresentam, fazendo o pequeno produtor optar por ainda usar os cadernos de campo para armazenar os seus dados. O objetivo do projeto é a construção de um aplicativo de controle leiteiro com armazenamento de dados em nuvem, com interface simples, funcional e de fácil manuseio, no intuito de fazer os produtores deixarem de usar os cadernos de campo para usar um aplicativo que garanta mais segurança para os dados.

PALAVRAS-CHAVE: Android, Computação em Nuvem, Controle de dados, Leite e Derivados.

ABSTRACT

It is undeniable that technology, especially smartphones, is becoming part of people's routine every day. In addition to people, companies are also accepting that technology can help them earn more and lose less by integrating it into the work environment. One of these sectors is the dairy, where they use milking machines to perform the milking, simulating a calf being nursed, making the cow accept the milking more easily. In addition to the milking, something that large dairy producers also use are product data control applications, making statistics and helping the producer in decision making. But there are still small producers who do not use this application due to the complexity and different features it presents, making the small producer choose to continue writing down his data on paper. The objective of the project is the construction of a dairy control application with cloud data storage, with a simple, functional and easy-to-use interface, in order to make producers stop using notes on paper, to use an application that guarantees more data security.

KEYWORDS: Android, Cloud Computing, Data Control, Milk and Derivatives.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

APP Aplicativo

Embrapa Empresa Brasileira de Pesquisa Agropecuária

GCP *Google Cloud Platform*

IaaS *Infrastructure as a Service*

IBGE Instituto Brasileiro de Geografia e Estatística

IDE *Integrated Development Environment*

IFRO Instituto Federal de Rondônia

NoSQL *Not Only SQL*

PaaS *Platform as a Service*

SaaS *Software as a Service*

SO Sistema Operacional

UI *User Interface* (Interface de Usuário)

URI *Uniform Resource Identifier*

URL *Uniform Resource Locator*

UUID *Universally Unique Identifier*

FAEG *Federação da Agricultura e Pecuária de Goiás*

1 INTRODUÇÃO

A cada dia percebemos que a tecnologia está se tornando algo essencial e indispensável, não somente para tarefas do cotidiano, como também para a rotina dos agricultores. Esta é utilizada, principalmente, para auxiliar a produção, além de ajudar a minimizar os gastos e maximizar os ganhos do produtor.

A tecnologia é fundamental para o aumento da produção, por meio do aumento da produtividade, e para a gestão de todos os processos envolvidos com a produção. Segundo o IBGE, em 2006, é mostrado que a tecnologia foi a responsável por quase 70% do crescimento da produção de grãos, mostrando o quão importante e essencial é a utilização da tecnologia (EMBRAPA, 2017).

Carvalho et al. (2022 apud Statista, 2021; USDA, 2020) aponta que o Brasil tem sido classificado entre os cinco principais países produtores de leite, atingindo cerca de 23,5 milhões de toneladas métricas (MMT) em 2020. Para 2021, apesar das consequências econômicas adversas decorrentes da pandemia de COVID-19, prevê-se um aumento de 1,3% na produção em comparação com o ano anterior, uma vez que o setor de laticínios não foi tão significativamente impactado quanto outros setores de uma perspectiva global.

Além disso, o setor de produção leiteira no Brasil é uma das maiores geradoras de economia do país, sendo o sétimo maior dentre os produtos agropecuários nacionais e o terceiro maior produtor de leite mundial, perdendo apenas para Estados Unidos e Índia (ROCHA, 2020). Segundo o Ministério da Agricultura, Pecuária e Abastecimento, os produtores de leite são classificados em: agricultura familiar, pequeno, médio e grandes produtores.

O termo propriedade familiar é comumente utilizado para descrever propriedades de pequeno porte (Bassotto, et al. 2023 apud Uddin et al., 2021). Segundo Lima et al. (2021) 90% dos produtores são considerados pequenos, com baixo volume de produção diária, baixa produtividade por animal e pouco uso de tecnologias.

Shahsavari-Pour et al. (2023) afirma que técnicas de manejo adequadas melhoram o status da produção de leite. Além disso, citando uma pesquisa realizada por Khanal et al., onde é recomendado que, para maximizar a produtividade na indústria de produção de leite, é necessário usar tecnologias com uma abordagem sistemática integrada, por um lado, e

implementar uma técnica adequada, considerando o tamanho da fazenda, por outro lado. Uma dessas tecnologias é o sistema de ordenha automática, que é utilizado em relação às condições e ao meio ambiente.

Os dados sobre a produção de leite são fundamentais para o gerenciamento dos lucros e tomadas de decisões por parte do produtor, porém, muitos produtores, principalmente os de porte pequeno e familiar, usam o Caderno de Campo, ou seja, papel e caneta, para lidar com os dados da propriedade. Esse método dificulta o gerenciamento dos dados de produção, uma vez que pode trazer problemas, como a demora em consultas e manipulação dos dados, além de não garantir a segurança e confiança da informação.

Além disso, o caderno de campo que normalmente é utilizado pelos pequenos produtores, pode ser facilmente perdido, destruído ou ter suas informações modificadas, comprometendo os dados. Uma alternativa é o uso dos chamados aplicativos mobile, já que os smartphones se tornaram algo quase que indispensável para as pessoas no seu dia a dia, eles são a alternativa perfeita, uma vez que são de fácil acesso, estando sempre perto do produtor. Como ficam salvas em um banco de dados, as informações não são facilmente comprometidas ou corrompidas, além de permitir o acesso rápido e fácil ao produtor, sempre que este precisar.

Sendo assim, foi construído um aplicativo mobile que consiga ajudar os produtores a terem mais controle sobre suas produção de leite. O produtor têm acesso a relatórios da produção de leite, mantendo salvo os dados dos animais de sua propriedade, tendo mais controle sobre os mesmos, como quantidade de animais e produção por cada animal em específico.

Para a produção do aplicativo foi levantado os requisitos necessários, através de entrevistas que foram realizadas pela FAEG Jovem Urutaí com produtores e profissionais da área. Com todos os requisitos levantados, foi projetado as telas no aplicativo *Figma.* Escolhemos uma base de dados em nuvem *Firebase*, pois ele oferece vários benefícios tanto para os usuários como para os programadores, entre eles a possibilidade de usar o aplicativo quando estiver *offline,* algo que é bem provável pois o público está localizado na área rural e provavelmente não tem acesso quando vai para o campo. Com os requisitos e as telas projetadas para o aplicativo, iniciou-se o desenvolvimento do sistema.

Em suma, como os produtores precisam de um método seguro e prático de manusear os dados das propriedades, como trabalho de curso, temos como objetivo desenvolver um aplicativo mobile Android, que guarde os dados em em um banco de dados em nuvem, garantindo segurança e confiança nos dados, além de fácil mobilidade no campo. O aplicativo também conta com suporte offline, uma vez que a conexão com a internet no campo não é garantida.

A seguir é apresentada uma visão geral das Seções que compõem as partes deste Trabalho de Curso. A seção 2 discute a fundamentação teórica do trabalho, mostrando como é a indústria láctea e quão ela é importante, além de fazer uma revisão sobre o sistema Android que é a plataforma utilizada para o desenvolvimento do aplicativo e como funciona a Computação em Nuvem. A seção 3 descreve a metodologia utilizada do projeto, contendo os requisitos e diagramas do sistema, telas do aplicativo e uma parte de como foi a implementação. A seção 4 apresenta os resultados da construção do aplicativo. A seção 5 discorre sobre a conclusão do trabalho de curso e quais são os trabalhos futuros.

2 REVISÃO LITERÁRIA

Nesta seção é mostrado o resultado do estudo feito sobre o tema proposto. A princípio é falado sobre como é a produção leiteira no Brasil e no mundo, onde é destacado a importância do Brasil na produção leiteira. Após discorrer sobre a produção leiteira é mostrado como a tecnologia e os novos dispositivos podem ajudar os produtores. Com a sinopse sobre o mundo leiteiro discutido, foi mostrado como funciona a plataforma Android e como funciona alguns dos seus principais componentes. Ao final da seção é discutido como funciona uma computação em nuvem e a vantagem de utilizá-la para o armazenamento de dados.

2.1 Produção de leite no Brasil e no mundo

O agronegócio lácteo é um setor muito importante, pois como mostra Tabchoury (2022). ele é um dos setores que mais emprega no campo e na cidade, promove o melhor desenvolvimento, reduz a evasão e aumenta o aprendizado escolar das crianças, gera maior distribuição de renda, além de contribuir com a melhoria da nutrição e da saúde da população como um todo.

Além de que, o leite é um dos produtos mais vendidos e consumidos no Brasil e no mundo. Siqueira (2019) mostra que, em média, o equivalente a 116,5 kg de leite são consumidos por cada habitante por ano e essa quantidade tem aumentado a taxas de 1,2% ao ano. Maia et al. (2013) mostra dados do IBGE que demonstram que o Brasil saiu de uma produção de 7,1 bilhões de litros de leite em 1974, para 32,1 bilhões de litros de leite em 2011. Além disso, este produto também é uma fonte de renda para diversas pessoas ao redor do globo. Em relação a produção de leite, podemos afirmar que:

Aproximadamente 1 bilhão de pessoas no mundo depende do leite para sobreviver e 600 milhões de pessoas vivem em 133 milhões de fazendas leiteiras ao redor do mundo. Portanto, cerca de 10% da população mundial depende diretamente da produção leiteira (SIQUEIRA, 2019, apud GDP, 2017).

Além disso, como explica Tabchoury (2022), a produção de leite avançou 56% ao redor do mundo, atingindo 906 bilhões de litros, onde 51% desse total foram consumidos na sua forma fluída e derivados em geral, 20% em queijos, 18% em manteiga, 11% em leite em pó.

Oliveira (2022) mostra o aumento da quantidade de produção de leite de 12,1 para mais de 24,7 bilhões de litros ao ano. Mas houve queda na produção entre 2020 e 2021, como é mostrado na Figura 1, onde Oliveira (2022) explica que o ano de 2021 sofreu os efeitos da pandemia, com o aumento de dinheiro em circulação a partir de programas sociais aliado à diminuição de produção e desorganização das cadeias globais de suprimentos. Este cenário econômico adverso levou ao forte aumento do custo de produção de leite.

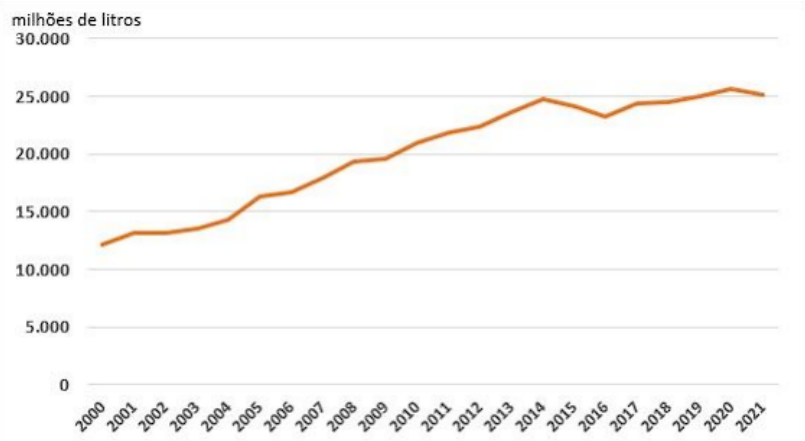


Figura 1: **Gráfico de produção de leite no Brasil entre 2000 e 2021**. Fonte: Oliveira (2022 apud IBGE, 2022)

O que faz os consumidores comprarem bastante leite, não somente para tomá-lo puro, mas para utilizá-lo em outros pratos da culinária, é por causa de “constituírem um grupo de alimentos de grande valor nutricional, uma vez que são fontes consideráveis de proteínas de alto valor biológico, além de conterem vitaminas e minerais.” (AMANCIO, 2015). Algumas vitaminas e minerais que podem constituir o leite são: cálcio, magnésio, selênio, riboflavina, vitamina B12 e vitamina B5.

