Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Soluções Tecnológicas na Rastreabilidade de Adulterações por Formol no Leite

Thales Moreira Vinkler

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientadora
Profa. Dra. Celia Ghedini Ralha

Brasília
2018

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Soluções Tecnológicas na Rastreabilidade de Adulterações por Formol no Leite

Thales Moreira Vinkler

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Profa. Dra. Celia Ghedini Ralha (Orientadora)
CIC/UnB

Profa. Dra. Marcia Aguiar Ferreira
FAV/UnB

Dra. Vanessa Tavares Nunes
PPGInf/CIC/UnB

Prof. Dr. Rodrigo Bonifácio de Almeida
Coordenador do Bacharelado em Ciência da Computação

Brasília, 20 de fevereiro de 2018

Dedicatória

Dedico este trabalho à minha família, que sempre me apoiou e confiou em mim nas mais difíceis decisões, sem faltar em nenhum momento. Dedico também à todos os meus amigos e colegas, com os quais convivi durante este extenso período universitário.

Agradecimentos

Agradeço aos meus pais e irmã, que me apoiaram incondicionalmente neste trabalho e em todas as etapas da minha passagem pela UnB, que estão sempre ao meu lado em cada novo desafio que enfrento, acreditando sempre no meu sucesso.

À minha orientadora, Professora Doutora Célia Ghedini Ralha, que aceitou enfrentar comigo este novo desafio, contribuindo para a minha caminhada até aqui.

À Profa. Dra. Márcia de Aguiar Ferreira, da Faculdade de Agronomia e Medicina Veterinária, com toda a sua ajuda sobre a área de domínio do trabalho, e à Macofren Tecnologias Químicas, em nome de Renato Santana, que contribuiu com os experimentos práticos e com todo o estudo do reagente que possibilitou os resultados dos testes afim do leite.

Também agradeço aos amigos e colegas de curso que contribuíram para que este trabalho tivesse sucesso, à Empresa Júnior de Computação - CJR e ao Movimento Empresa Júnior - MEJ, que ajudaram no meu desenvolvimento tanto pessoal como profissional, e me mostraram a importância de estar sempre inconformado com a realidade em que vivemos.

Resumo

Neste trabalho descrevemos o projeto e a implementação de uma solução *web* e *mobile* para melhoria do controle de rastreabilidade e de adulteração por formol na cadeia produtiva de leite. Pretendeu-se construir uma solução que pudesse atuar em conjunto com um reagente químico que detecta presença de formol no leite por meio de teste colorimétrico, o FORMFIX®. A construção da solução passou pelas etapas de levantamento de requisitos, projeto da arquitetura da solução e codificação. Ao final, para validação da solução foram feitos testes de aceitação de funcionalidade com usuários finais. A solução construída ainda permite muitas evoluções, principalmente quanto a funcionalidades, como o acréscimo de técnicas de tratamento de imagens via *software* que consiga analisar a foto de um teste e apontar pela coloração do teste qual o resultado do mesmo.

Palavras-chave: Sistemas de Informação, Desenvolvimento *Web* e *Mobile*, Rastreabilidade, Adulteração no Leite, Cadeia Produtiva de Leite

Abstract

This study described the project and the development of a web and mobile solution to improve the tracebibity control and Formaldehyde adulteration control in milk supply chain. It intended to build a solution that worked together with a chemical reagent called FORMFIX®, that find out if there is Formaldehyde in milk with colorimetric tests. It had some steps as requirements gathering, software architecture and coding. At the end, final users made some acceptance tests with the purpose of validate the solution. The solution still needs more features, for example, some software intelligence that succeed in get the test result, automatically, just by analyzing the test's photo.

Keywords: Information Systems, Web and Mobile Development, Traceability, Milk Adulteration, Milk Supply Chain

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

BPMI Business Process Management Initiative.

BPMN Business Process Modeling Notation.

DRY Don't Repeat Yourself.

MER Modelo Entidade Relacionamento.

MTV Model-Template-View.

MVC Model-View-Controller.

OMG Object Management Group.

PYPL PopularitY of Programming Language.

SI Sistemas de Informação.

UML Unified Modeling Language.

UP Unified Process.

Capítulo 1

Introdução

O crescimento da produção de leite no Brasil, nos últimos dois anos, foi considerável, aproximando-se dos 4% ao ano [3]. Segundo [4], esses números são gerados por cerca de 23 milhões de vacas ordenhadas, 1,3 milhão de produtores e 2 mil laticínios com inspeção.

O Brasil detém uma grande produção de leite, estando na 4ª posição entre os maiores produtores mundiais. Essa produção de leite, em 2016, chegou à 35,3 bilhões de litros [4].

1.1 Problema

Apesar do crescimento da produção leiteira e de grandes avanços tecnológicos que as indústrias têm vivenciado, ainda são recorrentes as autuações de produtos, adulterações, problemas de qualidade e recolhimento de lotes. Há algum tempo esses tipos de acontecimentos tem se repetido e com isso várias investigações começaram a ser feitas.

Tal fato foi evidenciado pela investigação da Polícia Federal e Ministério Público, articulados pelo Ministério da Agricultura (MAPA), chamada de "Leite Compen$ado" [5], que apontou um esquema de adulteração de leite bovino no Rio Grande do Sul. Indícios mostraram que foram adulterados cerca de 100 milhões de litros da substância.

Além das adulterações, os laticínios também estão vulneráveis à contaminação por fontes biológicas, químicas e físicas. Devido a um rigoroso processo de controle de qualidade do leite, feito na chegada dos caminhões aos laticínios, uma quantidade desta matéria prima pode acabar sendo descartada devido às contaminações. Por isso a identificação e eliminação destas fontes de contaminação são de grande interesse para esta indústria.

Um problema ainda maior a ser considerado para a saúde humana pode ser causado pelo consumo do leite adulterado por formol. Desde julho de 2004 a Agência Internacional para Pesquisa em Câncer - IARC[6] classificou o formol como um produto carcinogênico, tumorogênico e teratogênico.

1.2 Questão de Pesquisa

Diante dos problemas existentes no cenário exposto, várias sugestões e hipóteses acabam surgindo como forma de resolução das dificuldades enfrentadas pelos laticínios. Em um mundo cada vez mais digital, muitas dessas indagações acabam por utilizar um viés tecnológico, relacionado à utilização de dispositivos móveis, sistemas e aplicativos.

Pretende-se investigar neste trabalho se o uso de tecnologias Mobile e Web podem melhorar a rastreabilidade de adulterações por formol no leite.

1.3 Objetivos

Já atuando no ramo de laticínios com foco na qualidade do leite, existe a Macofren, uma empresa de tecnologia química, incubada em 2016 no CDT – Centro de Desenvolvimento Tecnológico da Universidade de Brasília.

A Macofren possui um produto chamado FORMFIX, que consiste em um kit para desenvolvimento de testes rápidos para análise da qualidade do leite. O kit possui um reagente, que em contato com uma amostra de leite, a deixa com uma coloração específica que aponta, ou não, uma quantidade ilícita de formol na amostra.

Para complementar o kit do FORMFIX, vislumbrou-se a necessidade do desenvolvimento de uma aplicação que auxiliasse no registro virtual dos testes realizados e também no registro dos locais de coletas/entregas na qual estes testes foram feitos.

Desta forma, a solução deve abranger tanto um aplicativo mobile quanto um sistema web. Espera-se que os mesmos sejam capazes de proporcionar mais segurança ao controle das amostras e testes, a partir de uma foto retirada no aplicativo. Além disso, também é esperado poder registrar os locais de coleta/entrega de cada um dos testes realizados pelos caminhões dos laticínios.

Essas informações contribuem para a otimização da coleta e da distribuição de leite e derivados, assim como na redução de custos com ineficiências, perdas e desperdícios na cadeia.

Desta forma, o objetivo geral deste trabalho é construir uma solução que permita a rastreabilidade no transporte do leite, bem como aumentar a segurança da informação mantendo a veracidade dos resultados dos testes com o FORMFIX.

Como objetivos específicos, apresentamos:

- Especificar o sistema através do uso de modelagem orientada a objetos;
- Implementar a solução (aplicativo mobile e sistema web);
- Testar o uso da solução em ambiente real.

1.4 Metodologia

A metodologia utilizada neste trabalho pode ser classificada como pesquisa aplicada, uma vez que o trabalho propõe o uso de soluções tecnológicas que auxiliam na rastreabilidade de adulterações por formol no leite. Mais especificamente, o trabalho aborda uma ferramenta para a rastreabilidade do leite e para registro de testes colorimétricos que indicam presença de formol no leite.

Foram realizadas entrevistas de coleta de requisitos e análise de documentos seguida pelo desenvolvimento de um artefato de software para avaliar a questão de pesquisa levantada, ou seja, o trabalho é uma pesquisa exploratória, com abordagem qualitativa.

O processo utilizado nesse trabalho inclui quatro fases:

1. Definição do problema e levantamento de dados com especialistas em leite e em testes colorimétricos

   Antes do desenvolvimento da solução é necessário primeiro um estudo da cadeia produtiva de leite, bem como seus atores e algumas características próprias advindas da natureza dos laticínios. Após esse estudo foi realizado o "Levantamento de Requisitos", um momento no qual se entende quais serão as funcionalidades pretendidas pelo usuário da aplicação e quais as regras de negócio que a aplicação precisará respeitar.

2. Projeto da solução

   Nessa fase é feita à modelagem da solução, tomando como base as características do problema e também as funcionalidades e regras levantadas na fase anterior. A modelagem passa por várias pequenas etapas de construção de modelos e diagramas. Essas construções nada mais são que abstrações de padrões que serão seguidos para o desenvolvimento da solução.

3. Desenvolvimento

   Após feita a modelagem da solução, inicia-se a codificação, onde tudo que foi modelado será construído. Para construção são utilizadas linguagens de programação, servidores para alocação de código e bancos de dados para armazenamento de informações.

4. Testes de validação da solução

   Com a implementação realizada são feitos testes das funcionalidades da mesma com usuários finais de forma a validar os requisitos levantados.

O restante desta monografia apresenta no Capítulo 2 uma revisão teórica dos fundamentos aplicados. No Capítulo 3 será apresentada a proposta de solução, enquanto no

Capítulo 4 a validação da mesma por meio de testes. As conclusões e trabalhos futuros são apresentados no Capítulo 5.

Capítulo 2

Referencial Teórico

Nesse capítulo será possível equalizar conhecimentos sobre o que são Sistemas de Informação (SI) e o seu relacionamento com pessoas e negócios dentro de uma organização. Além disso é importante também observarmos como esses SI são modelados, documentados e, principalmente, desenvolvidos. Serão mostrados também todas as ferramentas utilizadas para as etapas supracitadas.

Será explanado ainda sobre o domínio da aplicação, o Leite. Serão abordados assuntos, como, sua cadeia produtiva, fraudes e adulterações comuns e sobre o produto FORMFIX® [7].

2.1 Sistemas de Informação

O uso dos SI tem crescido cada vez mais no mundo dos negócios. Sua implementação é de grande valia para organizações, pois gera melhorias na excelência operacional, criação de novos produtos e serviços e, principalmente, melhoria da tomada de decisão.

Nessa visão, os SI são cada vez mais cruciais para organizações pois acabam envolvendo três importantes dimensões através da visão sociotécnica: 'Organizacional', 'Humana' e 'Tecnológica' [8].

A dimensão organizacional leva em consideração a estrutura hierárquica, a cultura, os objetivos e os processos organizacionais. Nessa dimensão se verifica como essas características organizacionais se relacionam com os SI existentes, e em muitas vezes como estão embutidos nos mesmos.

A dimensão humana se refere à pessoas. Pessoas essas que utilizam os SI para alcançarem os objetivos da organização. Assim essa dimensão avalia a capacidade do ser humano de resolver problemas organizacionais transformando os SI em soluções.

A dimensão tecnológica aborda todos os aspectos da infraestrutura de TI da organização: hardware (equipamento físico), software (conjunto de programas/instruções que

coordenam os hardwares), tecnologias de armazenagem de dados e tecnologias de comunicação e de redes. Essa dimensão permite entender como as pessoas coordenam e utilizam essas tecnologias, permitindo assim a existência dos SI e a utilização dos mesmos por outros funcionários.

Baseado nessas três dimensões, [8] definem tecnicamente um SI como um conjunto de componentes inter-relacionados. Esses componentes absorvem, armazenam, processam e disponibilizam informações com o objetivo de facilitar e apoiar três atividades:

- tomada de decisões;
- coordenação;
- controle de uma organização.

Os SI ainda auxiliam, na perspectiva humana, trabalhadores, gerentes e coordenadores a encontrar problemas na operação e vislumbrar novos produtos e soluções.

2.2 Modelagem de Sistemas

Nessa seção apresentamos brevemente conceitos relacionados à linguagens, ferramentas e técnicas que auxiliam a modelar graficamente a representação dos SI a serem desenvolvidos, bem como a especificar o processo de negócio relacionado.

2.2.1 BPMN

A modelagem BPMN - *Business Process Modeling Notation*, é uma notação criada pela *Business Process Management Initiative* (BPMI), atualmente mantida pelo *Object Management Group* [9].

Segundo a OMG, o BPMN é uma notação gráfica padrão utilizada para modelagem de processos de negócios por meio de diagramas, assim como o UML é a notação padrão utilizada para modelagem de sistemas Orientados a Objetos.

O BPMN é uma notação gráfica que descreve a lógica passo a passo do processo, para facilitar assim o entendimento do mesmo. Assim como também ajuda na visualização de gaps, problemas e gargalos nos processos.

O BPMN contém quatro grupos de entidades:

- **Objetos de Fluxo (*Flow Objects*)**

  São os elementos que definem o comportamento do processo de negócio. Existem 3 tipos:

1. Eventos (*Events*)

   Algo que acontece ou pode acontecer em um processo. Como mostra a Figura 2.4, podem ser 'Eventos de Início', 'Eventos Intermediário' ou 'Eventos de Fim'.

Figura 2.1: Eventos do BPMN.

2. Atividades (*Activities*)

   Representa trabalhos/passos executados dentro do processo. Como mostra a Figura 2.5, pode ser uma 'Tarefa' ou um 'Sub-processo'.