Desse modo, com tantos minerais e vitaminas presentes, Amancio (2015) mostra quão importante esse produto pode ser na saúde da pessoa, conseguindo ajudar na saúde óssea e muscular, ajuda a prevenir algumas doenças como Diabetes, além de ajudar o gerenciamento de peso.

Segundo Leite (2020), no que se refere à exportação, a participação brasileira tem sido bastante acanhada, oportunista, sem estratégia definida, figurando como importador líquido. Para que o Brasil possa competir no cenário de exportação, além de focar no aumento da qualidade dos produtos, precisa trabalhar a questão da eficiência dos sistemas de produção de forma a obter preços competitivos da matéria prima. Uma das soluções para melhorar a eficiência do sistema de produção, é incentivar os produtores a investir em tecnologias que possam ajudar na produção, coleta e gestão dos dados.

2.2 Influência da tecnologia na produção de leite e derivados

A produção leiteira no Brasil é um dos principais agronegócios do país, estando entre os maiores produtores do mundo. “A cadeia produtiva do leite movimenta economias locais, é fonte de renda constante para milhares de pequenos produtores, gera impostos, fixa o homem no campo e é fonte geradora de postos de trabalho.” (PEREIRA, 2018)

Mas com tanta produção, houve também uma forte exigência do produto. Os produtores precisam investir em tecnologias, estudos e profissionais para conseguir vender seu produto em ótimo estado, já que o mercado consumidor e a competitividade estão maiores a cada dia.

Educapoint (2022) aponta que os números mostram que ainda existem muitas fazendas leiteiras com baixa produtividade e tecnificação, que poderiam melhorar muito seus

índices produtivos com a aplicação de tecnologias no campo, muitas delas não exigindo um investimento muito grande.

A agricultura da atualidade conta com vários instrumentos tecnológicos que auxiliam na produção e na gestão do produto. Viana et al. (2020) mostram que a agricultura moderna obriga cada vez mais a utilizar práticas que levem os agricultores a melhorar os seus custos de produção, evitar perdas e garantir uma produção.

No campo de produção leiteira, o instrumento tecnológico mais utilizado e mais investido entre os trabalhadores é a ordenhadeira mecânica, que segundo Alves (2020) é um equipamento mecânico que automatiza a ordenha manual. Desta forma, a máquina simula como se um pequeno bezerro estivesse se amamentando. Isso faz com que a vaca aceite a ordenhadeira mais facilmente.

Além da ordenhadeira, Educapoint (2022) cita algumas tecnologias usadas na indústria de lácteos, sendo elas:

- Sensores: Esses dispositivos coletam dados que são enviados para softwares, que processam as informações e elaboram relatórios, que, por sua vez, são usados para que se tomem melhores decisões na fazenda.
- Inseminação artificial: Utilizadas para que a fazenda mantenha sempre um rebanho de alto padrão genético. Essa tecnologia permite mudar o melhoramento genético, permitindo apenas que os melhores animais tenham sua genética multiplicada no rebanho.
- Softwares: Conseguir softwares nas áreas de produção, manejo, reprodução, nutrição, economia e finanças é essencial para o sucesso de qualquer fazenda, pois ajudam no planejamento e no controle da propriedade e dos animais.

Este instrumento traz aos produtores diversos benefícios como: agilização de ordenha de cada animal; economia na mão de obra; possibilidade de ordenha em vários animais simultâneos. Outro tipo de tecnologia muito utilizado principalmente por grandes produtores de leite, são os aplicativos de gestão, pois ajudam na organização, armazenamento e monitoramento dos dados do produtor.

A Embrapa e a IFRO desenvolveram um aplicativo chamado +Leite. Rocha (2020) faz uma citação de Pfeifer, pesquisador da Embrapa, mostrando que os dados gerados pelo app, podem auxiliar o produtor na tomada de decisão em relação ao manejo nutricional e

reprodutivo do rebanho, para que ele possa buscar a máxima eficiência na produtividade da propriedade leiteira. Isso significa maior produção de leite e mais renda.

2.3 Plataforma Android

Os telefones celulares são os dispositivos mais usados pelas pessoas no mundo inteiro. Muitos utilizam essa tecnologia como fonte de renda ou simplesmente para guardar suas coisas e ter acesso a elas sempre onde estiver, tornando estes dispositivos muito importantes para o cotidiano das pessoas.

Através dos anos o número de pessoas ao redor do globo que usam os smartphones vem crescendo a cada ano como mostrado na Figura 2, onde podemos ver que o uso de celulares aumentou em 50% entre 2002 e 2021.

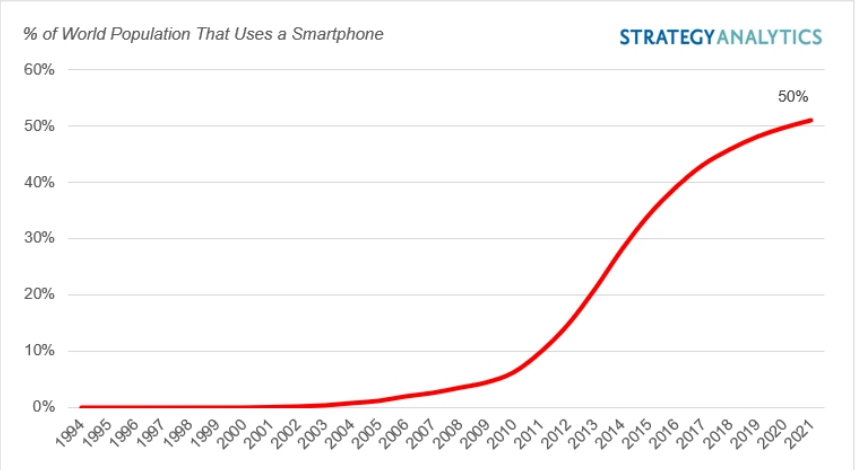


Figura 2: **Gráfico do uso de telefones celulares pela população mundial**. Fonte: Divulgação/ Strategy Analytics

Entre as plataformas mais utilizadas está a plataforma Android que "é uma plataforma para tecnologia móvel completa, envolvendo um pacote de programas para celulares, incluindo um sistema operacional, middleware, aplicações e interface do usuário" (Pereira e Da Silva, 2009). Esta plataforma possui muitas vantagens, quando os desenvolvedores optaram por escolhê-la. Entre estas vantagens, Abreu (2022) menciona:

- Código aberto: Como essa plataforma é baseada em Linux, ela possui uma licença aberta, permitindo que programadores possam contribuir para o desenvolvimento e melhoria da plataforma.
- "Interface mutante": que é a adaptação da plataforma para cada tipo de fabricante, ou seja, cada fabricante tem seu próprio tipo de interface android;

- Aplicativos e novidades: com o Android a possibilidade de adaptar novos aplicativos para o SO é muito maior por causa do fato de ser código aberto.
- Criação de aplicativos: é mais fácil e bastante prático para desenvolvedores que desenvolvem android, já que é possível utilizar um ambiente disponibilizado pela Google chamada Android Studio, que oferece ao desenvolvedor várias funcionalidades prontas, para otimizar o processo de desenvolvimento da aplicação.

Para o desenvolvimento do aplicativo Android, é preciso saber alguns conceitos sobre o ambiente. Como por exemplo as classes *Activity* e o *Service*, que são necessárias para a criação de um aplicativo. Mas cada uma dessas classes têm seu próprio ciclo de vida, com métodos que ajudam o desenvolvedor decidir, por exemplo, qual o melhor momento para executar determinado código, é preciso entender muito bem os ciclos de vida, pois se for implementada de maneira errada, algumas funções do seu código não funcionarão .

Além disso, o Android utiliza o objeto Intent para chamar outros componentes, como por exemplo *Activity* e *Service*. Outro arquivo que é necessário estar presente em todas aplicações Android é chamado de AndroidManifest.xml, ele é criado na raiz do sistema e contém informações essenciais desse sistema. Além do mais, a plataforma Android Studio é muito importante pois é a plataforma oficial para a criação de sistemas Android e consegue trazer muitas funcionalidades para o desenvolvimento. Todos esses assuntos serão mais aprofundados nas próximas seções.

2.3.1 *Activity*

Em um aplicativo Android, cada tela de interação com o usuário tem uma ou mais *Activity*. Cada *Activity* com seus próprios elementos UI (*User Interface*). Ableson, el al. (2012) mostra que a classe *activity* é parte do pacote Java android.app, encontrado no tempo de execução Android e o tempo de execução Android é implantado no arquivo android.jar.

A classe *Activity* possui uma interface de usuário composta por *Views*, consistindo de várias telas que respondem a eventos previamente programados. (Pereira e Da Silva, 2009). Uma *Activity* é, em outras palavras, uma tela que o usuário poderá ver e interagir com ela. Ela é a classe mais utilizada do sistema, já que cada tela visível e que o usuário possa interagir é uma *Activity*.

É comum que cada *Activity* tenha várias *Views*, que são subcomponentes da *Activity*. *Views* são o que seus usuários veem e aquilo com que eles interagem. *Views* lidam com o

Layout, fornecem elementos de texto para títulos e *feedback*, fornecem botões e formas para entradas do usuário e desenham imagens na tela do dispositivo. (Ableson, el al. 2012)

Android (2020) mostra que normalmente, uma atividade em um *app* é especificada como a main *activity*, ou a atividade principal do programa, que é a primeira tela a ser exibida quando o usuário inicia o app. Cada atividade pode iniciar outra para realizar ações diferentes. Por exemplo, ao entrar em um *app* de e-mail e a primeira tela é a que aparece os e-mails, essa é a main *activity*, ao clicar em um e-mail ou outro botão, é redirecionado para outras telas secundárias.

Uma das principais funções que uma *Activity* executa é exibir os elementos de UI, que são implementados como views e são geralmente definidos em arquivos de layout XML. (Ableson, el al. 2012).

2.3.1.1 Ciclo de vida de uma *activity*

Cada *Activity* tem seu próprio ciclo de vida, onde é possível programar cada uma delas, guardando informações para serem usadas na próxima vez que a *Activity* for usada. A representação do ciclo de vida de uma *Activity* é mostrada na Figura 3.

A classe *Activity* fornece uma quantidade de callbacks que permite que a atividade saiba sobre a mudança do estado: informa a respeito da criação, interrupção ou retomada de uma atividade ou da destruição do processo em que ela reside por parte do sistema (Android, 2020).

Figura 3: **Ciclo de Vida de uma Activity**. Fonte: Elaborada pelos autores

A classe *Activity* do Android fornece alguns métodos para navegar entre as fases do ciclo de vida como explica Android (2020), oferecendo controle sobre a *activity*, sendo elas: onCreate(), onStart(), onResume(), onPause(), onRestart(), onStop() e onDestroy().

- onCreate(): É ativado quando está havendo a criação do *activity*, ele somente será executado uma vez no ciclo do *activity* , e logo após será chamado o método onStart(). Ele também recebe o savedInstanceState, que é um objeto

Bundle, esse objeto é utilizado quando se quer passar parâmetros entre as *Activitys*. Além disso, é nesse método que é possível colocar lógicas básicas como instalar variáveis e vincular dados na lista. Após a conclusão do método a atividade é colocada em estado de "Criado".

- onStart(): Esse método é chamado logo após o término no onCreate(), tornando a *Activity* visível ao usuário, colocando o sistema no estado "Iniciado" e qualquer componente ciente do ciclo de vida que esteja ligado ao ciclo de vida da atividade receberá o evento onStart().
- onResume(): Nesse método a *Activity* é colocada no estado "Retomado", é nesse estado que o aplicativo irá interagir com o usuário. Esse estado é mantido até que algo faça a *Activity* perder o foco, como ligações ou mudança de *Activity* . Quando há essa perda de foco a *Activity* é colocada em estado de "Pausada", chamando o método onPause, e será retomada assim que o foco voltar para a *Activity*. Esse método diferente do onCreate pode ser chamado várias vezes no ciclo de vida.
- onPause(): Esse método é chamado assim que a *Activity* perde o foco, colocando em um estado de "Pausada". É bom lembrar para não utilizar esse método para guardar dados, ou fazer alterações no banco de dados, pois é possível que o comando não seja finalizado antes do término do onPause. Como as *Activitys* são usadas em forma de pilha, é bom usar esse método para quando a *Activity* sair do foco, liberar recursos e espaço na memória. Após esse método os próximos possíveis são o onResume, para quando a *activity* voltar a ter o foco ou o onStop para quando a *activity* ficar totalmente invisível para o usuário, colocando-a em estado de "Interrompida".
- onStop(): Ele é chamado assim que a *Activity* fica completamente invisível para o usuário, colocando-a em estado de "Interrompida". Assim que ele entra nesse estado o app começa a liberar alguns recursos que não precisavam mais ser guardados na memória. Esse é o momento de implementar funcionalidades como alterações no banco de dados, pois ele somente irá terminar assim que acabar os comandos. Após esse método o sistema pode chamar ou o onRestart para reiniciar a *Activity* ou o onDestroy para destruir a *Activity*.
- onRestart(): Após o término do onStop, o aplicativo pode chamar esse método para reiniciar a *Activity*, restaurando o estado em que ela estava antes de ficar no estado de "Interrompida", após isso chamando o método onStart.

- onDestroy(): É o estágio final do ciclo de vida de uma *Activity*, assim que o método onStop é finalizado é possível que o sistema chame esse método para destruir a *Activity* completamente. Pode ser que esse método seja chamado destruindo a *Activity* temporariamente, por exemplo quando a rotação de tela, quando isso acontece o sistema chama o onCreate automaticamente. Nesses casos onde é preciso guardar os dados da visualização antes de ser destruída é bom utilizar o ViewModel, para quando o sistema chamar o onCreate, as informações serem restauradas. Caso a *Activity* esteja sendo finalizada, o sistema liberará todos os recursos restantes que não foram finalizados anteriormente.

2.3.2 *Service*

Um *service* é um componente de aplicativo que representa o desejo de um aplicativo para executar uma operação de execução mais longa sem interagir com o usuário ou para fornecer funcionalidade para outros aplicativos usarem, como cita Android (2020). Como o *service* é feito para não interagir com o usuário, não têm uma UI e sua execução é feita no back-end do aplicativo.

Ableson, et al. (2012) explica que os *services* estendem uma classe Service do Android e também implementa uma interface chamada Runnable, para executar suas tarefas em uma *thread* separada. Uma service é mostrado em outras palavras por Android (2020), como um recurso para o aplicativo informar o sistema sobre algo que ele deseja fazer em segundo plano. Um *Service* tem um ciclo de vida diferente que uma *Activity*, como mostrado na Figura 4.

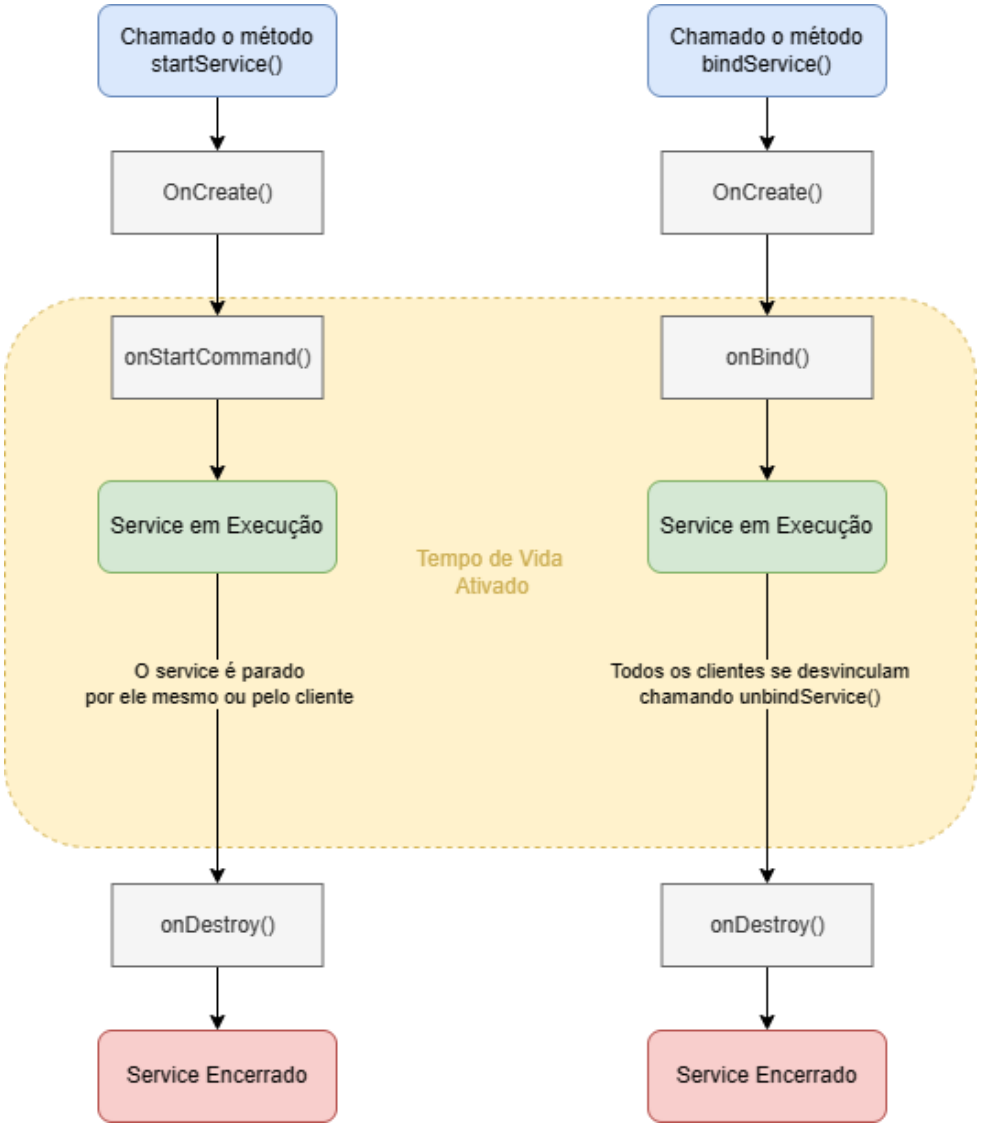


Figura 4: **Ciclo de vida de um *service***. Fonte: Elaborada pelos Autores.

O ciclo de vida de um *service* é composto por onCreate(), onBind(), onUnbind() e onDestroy() que é descrito por Android (2020) das seguintes maneiras:

- onCreate: esse método é chamado para criar o Service pela primeira vez e não é indicado chamá-lo diretamente.
- onBind: é retornado um canal de comunicação ao *Service*. Esse método recebe como parâmetro um Intent que será discutido na seção 2.3.4.
- onUnbind: esse método é chamado quando todos os clientes são desconectados da interface particular do *Service*.

- onDestroy: é chamado para informar ao aplicativo que o *Service* não será mais usado e pode ser destruído, limpando todos os recursos que estavam sendo utilizados em sua execução.

2.3.3 Intents

Activitys e *Services* são chamados através de mensagens assíncronas chamadas Intents (Pereira e Silva, 2009). Um objeto tipo Intent carrega as informações que um sistema Android precisa para determinar outro componente iniciar. (Android, 2020).

Valney (2018) explica que os *intents* são divididos em dois tipos: os explícitos e os implícitos. Os *intents* explícitos são quando é especificado qual o componente, por exemplo, quando se sabe o nome da classe da *Activity*, e deseja que o *intent* seja levado para essa *Activity*. Já os implícitos são quando não é definido o componente, por exemplo, quando deseja compartilhar algo, o android mostra uma lista de outros aplicativos que pode ser selecionado pelo usuário.

2.3.4 AndroidManifest.xml

Ao criar uma aplicação android o arquivo AndroidManifest.xml precisa estar entre os arquivos criados na raiz do projeto, pois ele descreve informações essenciais sobre o sistema. Nesse arquivo precisa haver: as *Activitys, Services e broadcasts* criados, o tipo de linguagem que o sistema irá utilizar (Java ou Kotlin).

Além disso, este arquivo é usado para quando o sistema precisar de alguma permissão, como utilização da internet, e também é possível declarar quais são os tipos de recursos de hardware ou software exigidos pelo app, pois se achar que o aplicativo for muito pesado para determinados dispositivos é bom limitar, para garantir uma boa experiência para o usuário.

2.3.5 Android Studio

A empresa Google desenvolveu sua própria IDE (Ambiente de Desenvolvimento Integrado), que segundo Harada (2019) é um programa de computador que reúne as características e ferramentas de apoio para a criação de aplicativos para dispositivos móveis para Android.

O Android Studio é a IDE oficial e também a mais utilizada para desenvolvimento Android. Segundo Android (2020) as principais vantagens de se utilizar essa IDE são:

- Um sistema de build flexível baseado em Gradle;
- Um emulador rápido com inúmeros recursos
- Um ambiente unificado que possibilita o desenvolvimento para todos os dispositivos Android
- A aplicação de mudanças para enviar mudanças de código e recursos ao aplicativo em execução sem reiniciar o app
- Modelos de código e integração com GitHub para ajudar a criar recursos comuns de apps e importar exemplos de código
- Frameworks e ferramentas de teste cheios de possibilidades
- Ferramentas de lint para detectar problemas de desempenho, usabilidade, compatibilidade com versões, entre outros
- Suporte a C++ e NDK
- Compatibilidade integrada com o Google Cloud Platform, facilitando a integração do Google Cloud Messaging e do App Engine.

Além dessas vantagens, o Android Studio consegue mostrar ao desenvolvedor uma visualização de como está ficando a aplicação na tela do dispositivo, além de facilitar o desenvolvimento, pois ele tem duas possibilidades de construções de tela, sendo elas o estilo de "arrastar e colar", onde é possível pegar componentes como por exemplos botões e posicionar na tela onde preferir, e o nativo desenvolvimento usando as linhas de código.

A IDE também ajuda o desenvolvedor, disponibilizando um simulador de um dispositivo Android para compilação, caso o desenvolvedor prefira é possível conectar o próprio dispositivo, caso tenha um dispositivo Android. Ela também tem suporte para gerenciar as dependências do aplicativo através do arquivo build.grandle, que é configurado para utilizar por padrão Maven Central Repository.

2.4 Computação em Nuvem

A *Cloud Computing* ou Computação em Nuvem significa "ser capaz de acessar arquivos, dados, programas e serviços de terceiros de um navegador da Web via Internet que são hospedados por um provedor terceirizado" (Kim et al. 2009 apud Hodson, 2008).

A ideia da *Cloud Computing* é conseguir acessar os dados salvos na nuvem em qualquer local, a qualquer hora, contanto que tenha acesso a internet. Tornando assim desnecessário salvar os arquivos em mídias físicas como computadores, celulares e pen-drives. Uma vez que os dados são guardados na nuvem, diminuem os riscos de serem perdidos ou destruídos, oferecendo mais segurança do que os dispositivos físicos.

Os tipos de implementações em nuvem são divididas em: Nuvem Privada, Nuvem Híbrida e Nuvem Pública. Já os modelos oferecidos pelos provedores de nuvem são: SaaS, PaaS e IaaS, como mostrado na Figura 5.

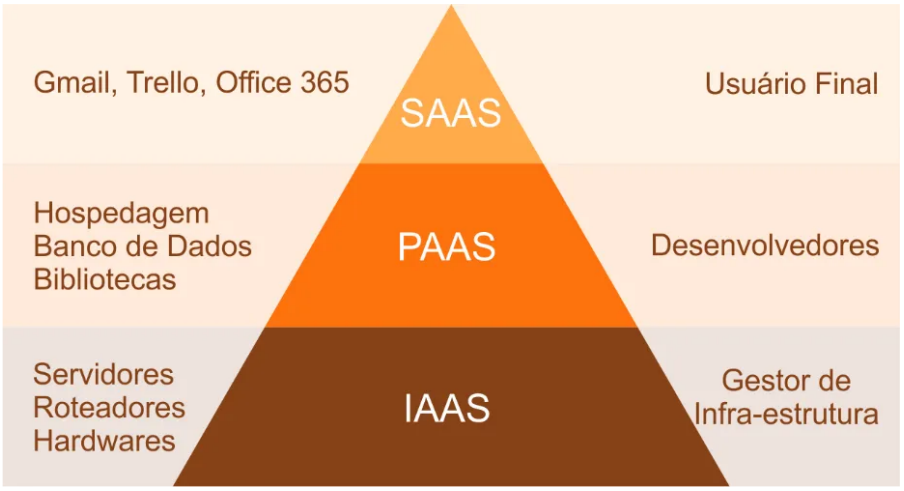


Figura 5: **Modelos de Arquitetura de serviços em nuvem**. Fonte: Alves apud GLEB B. (2017).