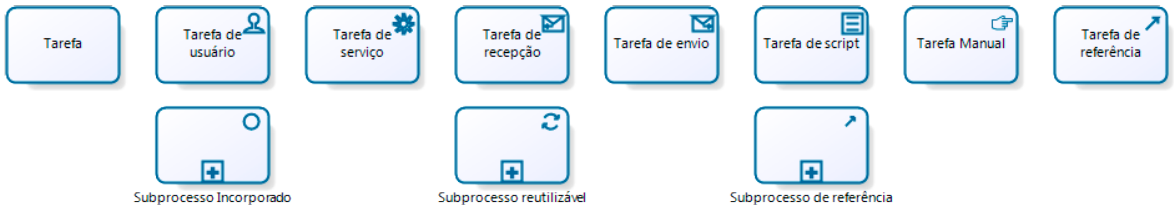

Figura 2.2: Atividades do BPMN.

3. Decisões (*Gateways*)

   Responsáveis por controlar o fluxo de sequência, determinando decisões, como juntar ou dividir um trajeto. Como mostra a Figura 2.6, existem seis tipos desses elementos.

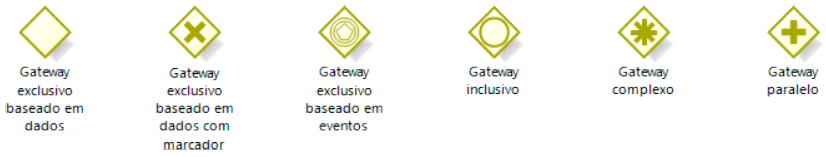

Figura 2.3: Decisões do BPMN.

- **Objetos de Conexão (*Connecting Objects*)**

  Representam a forma como os objetos de fluxo se conectam. Como mostra a Figura 2.7, existem três tipos:

Figura 2.4: Objetos de Conexão do BPMN.

1. Fluxo de Sequência

   Mostra a ordem do fluxo, mostrando de onde o fluxo sai e para onde ele vai.

2. Fluxo de Mensagem

   Mostra o fluxo de mensagens, saindo do emissor e chegando no receptor.

3. Associação

   Utilizado para associar dados, artefatos ou textos aos objetos de fluxo.

- **Raia de piscina (*Swimlanes*)**

  Agem como um contêiner para os objetos de fluxo, sendo assim uma forma de organização das atividades em categorias visuais separadas. Como mostra a Figura 2.8, existem a 'Piscina' e a 'Raia':

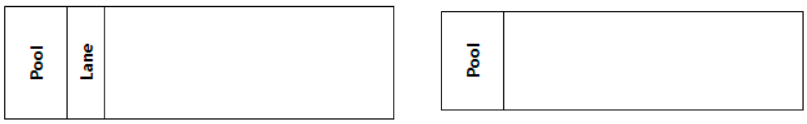

Figura 2.5: Raia de Piscina do BPMN.

1. Piscina (*Pool*)

   São utilizadas normalmente para separar duas entidades do negócio (ou participantes) que estão no mesmo diagrama, separando as atividades que cada um desenvolve.

2. Raia (*Lane*)

   São as subdivisões de uma piscina. Utilizadas normalmente para separar as atividades de acordo com papéis/funções existentes no processo. A raia seria, por exemplo, um departamento de uma organização e a organização seria a piscina.

- **Artefatos (*Artifacts*)**

  Utilizados para colocar informações adicionais no processo e também para representar as entradas ou saídas de uma atividade. Como mostra a Figura 2.9, existem dois tipos de artefatos:

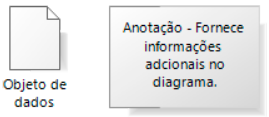

Figura 2.6: Artefatos de Conexão do BPMN.

1. Objetos de Dados

   São elementos que uma atividade requer como entrada ou produz como saída, conectados por associações.

2. Anotações

   São informações adicionais sobre uma atividade, colocadas para facilitar o seu entendimento.

2.2.2 UML

UML significa *Unified Modeling Language* (Linguagem de Modelagem Unificada). A UML é uma linguagem visual que permite a modelagem de sistemas computacionais [10]. Devido à sua grande utilização, a UML tornou-se internacionalmente uma linguagem padrão de modelagem de sistemas.

Vale ressaltar que a UML não é uma linguagem de programação e sim uma linguagem de modelagem que ajuda principalmente no entendimento das características do software. Entre essas características, a UML permite a definição de comportamentos, requisitos, estruturas lógicas, pré-requisitos físicos e também processos que existam no software.

Segundo [10], a UML compreende vários tipos de diagramas divididos em três famílias:

1. Diagramas Estruturais

   - Diagrama de Pacotes
   - Diagrama de Classes
   - Diagrama de Estrutura Composta
   - Diagrama de Distribuição
   - Diagrama de Componentes

2. Diagramas Comportamentais

    - Diagrama de Casos de Uso
    - Diagrama de Atividades
    - Diagrama de Máquina de Estados

3. Diagramas de Interação

    - Diagrama de Comunicação
    - Diagrama de Sequência
    - Diagrama de Tempo
    - Diagrama de Visão Geral de Integração

Wazlawick [11], ainda reforça que embora cada diagrama tenha sua funcionalidade, nem sempre se faz necessária a utilização de todos os diagramas para modelagem de um sistema. São utilizados apenas aqueles que realmente acrescente algum valor para o processo.

Neste trabalho, como foco na simplificação da solução e criação de uma versão beta, teremos mais foco na utilização dos diagramas de caso de uso, de classes e de pacotes.

Diagrama de casos de Uso: Segundo [12], todo sistema interage com alguns atores, humanos ou não, que se utilizam do sistema para executar tarefas. O diagrama de caso de uso, nesse contexto, especifica o comportamento de um sistema (ou parte dele) na forma de uma sequência de passos, a fim de se chegar à conclusão de uma tarefa por um ator.

Assim, os casos de uso fornecem para os desenvolvedores uma maneira de entender qual tarefa determinado ator poderá executar no sistema e também como ele faz isso e qual a sequencia de passos ele realiza.

Nos diagramas de caso de uso existem alguns artefatos importantes:

- Atores: são os usuários do sistema ou tipos de usuários.
- Caso de uso: ação/funcionalidade/tarefa realizada por um usuário.
- Comunicação: objeto que interconecta um ator a um caso de uso.

Tendo como base o exemplo de diagrama de caso de uso da Figura 2.1, é possível identificar dois atores, ‘Cliente’ e ‘Gerente da Agência’.

Segundo o diagrama, existem quatro casos de uso, ou quatro ações possíveis no sistema, que algum ator poderá realizar: ‘Consultar Saldo’, ‘Depositar Dinheiro’, ‘Sacar dinheiro’ e ‘Abrir Conta’.