2.4.1 Nuvem Privada

De acordo com Sousa et al. (2009) "a infraestrutura de nuvem é utilizada exclusivamente para uma organização, sendo esta nuvem local ou remota e administrada pela". Em outras palavras, são nuvens normalmente utilizadas por uma organização ou empresa, por fins de segurança.

O mais comum é que os dados mantidos em nuvens privadas sejam dados de extremo sigilo, por isso na maioria das vezes as empresas utilizam a própria nuvem para manter esses dados. Pedrosa e Nogueira (2011) mostram que o gerenciamento dessa rede é feito pela própria empresa ou por terceiros, e quando for por terceiro a infraestrutura utilizada pertence ao usuário.

2.4.2 Nuvem Pública

Diferentemente da nuvem privada, “a infraestrutura de nuvens é disponibilizada para o público em geral, sendo acessada por qualquer usuário que conheça a localização do serviço” (Sousa et al. 2009). O mais comum de se utilizar esse tipo de nuvem é o público geral, já que não há restrição de gerenciamento desse tipo de rede.

Além disso, nesse tipo de rede, uma organização possui uma infraestrutura que disponibiliza esse serviço para o público utilizá-lo. Essa organização faz o gerenciamento dessa nuvem. “São serviços em nuvem fornecidos por terceiros (fornecedores). Elas existem além do firewall da empresa e são completamente hospedadas e gerenciadas pelo provedor da nuvem” (POSSOBOM, 2010 apud AMRHEIN, 2009).

2.4.2 Nuvem Híbrida

Nesse tipo de nuvem há uma junção dos tipos de nuvem pública e privada. A nuvem híbrida, segundo Pedrosa e Nogueira (2011), é caracterizada pela possibilidade da nuvem privada ter seus recursos ampliados pela reserva de recursos em uma nuvem pública. Isto permite manter os níveis de serviço mesmo no caso de flutuações rápidas na necessidade de recursos.

Uma nuvem híbrida bem construída poderia atender processos seguros críticos para uma missão, como o recebimento de pagamentos de clientes, assim como aqueles secundários para os negócios, como processamento de folha de pagamento de funcionários (POSSOBOM, 2010 apud AMRHEIN, 2009).

2.4.3 SaaS

O tipo de serviço em nuvem *SaaS* significa *Software as a Service* (*Software* como um serviço). Possobom (2010) mostra que trata-se de uma forma de trabalho onde o *software* é oferecido como serviço, assim, o usuário não precisa adquirir licenças de uso para instalação ou até mesmo comprar computadores ou servidores para executá-lo.

Nesse tipo de serviço em nuvem, é o usuário final que consome o serviço. Sousa et al. (2009) mostra que o usuário não administra ou controla a infraestrutura subjacente, incluindo rede, servidores, sistemas operacionais, armazenamento ou mesmo as características individuais da aplicação, exceto configurações específicas. Com isso, os desenvolvedores se

concentram em inovação e não na infraestrutura, levando ao desenvolvimento rápido de sistemas de *software*.

2.4.4 PaaS

O tipo de serviço em nuvem *PaaS* significa *Platform as a Service* (Plataforma como um Serviço). Essa camada é utilizada pelos desenvolvedores e segundo Pedrosa e Nogueira (2011) esse tipo de serviço é oferecido como serviço, um ambiente no qual o desenvolvedor pode criar e implementar aplicações sem ter que se preocupar em saber quantos processadores ou o quanto de memória está sendo usada para o executar a tarefa.

Para Alves (2019), a *PaaS* fornece um ambiente de desenvolvimento de aplicações, incluindo um sistema operacional e linguagens de programação que, de modo geral, é oferecido aos desenvolvedores, um ambiente de desenvolvimento escalável. No entanto, pode haver restrições sobre o tipo de aplicação que ele poderá desenvolver, devido a possíveis limitações que o ambiente do provedor de nuvem pode oferecer.

2.4.4 IaaS

O tipo de serviço em nuvem *IaaS* significa *Infrastructure as a Service* (Infraestrutura como um Serviço) e é o nível mais baixo da camada. “A idéia básica é que o usuário, em vez de adquirir e instalar servidores e equipamentos de rede em um *datacenter*, poderia usar estes recursos a partir de um provedor externo” (Possobom, 2010).

Conforme Pedrosa e Nogueira (2011), nesta classe são oferecidos os serviços de infraestrutura sob demanda, isto é, oferece recursos “de hardware” virtualizados como computação, armazenamento e comunicação. Este tipo de serviço prove servidores capazes de executar softwares customizados e operar em diferentes sistemas operacionais.

Nesse serviço o usuário não administra ou controla a infraestrutura da nuvem, mas tem controle sobre os sistemas operacionais, armazenamento e aplicativos implantados, e, eventualmente, seleciona componentes de rede, tais como *firewalls* (Sousa et al., 2009).

3 MATERIAIS E MÉTODOS

3.1 Levantamento de Requisitos

Os requisitos foram levantados pela FAEG Jovem Urutaí por meio de entrevistas com produtores da região. Em posse dos requisitos, teve início a primeira fase do projeto, onde os requisitos foram analisados. Estes requisitos foram classificados em Requisitos Funcionais e Requisitos Não-Funcionais, e são descritos nos tópicos seguintes.

3.1.1 Requisitos Funcionais

A Tabela 01 mostra os requisitos funcionais do sistema e seu vínculo com os Casos de Uso, descritos posteriormente na Seção 3.2.2.

Tabela 01. Requisitos Funcionais do sistema.

Requisito Funcional	**Caso de Uso Relacionado**
RF01: Criar uma conta	UCD 01
RF02: Acessar conta	UCD 02
RF03: Alterar dados da conta	UCD 03
RF04: Deletar conta	UCD 04
RF05: Cadastrar animal	UCD 05
RF06: Buscar animal	UCD 06
RF07: Alterar dados do animal	UCD 07
RF08: Deletar animal	UCD 08
RF09: Cadastrar produção	UCD 09
RF10: Buscar produção	UCD 10
RF11: Alterar dados da produção	UCD 11
RF12: Deletar produção	UCD 12
RF13: Gerar um relatório	UCD 13

3.1.2 Requisitos Não-Funcionais

A Tabela 02 apresenta os Requisitos Não-Funcionais do sistema.

Tabela 02. Requisitos Não-Funcionais do sistema.

Requisito Não-Funcional	**Descrição**
RNF01: Dispositivos Android	O sistema será executado em sistemas Android somente.
RNF02: Conexão com a internet	É necessário conexão com a internet para realizar o cadastro de uma conta no aplicativo e para a sincronização dos dados com a nuvem.
RNF03: Banco de dados	É necessário um banco de dados para a persistência dos dados do produtor e garantir a segurança e disponibilidade.
RNF04: Suporte offline	É necessário o suporte a operações offline, ou seja, sem conexão com a internet, como o cadastro de um animal e o registro de uma produção no campo.

3.2 Análise de Requisitos

3.2.1 Diagrama de Casos de Uso

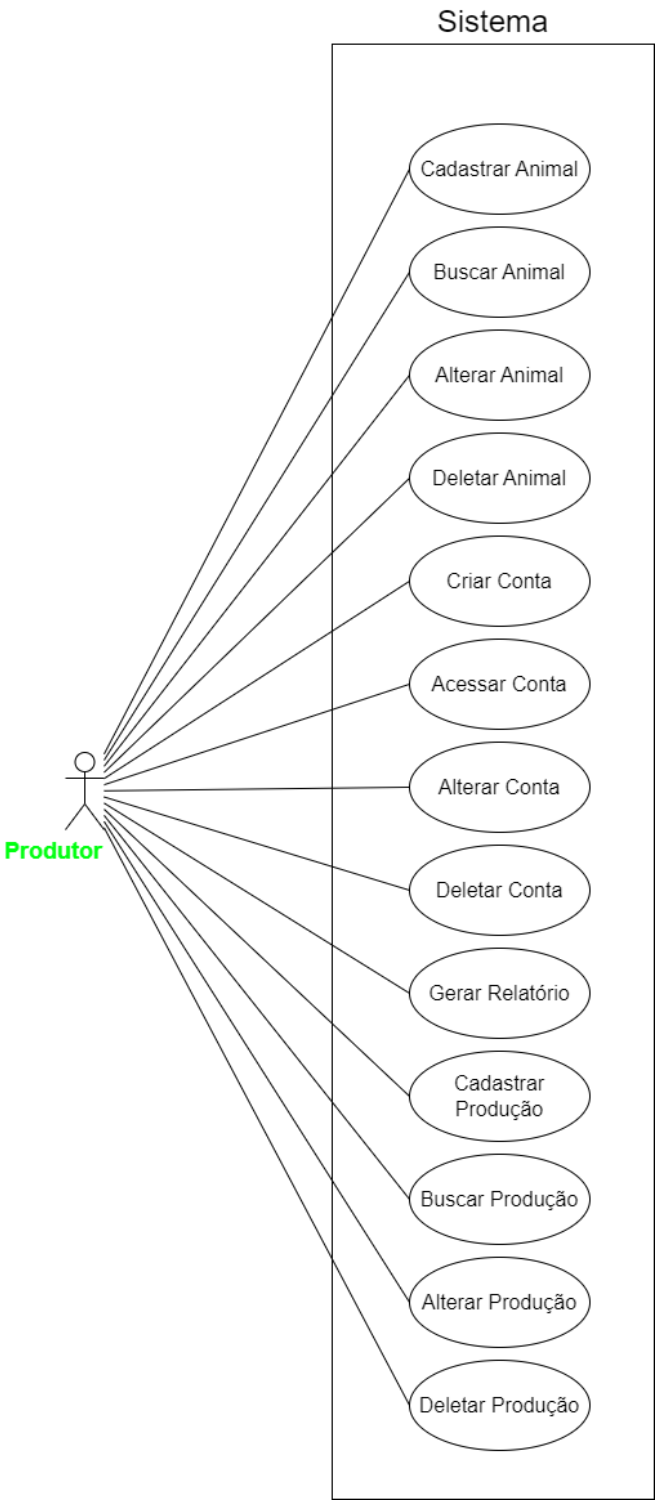


Figura 6. **Diagrama de Casos de Uso do sistema.** Fonte: elaborada pelos autores.

3.2.2 User Case Description (UCD)

Tabela 03: UCD 01 - Criar Conta

Nome do Caso de Uso	Criar Conta
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para criar uma conta no aplicativo.
Pré-condições	
Pós-condições	
Curso Normal	1. O Produtor acessa a tela de cadastro de usuários. 2. O sistema exibe o formulário para o Produtor preencher com seus dados de acesso. 3. O Produtor preenche o formulário e faz seu envio. 4. O sistema valida os dados. 5. O sistema faz o cadastro do Produtor e o redireciona para a tela *home*. 6. Finalizar caso de uso.
Curso alternativo 1	4.1 O sistema emite uma mensagem de erro. 4.2 Retornar para o passo 3.

Tabela 04: UCD 02 - Acessar Conta

Nome do Caso de Uso	Acessar Conta
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para acessar sua conta no aplicativo.
Pré-condições	
Pós-condições	
Curso Normal	1. O Produtor acessa a tela de *login* de usuários. 2. O sistema exibe o formulário para o Produtor preencher com seus dados de acesso.

	3. O Produtor preenche o formulário e faz seu envio. 4. O sistema valida os dados. 5. O sistema autentica o Produtor e o redireciona para a tela *home*. 6. Finalizar caso de uso.
Curso alternativo 1	4.1 O sistema emite uma mensagem de erro. 4.2 Retornar para o passo 3.

Tabela 05: UCD 03 - Alterar Conta

Nome do Caso de Uso	Alterar Conta
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para alterar os dados de sua conta no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela de perfil, por meio da tela *home*. 2. O sistema exibe o formulário para o Produtor preencher com os dados a serem alterados. 3. O Produtor preenche o formulário e faz seu envio. 4. O sistema valida os dados. 5. O sistema exibe uma mensagem de confirmação de alteração dos dados. 6. Finalizar caso de uso.
Curso alternativo 1	4.1 O sistema emite uma mensagem de erro. 4.2 Retornar para o passo 3.

Tabela 06: UCD 04 - Deletar Conta

Nome do Caso de Uso	Deletar Conta
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para deletar os dados de sua conta no aplicativo.

Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela de perfil, por meio da tela *home*. 2. O sistema exibe o botão para o Produtor pressionar e uma caixa de diálogo para o Produtor confirmar a exclusão de sua conta. 3. O sistema inicia a exclusão da conta do Produtor. 4. O sistema exibe uma mensagem de confirmação da exclusão da conta e redireciona o Produtor para a tela de *login*. 6. Finalizar caso de uso.
Curso alternativo 1	3.1 O sistema emite uma mensagem de erro e solicita que o Produtor tente novamente. 3.2 Retornar para o passo 2.

Tabela 07: UCD 05 - Cadastrar animal

Nome do Caso de Uso	Cadastrar animal
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para cadastrar um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista dos animais, por meio da tela *home*. 2. O Produtor acessa a tela de adicionar animais. 3. O sistema exibe o formulário para o Produtor preencher, com os dados do animal a ser cadastrado. 4. O Produtor preenche o formulário e faz seu envio. 5. O sistema valida os dados. 6. O sistema exibe uma mensagem de confirmação do cadastro do animal. 7. Finalizar caso de uso.
Curso alternativo 1	5.1 O sistema emite uma mensagem de erro. 5.2 Retornar para o passo 4.

Tabela 08: UCD 06 - Buscar animal

Nome do Caso de Uso	Buscar animal
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para buscar um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de animais, por meio da tela *home*. 2. O sistema exibe todos os animais do Produtor. 3. O Produtor seleciona o animal que busca. 4. Finalizar caso de uso.

Tabela 09: UCD 07 - Alterar Animal

Nome do Caso de Uso	Alterar Animal
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para alterar os dados de um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de animais, por meio da tela *home*. 2. O sistema exibe todos os animais do Produtor. 3. O Produtor seleciona o animal que deseja alterar os dados. 4. O sistema exibe o formulário com os dados do animal. 5. O Produtor altera os dados do formulário e faz seu envio. 6. O sistema valida os dados. 7. O sistema exibe uma mensagem de confirmação da alteração e redireciona o Produtor para a tela de lista de animais. 8. Finalizar caso de uso.

Curso alternativo 1	6.1 O sistema emite uma mensagem de erro. 6.2 Retornar para o passo 5.

Tabela 10: UCD 08 - Deletar animal

Nome do Caso de Uso	Deletar animal
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para deletar um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de animais, por meio da tela *home*. 2. O sistema exibe todos os animais do Produtor. 3. O Produtor acessa a tela de edição de animal, selecionando o animal que deseja deletar. 4. O sistema exibe o botão para deletar o animal. 5. O Produtor pressiona o botão e confirma a deleção por meio de uma caixa de diálogo. 6. O sistema inicia a exclusão do Animal. 7. O sistema exibe uma mensagem de confirmação da exclusão e redireciona o Produtor para a tela com a lista de animais. 8. Finalizar caso de uso.
Curso alternativo 1	6.1 O sistema emite uma mensagem de erro. 6.2 Retornar para o passo 5.

Tabela 11: UCD 09 - Registrar Produção

Nome do Caso de Uso	Registrar Produção
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para registrar uma produção de um animal no aplicativo.

Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de produções, por meio da tela *home*. 2. O Produtor acessa a tela de registro de produção. 3. O sistema exibe o formulário de uma nova produção. 4. O Produtor preenche o formulário e faz seu envio. 5. O sistema valida os dados. 6. O sistema exibe uma mensagem de confirmação do registro e redireciona o Produtor para a tela com a lista de produções. 7. Finalizar caso de uso.
Curso alternativo 1	5.1 O sistema emite uma mensagem de erro. 5.2 Retornar para o passo 4.

Tabela 12: UCD 10 - Buscar Produção

Nome do Caso de Uso	Buscar Produção
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para buscar uma produção de um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de produções, por meio da tela *home*. 2. O sistema exibe todas as Produções da conta. 3. O Produtor acessa a tela de edição de Produção, selecionando a produção que busca. 4. Finalizar caso de uso.

Tabela 13: UCD 11 - Alterar Produção

Nome do Caso de Uso	Alterar Produção
Caso de Uso Geral	
Ator Principal	Produtor

Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para alterar os dados de uma produção de um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de produções, por meio da tela *home*. 2. O sistema exibe todas as Produções da conta. 3. O Produtor acessa a tela de edição de Produção, selecionando a produção que deseja alterar. 4. O sistema exibe o formulário, com os dados da produção. 5. O Produtor altera os dados e envia o formulário. 6. O sistema valida os dados. 7. O sistema exibe uma mensagem de confirmação da alteração e redireciona o Produtor para a tela de com a lista de produções. 8. Finalizar caso de uso.
Curso alternativo 1	6.1 O sistema emite uma mensagem de erro. 6.2 Retornar para o passo 5.

Tabela 14: UCD 12 - Deletar Produção

Nome do Caso de Uso	Deletar Produção
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para deletar uma produção de um animal no aplicativo.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela com a lista de produções, por meio da tela *home*. 2. O sistema exibe todas as Produções da conta. 3. O Produtor acessa a tela de edição de Produção, selecionando a produção que deseja alterar. 4. O sistema exibe o botão de deletar produção. 5. O Produtor pressiona o botão e confirma a exclusão, por

	meio de uma caixa de diálogo. 6. O sistema inicia a exclusão da produção. 7. O sistema exibe uma mensagem de confirmação da exclusão e redireciona o Produtor para a tela com a lista de produções. 8. Finalizar caso de uso.
Curso alternativo 1	6.1 O sistema emite uma mensagem de erro. 6.2 Retornar para o passo 5.

Tabela 15: UCD 13- Gerar Relatório

Nome do Caso de Uso	Gerar Relatório
Caso de Uso Geral	
Ator Principal	Produtor
Atores Secundários	
Resumo	Este caso de uso descreve as etapas percorridas por um produtor para gerar um relatório da produção de seus animais.
Pré-condições	O Produtor precisa acessar sua conta no sistema.
Pós-condições	
Curso Normal	1. O Produtor acessa a tela de relatório, por meio da tela *home*. 2. O sistema exibe um formulário, com os filtros de intervalo de tempo das produções registradas. 3. O Produtor preenche e envia o formulário. 4. O sistema exibe um relatório das produções. 5. Finalizar caso de uso.

3.2.3 Modelo de Classes

Com os requisitos analisados, a modelagem das classes foi iniciada, e o Diagrama de Classes foi desenvolvido, como mostra a Figura 7.

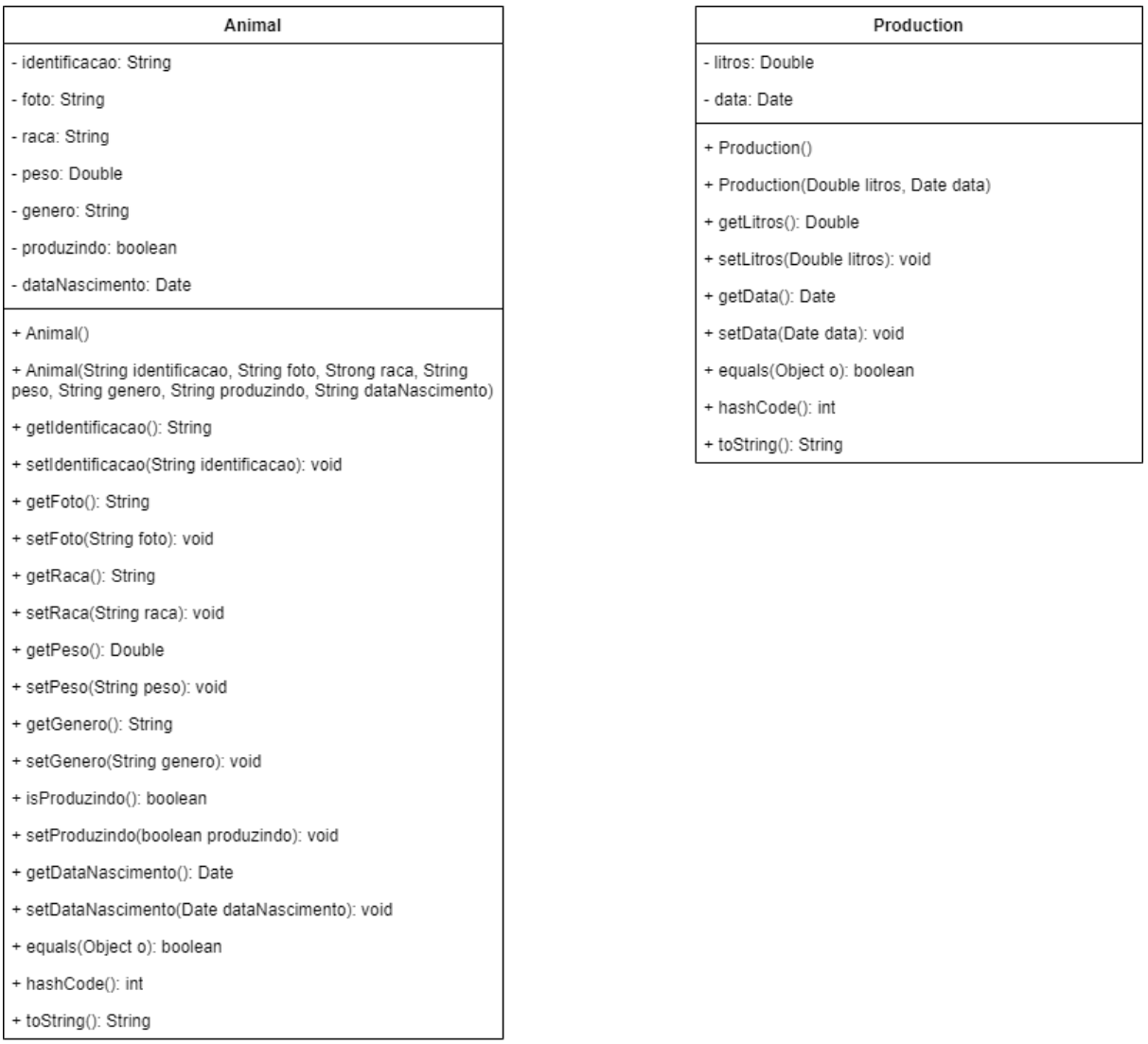


Figura 7. **Diagrama de Classes.** Fonte: elaborada pelos autores.

3.2 Projeto das Interfaces Gráficas

Após o levantamento e análise dos requisitos, teve início a segunda fase deste trabalho, a elaboração do projeto das interfaces gráficas e modelagem do banco de dados. O *design* das interfaces gráficas do projeto foi construído no software Figma, o qual é um editor gráfico de vetor e prototipagem de projetos de design, disponível na seguinte *URL: https://www.figma.com*.

Este software foi escolhido por permitir que vários desenvolvedores trabalhem juntos, e ao mesmo tempo, em um projeto, além possibilitar a configuração de uma pré-visualização da navegação entre as telas criadas. As telas projetadas deste projeto podem ser visualizadas

na seguinte *URL*: https://www.figma.com/file/j8aDHPcjWC3SZNZRv0pKLv/Telas. A Figura 8, apresenta algumas telas projetadas.