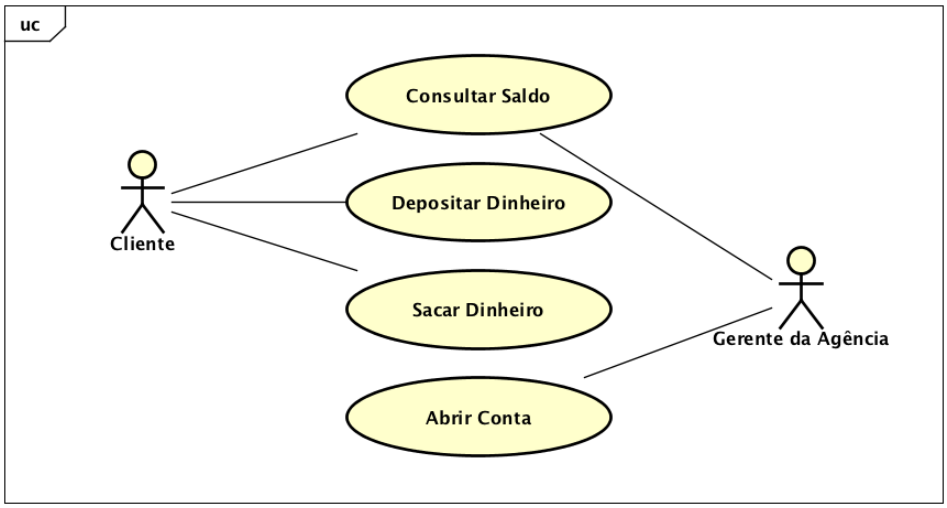

Figura 2.7: Exemplo de Caso de Uso.

Entretanto, nem todos os atores podem realizar todas as ações disponíveis, por isso a necessidade de saber qual a relação do ator com o caso de uso por meio da comunicação. No exemplo, é possível visualizar que:

- 'Cliente' pode consultar saldo da sua conta
- 'Cliente' pode depositar dinheiro na sua conta
- 'Cliente' pode sacar dinheiro da sua conta
- 'Gerente da Agência' consegue consultar saldo de alguma conta
- 'Gerente da Agência' pode abrir uma nova conta

Diagrama de Classes: Segundo [12], é um diagrama que mostra um conjunto de classes, colaborações e interfaces. Além disso também mostra os relacionamentos existentes entre eles. Visualmente, um diagrama de classes é um conjunto de vértices e arcos que os conectam.

Nos diagramas de classes existem alguns artefatos importantes:

- Classe: é a descrição de um conjunto de objetos que compartilham características semelhantes, ou seja, mesmos atributos, operações, relacionamentos e semântica.
- Interface: é um conjunto de operações que especificam um serviço de uma classe
- Relacionamento de dependência, generalização e associação: é uma conexão entre itens.

Tendo como base o exemplo de diagrama de classes da Figura 2.2, é possível identificar três classes, 'Cliente', 'Pedido' e 'Item de Pedido'.

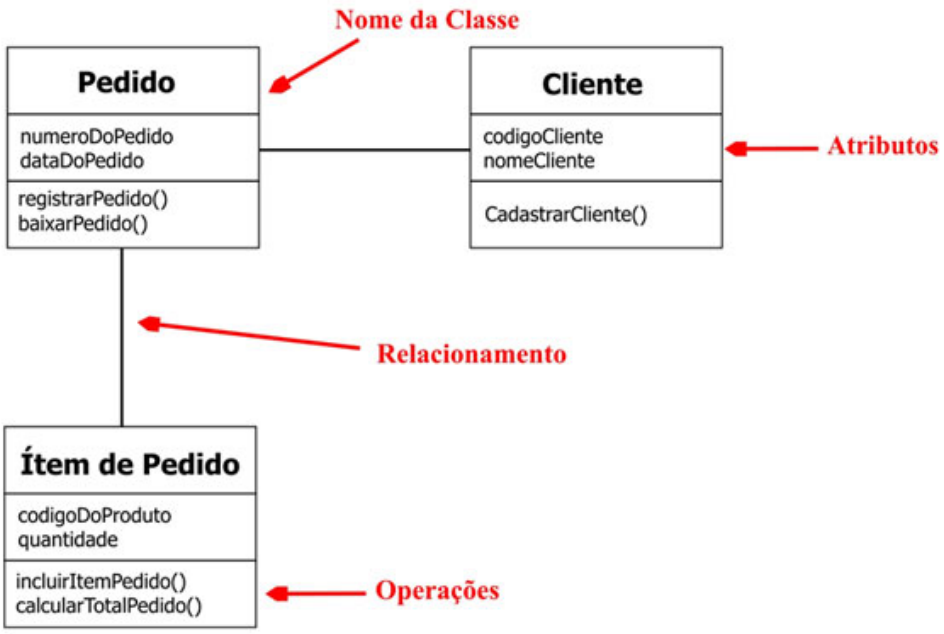

Figura 2.8: Exemplo de Caso de Uso.

Segundo o diagrama, todas as classes possuem atributos e operações, por exemplo, o 'Cliente' possui os atributos 'codigoCliente' e 'nomeCliente' e a operação 'CadastrarCliente'.

Também existem no diagrama dois relacionamentos, entre 'Cliente' e 'Pedido' e entre 'Pedido' e 'Item de Pedido', que podem ser entendidos como:

- 'Cliente' registra um 'Pedido';
- 'Pedido' recebe a inclusão de um 'Item de Pedido'.

Diagrama de Pacotes: Segundo [12], é um diagrama que descreve os módulos do sistema divididos em agrupamentos e mostrando as dependências entre eles. Expõe muitas vezes a hierarquia da solução e do seu modelo de arquitetura.

Nos diagramas de pacotes existem alguns artefatos importantes:

- Pacote: é um mecanismo para organização do modelo de uma arquitetura. Normalmente representada graficamente por uma pasta com uma guia.

Tendo como base o exemplo de diagrama de pacotes da Figura 2.3, é possível identificar quatro módulos, 'Login', 'Relatórios', 'Reembolsos' e 'Contribuições'.

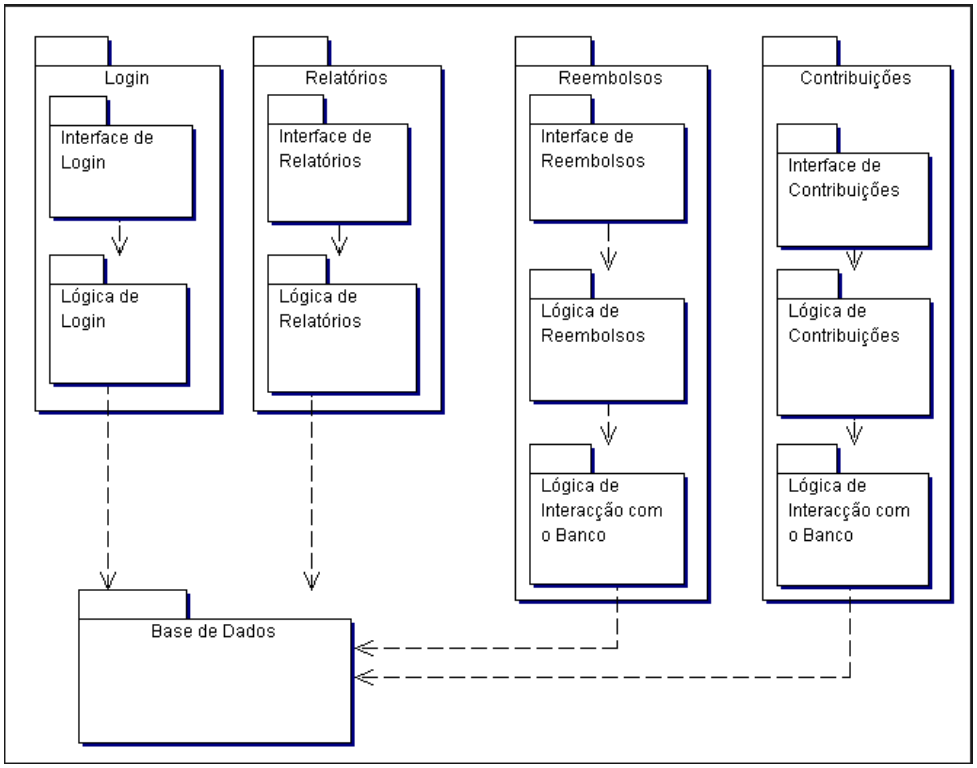

Figura 2.9: Exemplo de Diagrama de Pacotes.