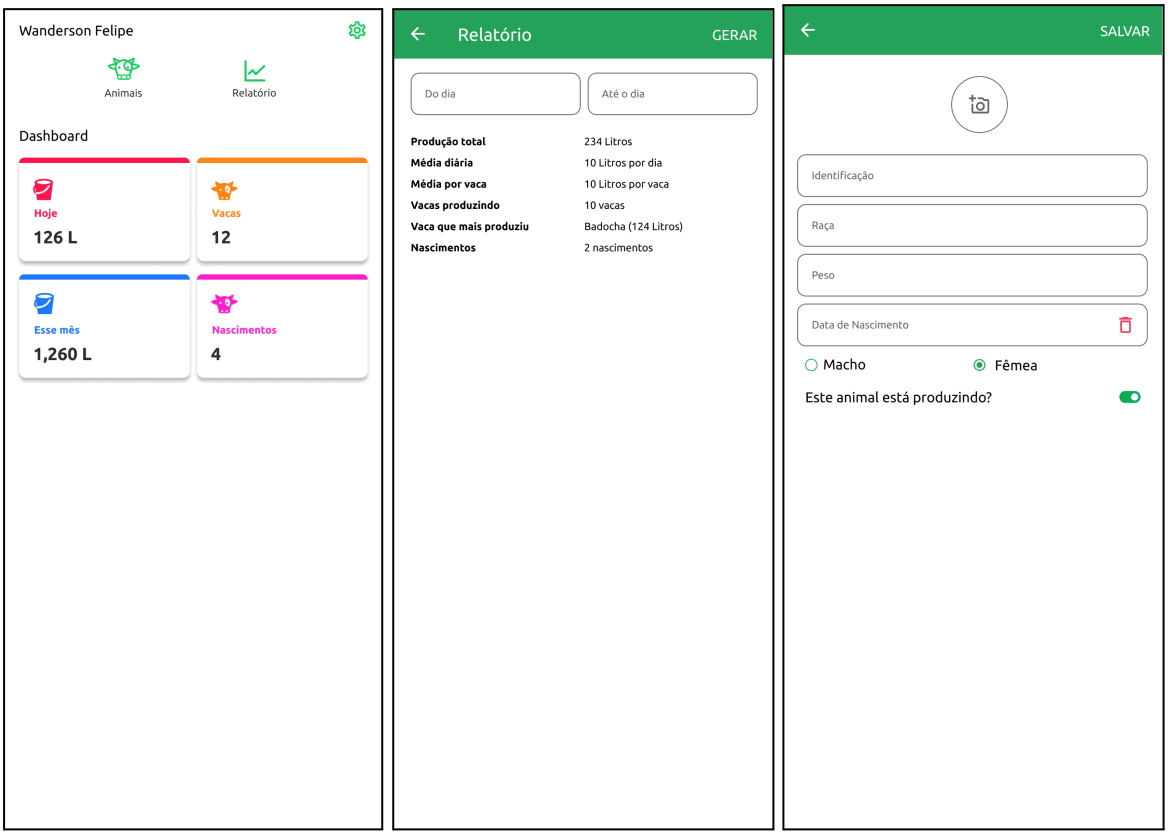


Figura 8. **Design das Telas de Home, Relatório e Cadastro de Animal.** Fonte: elaborada pelos autores.

3.3 Projeto do Banco de Dados

O Banco de Dados escolhido para o aplicativo foi o *Cloud Firestore*, o qual é um serviço em nuvem do *Firebase*. O *Firebase* é um conjunto de serviços de computação em nuvem para *back-end* e plataformas de desenvolvimento de aplicativos, fornecidos pelo Google. Como um produto da *Google Cloud Platform* (GCP), o *Firebase* permite hospedar bancos de dados e fornecer serviços de autenticação para uma variedade de aplicativos, incluindo Android, iOS, Javascript, Node.js, Java, PHP e C++.

O *Cloud Firestore* é um banco de dados *NoSQL* flexível e escalonável, para desenvolvimento focado em dispositivos móveis, que permite que os aplicativos o acessem diretamente, usando SDKs nativos, como os do Android, Node.js, Java, Python, C++ e Go, bem como em APIs REST e RPC.

O *Cloud Firestore* foi escolhido por oferecer suporte offline para dispositivos móveis, sendo possível criar aplicativos responsivos, que funcionem independentemente da latência da rede ou da conectividade com a Internet. Ele também possui um serviço de autenticação integrado, o que facilitou o desenvolvimento do aplicativo. Outro fator, que influenciou na escolha desse Banco de Dados, foi a fácil integração com a plataforma Android.

Segundo o Firebase (2022), o *Cloud Firestore* armazena dados em documentos que contêm mapeamentos de campos para valores. Esses documentos suportam muitos tipos de dados diferentes, como strings, números e objetos complexos e aninhados, e são armazenados em coleções, as quais são contêineres de documentos que podem ser usados para organizar dados e criar consultas.

Também é possível criar subcoleções dentro dos documentos e criar estruturas de dados hierárquicas, que podem ser escalonadas à medida que o banco de dados cresce. A Figura 9 mostra a estrutura hierárquica dos documentos e coleções do Banco de Dados elaborada para o sistema proposto.

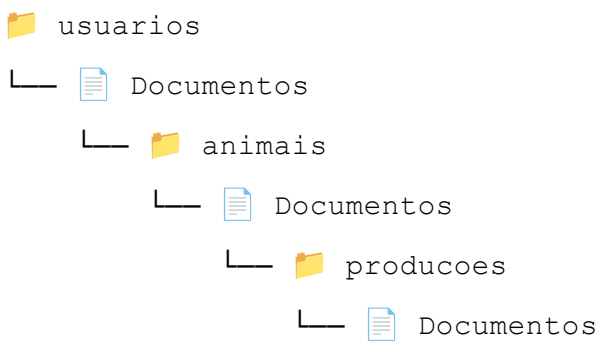


Figura 9. **Hierarquia dos documentos e coleções do Banco de Dados do sistema.** Fonte: elaborada pelos autores.

A coleção *usuarios*, possui os documentos com os dados dos Produtores cadastrados. Cada documento de um produtor tem uma coleção de animais e é estruturado da seguinte forma:

```
{
      created_at: timestamp
      animais: collection
}
```

A coleção *animais*, possui os documentos com os dados dos animais de um produtor. Cada documento de um animal tem uma coleção de produções, sendo estruturado da seguinte forma:

```
{
      identificacao: string
      foto: string
      raca: string
      peso: number
      genero: string
      produzindo: boolean
      data_nascimento: timestamp
      producoes: collection
}
```

A coleção *producoes*, possui os documentos com os dados das produções de um animal e é estruturado da seguinte forma:

```
{
      litros: number
      data: timestamp
}
```

3.3 Implementação do aplicativo

3.3.1 Construção do design das telas

Na construção das telas do aplicativo, foram utilizados vários *layouts* diferentes, algumas vezes combinados, como é o caso da tela de formulário de animal, onde foi utilizado um *ScrollView* que encapsula um *ConstraintLayout,* que por sua vez, possui um *LinearLayout.* A maioria dos elementos, como botões, imagens e botões de seleção, foram construídos usando os elementos nativos da plataforma Android. Alguns elementos, como o *DatePicker* e *TextInputLayout*, usados na tela de formulário de animal, são do pacote `com.google.android.material`, do *Material Design*.

No projeto, foi utilizada a biblioteca *Data Binding Library*, que é uma biblioteca de apoio, que permite vincular componentes de IU dos *layouts* a fontes de dados do app, usando um formato declarativo, em vez de programático. Para configurar o app para usar a vinculação de dados, adicionamos o elemento *dataBinding* ao arquivo `build.gradle`, no módulo do app:

```
android {
   ...
   dataBinding {
      enabled = true
   }
}
```

Após isso, convertemos o layout para um *layout data binding*, como mostrado no exemplo abaixo.

```
<layout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto">
    <data>
        <variable
            name="viewmodel"
            type="com.myapp.data.ViewModel" />
    </data>
    <ConstraintLayout... />
</layout>
```

E assim, podemos preparar o layout para ser utilizado em uma atividade:

```
private ActivitySignUpBinding binding;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivitySignUpBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
}
…
String name = binding.titName.getText().toString();
```

3.3.2 Criação de Usuários e Autenticação

Na criação e autenticação de usuários no sistema, foi utilizado o *Firebase Authentication*, que fornece serviços de back-end, SDKs e bibliotecas prontas para autenticar e criar usuários em aplicativos Android e outras plataformas. O *Firebase Authentication* oferece suporte à autenticação usando senhas, números de telefone, provedores de identidade federados conhecidos, como Google, Facebook, entre outros. O *Firebase Authentication* tem integração com outros serviços do *Firebase*, como o *Cloud Firestore*.

No sistema proposto, foi utilizado e-mail e senha, na criação e autenticação dos usuários. Assim, na criação dos usuários, obtemos uma instância do serviço, e posteriormente, chamamos o método que cria um usuário:

```
FirebaseAuth auth = FirebaseAuth.getInstance();
auth.createUserWithEmailAndPassword(email, password);
```

Esse método retorna uma tarefa (*Task*), com o resultado da autenticação (*AuthResult*), e dessa forma, podemos adicionar um *listener* que ficará encarregado de verificar se o tarefa foi concluída com sucesso ou não, e então podemos tomar ações dependendo do resultado:

```
auth.createUserWithEmailAndPassword(email,
password).addOnCompleteListener(task -> {
   if(task.isSuccessful()) {
      user.updateName(name);
      user.sendEmailVerification();
      user.createUserDocument();
   } else {
      // Tratamento da falha
   }
});
```

Se a criação do usuário ocorrer com sucesso, ele é autenticado automaticamente no sistema, podendo ser gerenciado após ser obtido a partir do objeto de autenticação:

```
FirebaseUser user = auth.getCurrentUser();
```

Dessa forma, enviamos um e-mail de verificação para o e-mail do usuário, usando o método `user.sendEmailVerification()`. Após o usuário verificar seu e-mail, ele terá acesso ao aplicativo. Verificamos se o email do usuário foi verificado com o método `user.isEmailVerified()`.

Na criação da conta, atualizamos o nome do usuário, usando o método `user.updateProfile(profileUpdate)`, onde *profileUpdate* é um objeto da classe *UserProfileChangeRequest*, no qual podemos usar o método `setDisplayName(String newName)`.

No processo de *login*, utilizamos o método `signInWithEmailAndPassword`, onde passamos o e-mail e senha do usuário. Nesse processo também é validado se o e-mail foi verificado. Na tela de login, o usuário pode solicitar a alteração de sua senha, caso tenha esquecido. Um e-mail de redefinição de senha será enviado para o e-mail do usuário, por meio do método `sendPasswordResetEmail(String email)`, da classe FirebaseAuth.

3.3.3 Banco de Dados

O *Firebase Authentication* gerencia o nome, e-mail e senha de um usuário. No processo de criação de conta no aplicativo, é criado um documento no Firestore, cujo ID é o

mesmo do identificador único do usuário, ou UID, cadastrado no *Firebase Authentication*. Criamos esse documento, pois, posteriormente, as coleções de animais e produções serão armazenadas nesse documento. Como não é possível criar um documento vazio, armazenamos nele o campo *created_at*, com a data em que a conta foi criada.

```
FirebaseFirestore db = FirebaseFirestore.getInstance();
db.collection("usuarios").document(user.getUid()).set(createdAt);
```

Esse documento permanece somente com esse campo até que o usuário verifique seu e-mail e consiga acesso ao sistema, ocasião na qual poderá cadastrar mais dados. Dessa forma, podemos armazenar os dados de cada usuário em um documento próprio, que por sua vez pode possuir coleções relacionadas. Na tela de verificação de e-mail, o usuário tem a opção de deletar a sua conta, onde será apagado seus dados do *Firebase Authentication* e do *Cloud Firestore*.