Segundo o diagrama, todas os pacotes, ou partes da solução, possuem suas interfaces e lógicas e além disso se relacionam com a 'Base de Dados'.

2.2.3 MER

O *Modelo Entidade Relacionamento* (MER) foi originalmente proposto por Peter Chen, como projeto de sistemas de bancos de dados relacionais [13]. Acaba sendo uma abstração de como será o banco de dados relacional da aplicação.

O MER é um modelo conceitual, que tem como objetivo descrever objetos reais do domínio de aplicação da solução e suas características, na forma de Entidades, seus atributos e como eles se relacionam entre si.

A Figura 2.10 denota os símbolos utilizados para modelagem do MER [14].

Entidade

É a representação dos objetos do mundo real dentro do modelo. No exemplo da Figura 2.11, temos 'Aluno', 'Curso' e 'Disciplina' como entidades do domínio de aplicação. Ou seja, essas entidades representam objetos reais do ambiente universitário.

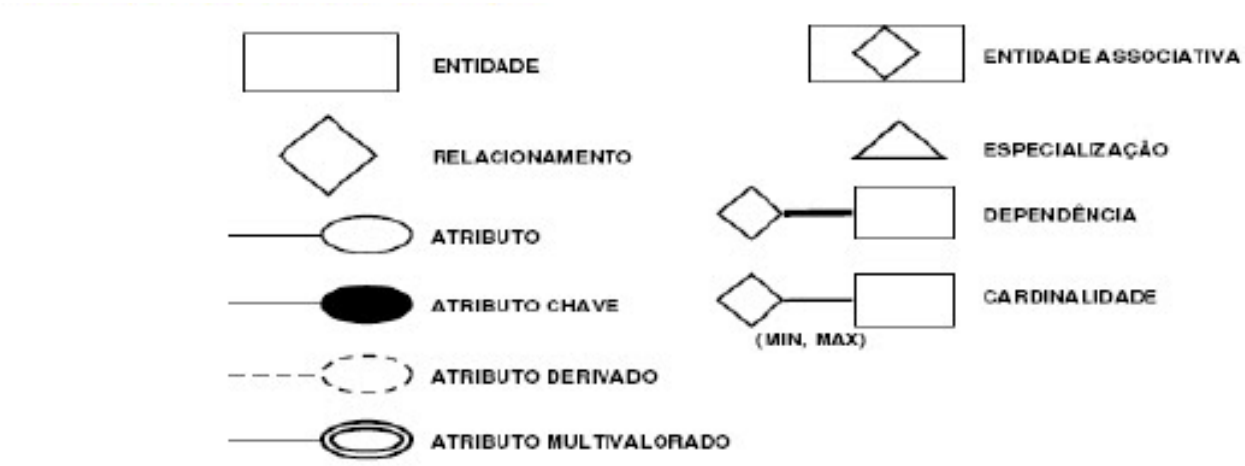

Figura 2.10: Notação Modelo Entidade Relacionamento.

Atributo

São as características que descrevem cada entidade dentro do modelo. No exemplo anterior, para a entidade 'Aluno' existem os atributos 'matrícula', 'nome', 'RG' e 'telefone', que funcionam como características para diferenciarmos cada aluno.

Chave

São características únicas que identificam um registro e que os diferenciam dos outros registros da mesma entidade. No exemplo anterior, dentre os atributos existentes para a entidade 'Aluno', a 'matrícula' e o 'RG' funcionam como identificadores únicos (chaves) para um determinado aluno, pois dois alunos não podem ter o mesmo RG e nem a mesma matrícula dentro da universidade.

Relacionamento

Uma vez identificadas as entidades, elas podem agora se relacionar e podemos definir como esses relacionamentos acontecem. No exemplo da Figura 2.11, as entidades 'Aluno' e 'Curso' se relacionam por meio do relacionamento 'cursa'. Ou seja, esse relacionamento conecta ambas as entidades fazendo com que um determinado aluno possa cursar uma ou mais disciplinas, e também com que uma determinada disciplina seja cursada por um ou mais alunos.

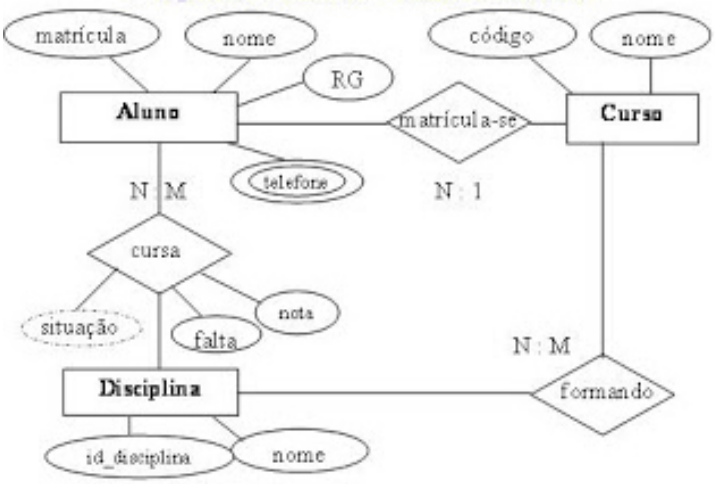

Figura 2.11: Exemplo de Modelo Entidade Relacionamento.

2.3 Documentação de Sistemas

No processo de desenvolvimento de software desse trabalho, foram utilizadas algumas macroetapas do modelo Processo Unificado (*Unified Process* - UP), proposto [12].

Segundo [11], o UP possui quatro grandes fases: concepção, elaboração, construção e transição.

Para fins de documentação, este projeto baseou-se nas duas macroetapas iniciais (concepção e elaboração), com microetapas adaptadas à necessidade do projeto. Em [11] encontramos a descrição geral dessas etapas:

Concepção: A fase inicial no processo, tem como objetivo principal levantar os requisitos da solução, compreender o sistemas de uma forma mais ampla e também entender melhor o domínio de aplicação da solução.

Como resultados dessa fase, temos documentos de requisitos e uma listagem mais simples dos diagramas de casos de usos.

Elaboração: A fase incorpora principalmente a análise e projeto da solução de uma forma mais detalhada que ajude a modelar o domínio da solução e a própria proposta de projeto.

Como resultado, essa fase entrega documentos de caso de uso mais aprofundados e detalhados e também possíveis modelos de dados a serem utilizados.

Com isso, a documentação tem grande importância no processo de desenvolvimento de um SI, pois atende a duas grandes necessidades:

1. Compreensão do Processo de Negócio

   Entender como funcionava o processo sem a utilização do SI (Modelo As-is) e como ficaria o processo após a inserção do SI (Modelo To-be), utilizando-se da modelagem BPMN.

2. Compreensão do SI

   Entender quais são os requisitos do SI. Utilizando-se da modelagem UML, o intuito é conhecer quais suas funcionalidades, quais os usuários o utilizarão e que tipos de atividades serão realizadas por eles.

2.4 Desenvolvimento de Sistemas

Conforme [11], para fins de desenvolvimento/programação, o projeto baseou-se nas duas macroetapas restantes (construção e transição) do UP, também com microetapas adaptadas à necessidade do projeto.

Como documentos base para o desenvolvimento, são utilizadas todas as modelagens e diagramações das fases anteriores, que servirão de guia para que o software desenvolvido atenda tanto as necessidades do usuário como também as do processo de negócio.

Construção: Fase responsável pela parte de implementação e testes. A entrega dessa fase é bem assertivas: software em funcionamento.

Transição: Nessa fase o sistema é instalado em seu ambiente final de utilização pelo usuário. A entrega final da etapa é ter o software funcionando em um ambiente final para utilização pelo usuário.

2.4.1 Tecnologias Web

Serão apresentadas a seguir as tecnologias utilizadas no desenvolvimento da solução: Linguagem *Python* utilizando *Django Framework* com banco de dados *PostgreSQL*.

Foi escolhido o Python, como linguagem de programação, devida à sua rápida curva de aprendizado e também ao desenvolvimento ágil que permite entregar valor ao cliente rapidamente.

Para agilizar ainda mais o desenvolvimento foi utilizado o Django Framework, por ser o mais conhecido dentre os existentes para essa linguagem e também ser o que fornece maior suporte devido à grande comunidade de apoio da ferramenta.

Para o banco de dados, foi utilizado o PostgreSQL, o sistema de bando de dados mais indicado/comum dentro da comunidade Python e Django. Sua escolha ocorreu também devido a sua robustez e estabilidade.

Python

Segundo [15], o Python é uma linguagem de programação que permite ao desenvolvedor trabalhar rapidamente e integrar vários sistemas de maneira mais efetiva. Algumas características peculiares do Python, unanimemente reconhecidas pela comunidade de desenvolvedores [16] são vistas como grandes benefícios ao se programar nessa linguagem:

1. **Linguagem amigável de fácil Aprendizado**

   Linguagem de alto nível construída para que fosse necessário investir pouco tempo, escrevendo poucas linhas de código, para produzir um programa.

   Foi desenvolvida pensando em técnicas como a DRY(Don't Repeat Yourself).

   Segundo [17], é uma linguagem de fácil aprendizado, devido às poucas palavras chaves existentes, à sintaxe simples e de fácil leitura. Isso à confia alta legibilidade/entendimento de código. Com tudo isso, a velocidade/curva de aprendizado da linguagem é altíssima. Isso faz dessa linguagem, inclusive, uma ótima opção para se iniciar os aprendizados de lógica de programação.

2. **Multiplataforma**

   Python é uma linguagem interpretada, portanto tem a capacidade de rodar em qualquer sistema que possua seu interpretador, seja Linux, Mac OS ou Windows.

   Isso permite, por exemplo, que um programador desenvolva um software em seu computador Mac OS e consiga fazer com que esse mesmo programa seja colocado e executado em um servidor Windows ou Linux.

3. **Plataforma OpenSource (Código Aberto)**

   Por ser uma linguagem livre, de código aberto, programadores de todo o mundo podem compartilhar seus módulos/pacotes, problemas e soluções visando a melhoria da linguagem, e até participando da construção da mesma.

   Essa característica também proporciona ao desenvolvedor um grande suporte à linguagem, advindo da comunidade e das suas extensas documentações e tutoriais [18].

4. **Integrações Rápidas e Eficientes**

   O Python se conecta facilmente a qualquer outro tipo de solução, ambiente ou servidor. Isso se deve à existência da grande quantidade e qualidade de módulos e pacotes (bibliotecas) desenvolvidos e compartilhados pela comunidade Python.

Pesquisas feitas pelo ranking *TIOBE Programming Community Index* [19], indicam que o Python em 2017 ocupa o 4º lugar na lista das linguagens mais populares (Nov/2017).

Esse índice analisa pesquisas realizadas nos buscadores mais populares, como, Google, Bing, Yahoo, Wikipedia, Amazon, YouTube e Baidu usando o termo "python programming". Além disso, o índice também indicou que o Python foi a Linguagem do Ano em 2007 e 2010.

Já o índice *PopularitY of Programming Language (PYPL)* [20], que analisa com qual frequência tutoriais sobre determinada linguagem são procurados no Google, posiciona o Python em 2º lugar no Ranking Mundial (Nov/2017). Segundo a Figura 2.12 [20], o Python foi a linguagem com maior crescimento em sua popularidade nos últimos 5 anos. O crescimento foi de 7,7% em 2012 a 18,6% em 2017 do total de interesses em tutoriais de linguagens de programação.

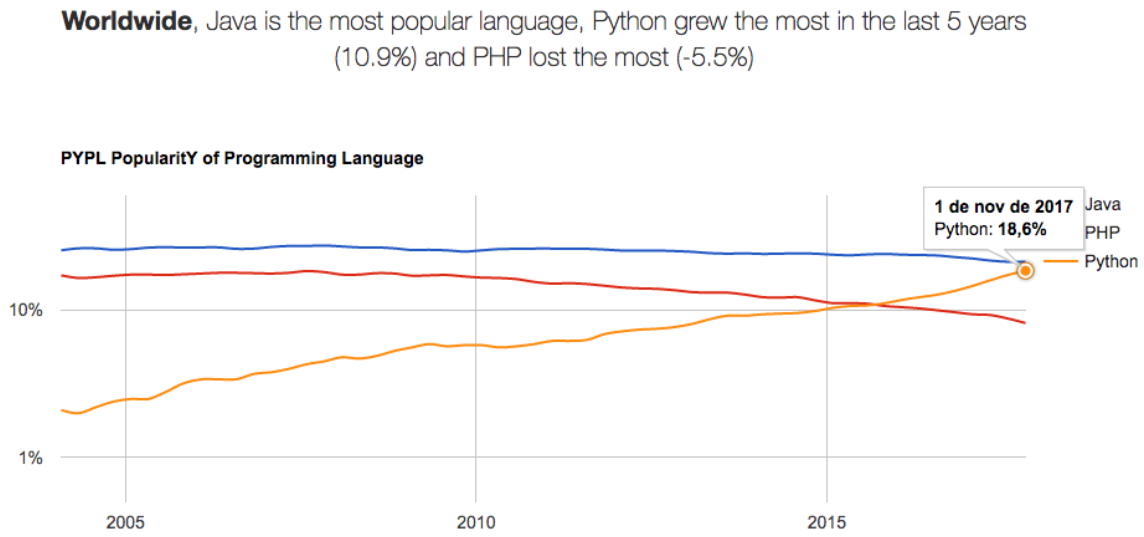

Figura 2.12: Gráfico de Popularidade pelo PYPL.

Segundo levantamento da *TechBeacon* [21], que analisou e comparou os principais rankings medidores de popularidade das linguagens de programação, em um resultado geral acoplado, o Python fica em 2º colocado na lista de linguagens de programação mais populares, perdendo apenas para o Java.