Uma vez que o usuário verificou seu e-mail, e obteve acesso ao aplicativo, ele pode cadastrar animais e produções. Os dados dos animais são armazenados em uma coleção chamada “animais”, que fica no documento com o mesmo UID do usuário. Usamos o método *set* para armazenar um objeto da classe Animal:

```
db
      .collection("usuarios")
      .document(userService.getUid())
      .collection("animais")
      .document()
      .set(animal);
```

Para atualizar esse documento, utilizamos o mesmo método acima, porém passamos o ID do documento na última chamada de *document*. Já para obter um documento em específico, usamos o código acima, porém informamos o ID do documento, como na atualização, e substituímos o método *set* por *get*.

Quando precisamos da lista de animais, usamos `db.collection("usuarios").document(user.getUid()).collection("animais").orderBy("identificacao").get()`, onde podemos ordenar pelo nome dos animais. Em alguns momentos precisamos de alguns filtros, como na geração de relatórios, e então adicionamos métodos *where*, como `whereEqualTo("produzindo", true)`.

Os dados das produções ficam armazenados em uma coleção chamada “producoes”, no documento de um animal. Dessa forma, cada produção está relacionada a um animal. Para acessar essa coleção, assim como seus documentos, foi usado: `db.collection("usuarios").document(user.getUid()).collection("animais").document(animalId).collection("producoes")`. Para exibir os dados vindo do Banco de Dados nas *RecyclerViews*, utilizamos o padrão Adapter, com a classe abstrata Adapter, da classe RecyclerView.

Para as operações sobre os dados, como escrita e leitura, usamos as regras de segurança do *Cloud Firestore*. Essas regras de segurança oferecem controle de acesso e validação de dados em um formato expressivo. Esse serviço foi usado em conjunto com o *Firebase Authentication*. As regras criadas permitem que apenas os usuários autenticados, que são donos dos documentos, possam fazer operações de escrita e leitura sobre os dados, como pode ser visto no esquema abaixo:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    function signedInAndOwner(userId) {
      return request.auth != null && request.auth.uid == userId;
    }
    match /usuarios/{userId} {
      allow create, read, update, delete, write: if
signedInAndOwner(userId);
    }
    match /usuarios/{userId}/animais/{animalId} {
      allow create, read, update, delete, write: if
signedInAndOwner(userId);
    }
    match
/usuarios/{userId}/animais/{animalId}/producoes/{productionId} {
      allow create, read, update, delete, write: if
signedInAndOwner(userId);
    }
  }
}
```

3.3.4 Gerenciamento de fotos de animais

Durante o cadastro e edição de um animal, o produtor pode escolher uma imagem do animal, que será salva na nuvem, no serviço *Cloud Storage*. O *Cloud Storage* para *Firebase* é um serviço de armazenamento de objetos. Usando os SDKs do *Firebase* para *Cloud Storage*,

é possível usar a segurança do Google para fazer upload e download de arquivos nos apps do Firebase.

As imagens dos animais são armazenadas em um *bucket* do Google *Cloud Storage* chamado “imagens”. O app gera um Identificador Único Universal (UUID), o qual será o nome da imagem no *bucket*. Se o upload ocorrer com sucesso, esse UUID é armazenado no campo foto, do documento do animal relacionado, para que possa ser recuperado posteriormente.

```
FirebaseStorage storage = FirebaseStorage.getInstance();
StorageReference
reference=storage.getReference().child("imagens");
StorageReference photo = reference.child(uuid);
photo.putFile(imageUri);
```

A imagem é recuperada da galeria do dispositivo do usuário, por meio de seu Identificador Uniforme de Recursos (URI), que é obtido a partir de uma Intent que obtém esse URI por meio do armazenamento interno do dispositivo:

```
Intent intent = new
Intent(Intent.ACTION_PICK,MediaStore.Images.Media.INTERNAL_CONTENT
_URI);
```

Quando iniciamos essa *Intent*, definimos um código que será verificado pelo aplicativo quando ele receber um resultado de uma Activity:

```
startActivityForResult(Intent.createChooser(intent, “title”),
CODE);
```

Verificamos o resultado dessa *Intent*, sobrescrevendo o método `onActivityResult` da Atividade onde a URI é recuperada. Nesse método, verificamos se foi obtido um resultado com sucesso, verificando se o `resultCode` é igual a `RESULT_OK`, e verificamos ainda se o código retornado é o mesmo que definimos na chamada da *Intent*. Após isso, temos a URI da imagem por meio da variável `data`:

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
@Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
   if (resultCode == RESULT_OK) {
      if (requestCode == GALLERY_PERMISSION_CODE) {
         if (data != null) {
```

```
            if (data.getData() != null)
                uri = data.getData();
        }
    }

}
```

Para exibição da imagem, usamos a biblioteca Glide, disponível em https://github.com/bumptech/glide. Glide é um carregador de imagens para Android que lida com decodificação, memória e cache. O uso dessa biblioteca é simple, como mostrado abaixo:

```
Glide
      .with(getBaseContext())
      .load(imageUri)
      .centerCrop()
      .into(binding.imageAnimal);
```

Dessa forma, quando a imagem precisa ser recuperada para ser exibida, é passado no método *load* seu caminho, que pode ser uma URL de uma imagem no *Cloud Storage*. Se o usuário desejar excluir sua conta do aplicativo, todos os dados do *Firestore*, do *Authentication* e do *Storage* serão apagados.

3.3.5 Telas Implementadas

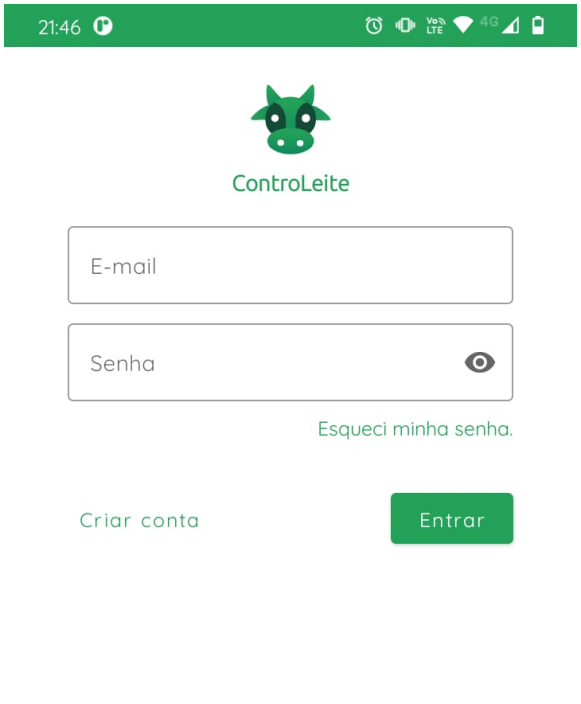


Figura 10: **Tela de Login.**

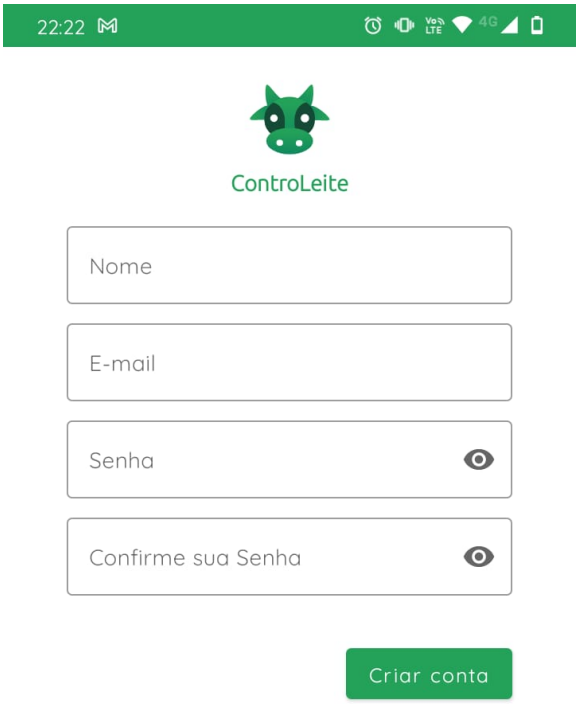
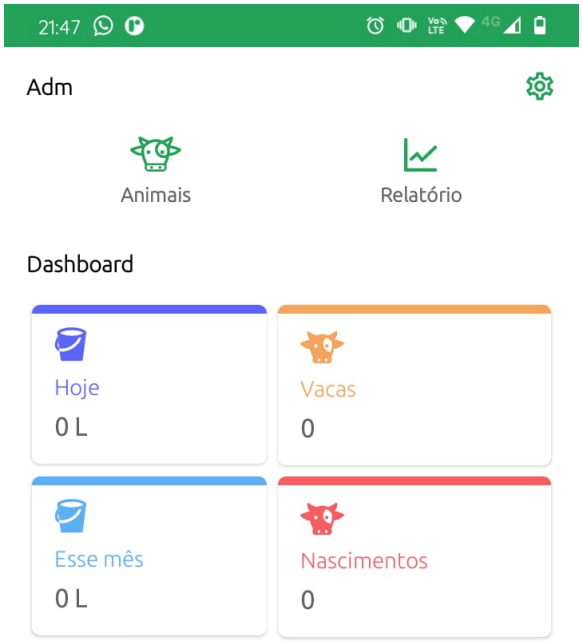


Figura 11: **Tela de Cadastro de Usuário.**

Na Figura 10, é apresentado a Tela de Login, onde o usuário insere seu e-mail e sua senha, para entrar no aplicativo. Nessa tela, também é possível acessar a tela de Cadastro de Usuários e solicitar uma redefinição de senha. Na Figura 11, é mostrada a Tela de Cadastro de Usuários, onde é possível criar uma conta no aplicativo.



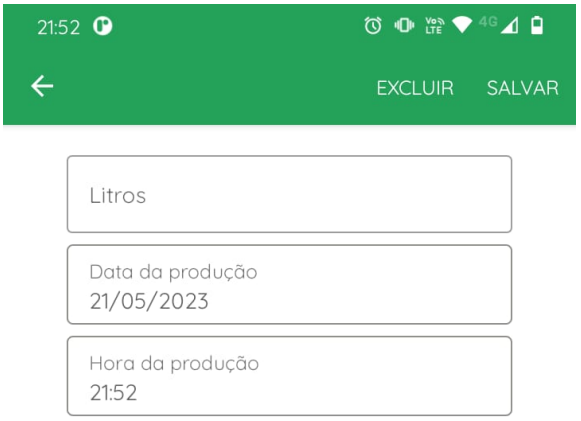


Figura 12. **Tela de Home**.

Figura 13. **Tela de Cadastro de Produção.**

Na Figura 12 temos a tela Home, onde é mostrado o menu principal, para acesso às telas de lista de animais, relatório e perfil. Também é exibido um *dashboard* com algumas informações sobre a produção e animais. A Figura 13 exibe a Tela de Cadastro de Produção, onde um produtor seleciona a data e hora, assim como a quantidade de litros produzidos.

Figura 14. **Tela de Lista de Animais.**

Figura 15. **Tela de Cadastro de Animal.**

A Figura 14 mostra a Tela de Lista de Animais, que exibe os animais em ordem alfabética, informando o nome e a foto do animal. Nessa tela também é possível acessar a Tela de Cadastro de Animal. A Figura 15 exibe a Tela de Cadastro de Animal, onde o usuário pode informar os dados do animal e selecionar uma foto.

Figura 16. **Tela de Relatório.**

Figura 17. **Tela de Relatório gerado.**

A Figura 16 mostra a Tela de Relatório, antes de ser gerado, onde a data pré-selecionada fica no intervalo referente a data de acesso à tela, até o primeiro dia do mês. Já a Figura 17 mostra um relatório gerado, com informações sobre produção e animais.

4 RESULTADOS E DISCUSSÕES

Neste trabalho, requisitos foram modelados, design de tela e banco de dados foram projetados e um aplicativo foi construído. Conseguimos desenvolver um aplicativo que pode ser usado pelos produtores para gerenciar a produção de leite de sua propriedade, assim como gerenciar os dados dos animais.

Por questão de tempo, não conseguimos realizar os testes com produtores. Com este trabalho, obtivemos aprendizado com o processo de análise e modelagem de requisitos, assim como experiência e aprendizado com desenvolvimento mobile. Também aprendemos características do ramo da agricultura e agropecuária.

5 CONCLUSÃO E TRABALHOS FUTUROS

Com este trabalho mostramos o quão importante é a produção leiteira no Brasil e no mundo, o cenário atual do Brasil nesse negócio e que podemos melhorá-lo investindo em tecnologia para ajudar os produtores. Invenções que ganharam espaço como ordenhadeira, chips de localização e inseminação mostram a melhora nos resultados.

Outra tecnologia que vêm sendo cada vez mais usada são os aplicativos para gestão do produto. Com base nisso e levando em conta que atualmente um *smartphone* é necessário e muito comum entre as pessoas, desenvolvemos um aplicativo que consiga ajudar produtores a terem mais controle sobre sua produção, cadastrando e visualizando animais e produções de leite, além de mostrar um relatório para o produtor.

Como em muitas propriedades familiares não há conexão com internet no campo, usamos o banco de dados em nuvem *Firebase,* que consegue fornecer um jeito de cadastrar dados offline e posteriormente, quando o *smartphone* estiver conectado, ele envia os dados para o banco de dados.

Conseguimos desenvolver um aplicativo que consegue arcar com os requisitos levantados e pode trazer vantagens para o gerenciamento da atividade leiteira, como gerar relatórios que ajudam os produtores a terem *insights* e tomarem decisões. Além disso, ele é bastante objetivo e prático em suas funcionalidades.

Como trabalho futuro planejamos realizar os testes em campo, analisar o *feedback* e sugestões dos usuários, se houver necessidade faremos alterações, adicionando ou removendo

funções para tornar sua experiência com o aplicativo ainda melhor. Em termos de monetização do aplicativo, a melhor opção levantada foi por assinatura.

REFERÊNCIAS

ABLESON, F., King, C., & Sen, R. (2012). Android em ação. Elsevier Brasil.

ABREU, Moises. Vantagens do sistema Android. 2022. Disponível em: <https://acadtec.com.br/blog/desenvolvimento-mobile/vantagens-do-sistema-android#:~:text=Por%20ser%20baseado%20no%20c%C3%B3digo,desenvolvimento%20e%20melhoria%20do%20sistema.>. Acesso em: 23/03/2023.

ALVES, Paulo Ricardo. Cloud Computing: cloud computing models. 2019. 34 folhas. Trabalho de Conclusão de Curso (Graduação em Sistemas de Informação) – Faculdade Pitágoras de Divinópolis, Divinópolis, 2019.

AMANCIO, Olga Maria Silverio. A IMPORTÂNCIA DO CONSUMO DE LEITE NO ATUAL CENÁRIO NUTRICIONAL BRASILEIRO. 2015. Disponível em: <http://sban.cloudpainel.com.br/source/SBAN_Importancia-do-consumo-de-leite.pdf>. Acesso em: 23/03/2023.

BASSOTTO, Leandro Carvalho, et al. Eficiência técnica em propriedades leiteiras familiares no Estado de Minas Gerais em 2021. Revista de Economia e Sociologia Rural, 2023, 62.

CANALRURAL. Destaque na produção mundial de leite, Brasil ainda tem dificuldade para exportar. 2021. Disponível em: <https://www.canalrural.com.br/noticias/leite-brasil-maiores-produtores-dificuldade-exportar/>. Acesso em: 20/04/2023.

CANALRURAL. Leite: mercado lácteo do Brasil busca recuperação em 2023. 2023. Disponível em: <https://www.canalrural.com.br/noticias/pecuaria/leite/no-brasil-mercado-lacteo-busca-recuperacao-em-2023/>. Acesso em: 20/04/2023.

Cloud Firestore. Firebase. 2022. Disponível em: <https://firebase.google.com/docs/firestore?hl=pt-br>. Acesso em: 23/03/2023.

COSTA, R. G. B. PINTO, C. L. O. ALBUQUERQUE, L. C. de. Leite e derivados: tecnologias, padrões de identidade e qualidade. Informe Agropecuário Belo Horizonte, n. 262, p. 1-84, 2013.

DA SILVA, Jerri Augusto; TSUKAMOTO, Ruth Youko. A modernização da pecuária leiteira e a exclusão do pequeno produtor. VICE-REITORA, p. 147, 2001.

DE CAMARGO, Artur Chinelato. Desafios da produção de leite. 2018.

DE LIMA, Fernanda Maximiano; DE OLIVEIRA GOMES, Lucas; MONTEIRO, João Vieira. IMPORTANCIA DA PECUÁRIA LEITEIRA NA AGRICULTURA FAMILIAR. In: Congresso de Tecnologia-Fatec Mococa. 2021.

EDUCAPOINT. Conheça as principais tecnologias utilizadas na cadeia produtiva do leite!. 2022. Disponível em: <https://www.educapoint.com.br/blog/pecuaria-leite/tecnologias-cadeia-produtiva-leite/#:~:text=Inova%C3%A7%C3%B5es%20e%20tecnologias%20na%20produ%C3%A7%C3%A3o,sua%20gen%C3%A9tica%20multiplicada%20no%20rebanho.>. Acesso em: 20/04/2023.

GOMES, Sebastião Teixeira. DIAGNÓSTICO E PERSPECTIVAS DA PRODUÇÃO DE LEITE NO BRASIL. 1999. Disponível em: <http://arquivo.ufv.br/der/docentes/stg/stg_artigos/Art_121%20-%20DIAGN%C3%93STICO%20E%20PERSPECTIVA%20DA%20PRODU%C3%87%C3%83O%20DE%20LEITE%20DO%20BRASIL%20(11-3-99).pdf >. Acesso em: 23/03/2023.

GONÇALVES, J. S. SOUZA, S. A. M. MODERNIZAÇÃO DA PRODUÇÃO AGROPECUÁRIA BRASILEIRA E O VELHO DILEMA DA SUPERAÇÃO DA AGRICULTURA ITINERANTE. SP, v.28, n.4, abr. 1998.

HARADA, Eduardo. O que é o Android Studio, ferramenta criada para desenvolver apps mobile. 2019. Disponível em: <https://www.tecmundo.com.br/software/146361-o-android-studio-ferramenta-criada-desenvolver-apps-mobile.htm>. Acesso em: 10/04/2023.

Impacto da tecnologia na produção de leite: avanços e produtividade. shopveterinario. 2021. Disponível em: <https://www.shopveterinario.com.br/blog/tecnologia-na-producao-de-leite/#:~:text=O%20uso%20de%20tecnologia%20na,animais%20e%20o%20seu%20conforto.>. Acesso em: 23/03/2023.

LAMAS, Fernando Mendes. Artigo: A tecnologia na agricultura. 2017. Disponível em: <https://www.embrapa.br/busca-de-noticias/-/noticia/30015917/artigo-a-tecnologia-na-agricultura>. Acesso em: 23/03/2023.

LEITE, JLB. O Comércio Internacional do Agronegócio do Leite. 2020.

LUIZ, Cristiane Rodrigues. A tecnologia no agronegócio. Fundação Educacional no município de ASSIS, 2013.

MACIEL, Alyne Martins, et al. Life cycle assessment of milk production system in Brazil: Environmental impact reduction linked with anaerobic treatment of dairy manure. Sustainable Energy Technologies and Assessments, 2022, 54: 102883.

MAIA, Guilherme Baptista da Silva et al. Produção leiteira no Brasil. BNDES Setorial, n. 37, mar. 2013, p. 371-398, 2013.

MATTOS, Tadeu Antonio. Pesquisa estima que metade da população mundial tem smartphones. 2021. Disponível em: <https://www.tecmundo.com.br/mercado/220009-pesquisa-estima-metade-populacao-mundial-tem-smartphones.htm>. Acesso em: 23/03/2023.

NETO, Eduardo Savarese. Computação em Nuvem: O que é, Como funciona e Importância. 2019. Disponível em: <https://fia.com.br/blog/computacao-em-nuvem/>. Acesso em: 23/03/2023.

OLIVEIRA, S. J. de M, et al. Produção de leite inspecionado no Brasil e Estados nos últimos 5 anos. 2022. Disponível em: <http://www.infoteca.cnptia.embrapa.br/infoteca/handle/doc/1144761>. Acesso em: 20/04/2023.

PEDROSA, Paulo HC; NOGUEIRA, Tiago. Computação em nuvem. Acesso em, v. 6, 2011.

PEREIRA, F.; MALAGOLLI, G. A. INOVAÇÕES TECNOLÓGICAS NA PRODUÇÃO DE LEITE. SIMTEC - Simpósio de Tecnologia da Fatec Taquaritinga, v. 4, n. 1, p. 11, 14 maio 2018.

PEREIRA, L. C. O., & da Silva, M. L. (2009). Android para desenvolvedores. Brasport.

PEREIRA, Lucio Camilo Oliva; DA SILVA, Michel Lourenço. Android para desenvolvedores. Brasport, 2009.

PORTO, M. A. G. BANDEIRA, A. A. A importância dos sistemas de informações gerenciais para as organizações. 2006. Disponível em: <https://simpep.feb.unesp.br/anais/anais_13/artigos/974.pdf>. Acesso em: 23/03/2023.

POSSOBOM, Camila Cerezer. Estudo de Caso: cloud computing-computação em nuvem. 2010.

ROCHA, Denis Teixeira de; CARVALHO, Glauco Rodrigues; RESENDE, João Cesar de. Cadeia produtiva do leite no Brasil: produção primária. Minas Gerais: Embrapa, 2020.

SEIBT, Uwe, et al. Development of an electronic interface for transfer of antimicrobial administration data in dairy farms. Plos one, 2022.

SHAHSAVARI-POUR, Nasser, et al. A System Dynamics Approach to Optimize Milk Production in an Industrial Ranch. Applied Sciences, 2023.

SILVA, Renata. Aplicativo auxilia no controle de desempenho do rebanho leiteiro. 2020. Disponível em: <https://www.embrapa.br/busca-de-noticias/-/noticia/50226113/aplicativo-auxilia-no-controle-de-desempenho-do-rebanho-leiteiro#:~:text=Chamado%20de%20%2BLeite%2C%20o%20software,forma%20r%C3%A1pida%2C%20simples%20e%20intuitiva.>. Acesso em: 23/03/2023.

SIQUEIRA, Kennya Beatriz. O mercado consumidor de leite e derivados. Circular Técnica Embrapa, v. 120, p. 1-17, 2019.

TABCHOURY, Wiliam. O MERCADO DE LEITE NO MUNDO. 2022. Disponível em: <https://www.adealq.org.br/blog/o-mercado-de-leite-no-mundo-2401>. Acesso em: 20/04/2023.

Uso de aplicativos na gestão de fazendas de leite. fundacaoroge. Disponível em: <https://www.fundacaoroge.org.br/blog/uso-de-aplicativos-na-gest%C3%A3o-de-fazendas-de-leite>. Acesso em: 23/03/2023.

VALNEY, Mário. Aula 10: O que são Intents (intenções)?. 2018. Disponível em: <https://fia.com.br/blog/computacao-em-nuvem/>. Acesso em: 10/04/2023.

VIANA, B. S. et al.A TECNOLOGIA COMO FERRAMENTA DE GESTÃO NA AGRICULTURA. 2020. Disponível em: <https://facunicamps.edu.br/cms/upload/repositorio_documentos/196_A%20TECNOLOGIA%20COMO%20FERRAMENTA%20DE%20GEST%C3%83O%20NA%20AGRICULTURA.pdf>. Acesso em: 23/03/2023.

VILELA, D. RESENDE, J. C. de. Artigo: Cenário para a produção de leite no Brasil na próxima década. 2014. Disponível em: <http://www.alice.cnptia.embrapa.br/alice/handle/doc/1019945>. Acesso em: 23/03/2023.

ZOCCAL, Rosangela. SOUZA, A. D. de. GOMES, A. T. Produção de leite na agricultura familiar. Juiz de Fora: Embrapa Gado de Leite, 2005. 20 p. (Embrapa Gado de Leite. Boletim de Pesquisa, 17).