Django

O *Django* é segundo [22], um *framework* para desenvolvimento *web* de alto nível, escrito em *Python*, que proporciona rápido desenvolvimento e um design simples e pragmático. O site ainda exalta algumas características do framework:

1. Rápido - foi desenvolvido para ajudar desenvolvedores a criar aplicações o mais rápido possível.

2. Seguro - trata segurança de uma forma séria e ajuda desenvolvedores a evitar os mais comuns erros de segurança.

3. Escalável - sites com grande tráfego de acessos podem se beneficiar da capacidade do Django de escalar rápido e flexivelmente.

Segundo [23], o *Django* é o *framework web* mais popular para *Python*.

O *Framework* organiza internamente sua estrutura a partir de Aplicativos (*Apps*). Cada *App* funciona como um componente de um sistema e podem ser adicionados e reutilizados em outros projetos.

O *Django* utiliza um padrão de projeto muito semelhante ao padrão de projeto *Model-View-Controller* (MVC), o padrão *Model-Template-View* (MTV) [24], próprio do *Django*.

Cada *App* possui suas próprias *Views*, *Models* e *Templates*. Como mostra a Figura 2.13, esses arquivos desempenham as seguintes funções:

Model - Responsável por definir a estrutura dos dados e seus relacionamentos e também interagir com o banco de dados.

Template - Parte visual que o usuário do site irá enxergar e interagir.

View - Onde ficam as funções Python responsáveis pela lógica de negócio e por retornar os dados que o usuário interage no Template. A views costumam utilizar as Models para buscar e entregar dados.

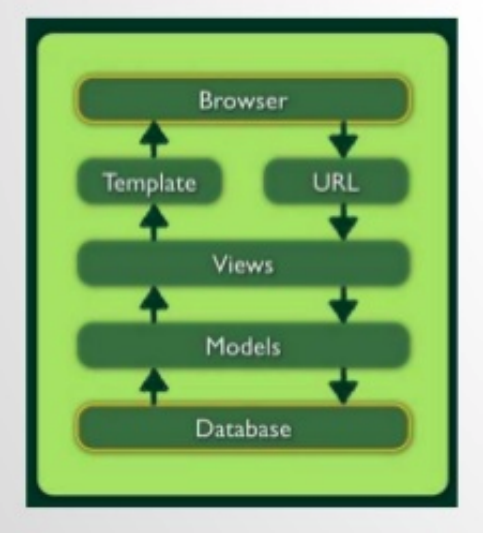

Figura 2.13: Padrão Model Template View (MTV) do Django.

O site Python Programming Language ainda comenta alguns itens existentes dentro de um *Framework Web*, que facilitam o desenvolvimento. O Framework contém todos os tipos de funcionalidades que são utilizadas quando se cria uma aplicação Web:

- Operações HTTP;
- Manipulação e acesso à banco de dados;
- Roteamento de URLs;
- Suporte à sessões;
- Templates visuais;
- Processos de Autenticação.

PostgreSQL

O *PostgreSQL*, segundo [25], um sistema de banco de dados relacional de código aberto *(open source)*.

Publicações feitas em [26] sobre os sistemas de banco de dados suportados pelo Python, apontam que PostgreSQL e MySQL são os dois bancos de dados mais comuns nas aplicações dessa linguagem.

Tais publicações ainda apontam que as versões do PostgreeSQL são vistas como mais robustas e estáveis quando comparadas ao MySQL, SQLServer e Oracle. Tudo isso, pois os desenvolvedores *Python* tendem a usar mais o PostgreeSQL, logo existe um maior número de exemplos de utilização, tutoriais e atualizações para esse banco de dados.

Por ser código aberto, o PostgreSQL permite que desenvolvedores utilizem um ou mais bancos de dados sem precisar ter custos de licença em suas aplicações. Ou seja, a licença de uso é bem mais barata quando comparada à outros fornecedores de banco de dados, ainda mais em aplicações que necessitam de alto desempenho.

Heroku

Segundo [27], o *Heroku* é um plataforma em nuvem que permite desenvolvedores construir, entregar, monitorar e escalar aplicações.

O Heroku funciona basicamente como um servidor para hospedagem de aplicações, mas que, ao contrário de outros servidores, facilita para o desenvolvedor e abstrai todas as complexidades de configuração de máquinas e ambientes. Assim é possível com o Heroku a configuração e entrega de aplicações para o cliente final na nuvem em poucos minutos.

Git

Baseado em [28], *Git* é um sistema para controle de versionamento de código distribuído, grátis e de código aberto. Foi desenvolvido para lidar desde pequenos a grandes projetos de uma maneira rápida e eficiente.

Segundo publicações da *Atlassian* [29], empresa de tecnologia focada em desenvolvimento de softwares de colaboração para times, utilizar a tecnologia Git permite que cada desenvolvedor possa trabalhar separadamente em sua cópia completa do código mestre e também saber o histórico de mudanças daquele código. Esse cenário possui um melhor desempenho do que ter o código completo da aplicação em um único lugar para todos os desenvolvedores trabalharem no mesmo código.

Após os desenvolvimentos em paralelo, a ferramenta permite que os códigos sejam mesclados ao código mestre, adicionando assim cada funcionalidade/correção de cada desenvolvedor sem causar impactos uns aos outros.

2.4.2 Tecnologias Mobile

Em [30], existe uma avaliação comparativa entre as duas abordagens de desenvolvimento mobile: a híbrida (Multiplataforma) e a nativa. Os autores confirmaram que a abordagem nativa não é melhor que a híbrida ou vice-versa, sendo elas apenas distintas entre si.

Híbrido x Nativo

Ao se desenvolver uma aplicação para dispositivos móveis, existem dois possíveis caminhos que podem ser escolhidos: desenvolvimento Híbrido (Multiplataforma) ou desenvolvimento Nativo com linguagens para cada Sistema Operacional (SO).

Esse desenvolvimento requer uma análise de vários fatores que influenciam na escolha de qual abordagem escolher em determinada situação.

Como mostra a Figura 2.14, a escolha da abordagem a ser utilizada necessita uma avaliação do projeto nos seguintes fatores:

- Complexidade da aplicação
- Expertise da equipe
- Nicho de mercado
- Prazo de desenvolvimento
- Capital disponível para investimento

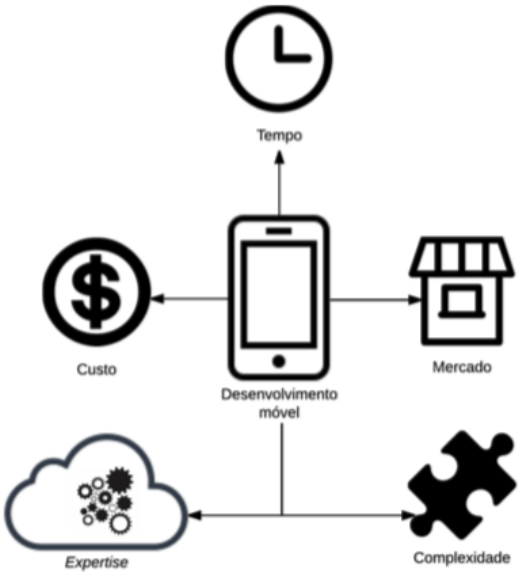

Figura 2.14: Fatores a serem avaliados no desenvolvimento móvel.

Em [30], os autores constataram que o desenvolvimento multiplataforma apresenta vantagens devido a implementação de apenas um código, que gera aplicativo para várias lojas (GooglePlay e AppStore). Outras vantagens foram detectadas, como, a redução do custo e do tempo de desenvolvimento. No quesito desempenho, a diferença entre as duas abordagens é imperceptível ao usuário final. A Figura 2.15, refere-se às vantagens e desvantagens de ambas abordagens que foram encontradas em suas pesquisas. Com isso a abordagem Híbrida pode ser a melhor escolha para projetos menores, mais simples, de curto prazo ou até mesmo versões Beta.

Ionic

O Ionic é segundo [31], o melhor e mais famoso framework para desenvolvimento de aplicativos para dispositivos móveis de código aberto.

Para construção de aplicativos com Ionic, o desenvolvedor precisa ter conhecimentos de linguagens e ferramentas web, como Angular [32], Javascript [33], TypeScript [34], HTML5 [35] e CSS [36]. A partir de um único código nessas linguagens o framework gera o aplicativo para ambas as plataformas (Android e iOS) em código nativo e pronto para executar.

Após essa conceituação sobre o que são Sistemas de Informação, como são modelados, documentados e desenvolvidos, pode-se passar ao entendimento do domínio da solução. Segue-se com o desdobramento da cadeia produtiva do leite, bem como as fraudes e

	Vantagens	Desvantagens
Nativo	• Explora todas as capacidades do dispositivos • Maior performance • Oferece experiência nativa de usabilidade para o usuário • Integração próxima com o sistema e outros aplicativos	• Necessário um aplicativo por plataforma • Mais caro e mais difícil de manter, por ter que manter vários *apps* diferentes • Requer domínio de vários ambientes e linguagens para cada plataforma
Multiplataformas	• Só é preciso dominar uma linguagem e um ambiente • Desenvolve apenas um código e pode distribuir em várias lojas de *apps*, o que faz com que o alcance do *app* seja maior • Reduz custo, tempo e esforço de desenvolvimento e manutenção	• Não possui acesso a todas as funcionalidades do dispositivo • Menor performance comparada ao nativo, linguagens interpretadas e compiladas • Não possui a mesma usabilidade e experiência de uso das aplicações nativas • Não possui as últimas atualizações lançadas pelo *SO*, pois para cada atualização do *SO*, é preciso ser feita uma atualização no multiplataforma

Figura 2.15: Vantagens e desvantagens das abordagens de desenvolvimento nativo e multiplataforma.

adulterações existentes na mesma e ainda a atuação do FORMFIX® no combate dessas fraudes.

2.5 Leite

Segundo [37], o leite é o produto integral da ordenha de uma fêmea leiteira, encontrado em estado líquido, branco, opaco, de sabor levemente adocicado e de odor pouco acentuado.

Segundo [38], o leite de vaca pode ter alterações em sua composição percentual de acordo com a idade do animal, a raça, o clima, o tipo de alimentação, entre outros fatores. Sua composição, em média, é de 12,5% de extrato seco e 87,5% de água.

Como mostra a Figura 2.16 [38], fazem parte do extrato seco os sais minerais com teor de 0,8%, as gorduras com 3,5%, as proteínas também com 3,5% e a lactose com o maior teor de 4,7%.

É perceptível o grande volume de água existente no leite, fato esse que influencia consideravelmente a sua densidade. Em casos de adulteração, esse fato evidencia bastante as mudanças da composição química do leite.

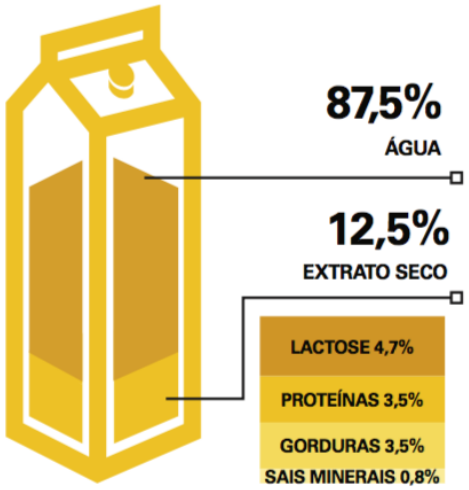

Figura 2.16: Composição do Leite de Vaca.

2.5.1 Cadeia Produtiva do Leite

Segundo [1], iniciativa lançada pela unidade Embrapa Gado de Leite, as oportunidades para desenvolvimento de soluções para o Agronegócio do Leite se dividem nas fases da cadeia produtiva desse bem. A cadeia produtiva do leite tem suas peculiaridades dependendo do laticínio ou da fazenda, porém, como mostra a Figura 2.17, pode ser geralmente compreendida em cinco macro etapas.

Figura 2.17: Cadeia Produtiva do Leite [1].

Insumos Agropecuários A primeira delas é relacionada aos fornecedores de insumos agropecuários. As fazendas precisam adquirir estes insumos, pois são de extrema importância para a criação de seus rebanhos leiteiros.

Dentro dos insumos agropecuários estão inclusos: sementes e adubos utilizados no plantio de pastagens para o engorde do gado, rações especiais e complementos nutricionais que os bovinos precisam ingerir. Fazem parte também as máquinas e equipamentos para ordenha e cuidado das vacas, produtos veterinários diversos, sêmen para inseminação artificial das vacas, embalagens, entre outros.

Fazenda O processo de produção do leite em si, se inicia nas fazendas, onde cada propriedade precisa passar por uma auditoria do laticínio para poder começar a ser seu fornecedor de matéria prima.

Na fazenda, a ordenha é feita alinhada à técnicas básicas de manejo de ordenha. Além disso, para garantir a qualidade após a ordenha, o leite é armazenado em tanques a uma temperatura de no máximo de 4°C.

Logística Na logística de transporte, depois do leite ser coletado nas fazendas com todo o cuidado necessário, o mesmo é transportado por modernos caminhões isotérmicos, mantendo-o fresco e protegido durante o trajeto até a fábrica (laticínio), garantindo que seja sempre entregue a uma temperatura inferior a 10°C.

Indústria de Laticínios Logo ao chegar na fábrica, o leite in natura de cada caminhão é submetido à diferentes análises. Também são feitas análises sensoriais para verificação do sabor, aroma e cor.

Mercado e Consumidores Essa etapa se caracteriza pela comercialização dos produtos lácteos, envolvendo atacadistas, supermercados, padarias e consumidores finais.

2.5.2 Fraudes e Adulterações

Desde a produção até a etapa de transporte, o leite pode estar sujeito a alterações. Essas alterações podem ter sido não intencionais, fruto da não observância de aspectos higiênico-sanitários no processo de obtenção e manipulação do produto. Entretanto, tais alterações podem também ser fruto de ações de adulteração do produto pela adição de substâncias fraudulentas.

É proibido, segundo a legislação brasileira [39], a adição de qualquer substância química na conservação do leite e estabelece, assim, padrões de qualidade para o leite cru e beneficiado. São utilizadas na fraude de leite substâncias neutralizantes, conservantes e reconstituintes, dentre as mais conhecidas das conservantes está o formol. Tal substância atua sobre a microbiota do leite inativando a ação deteriorante de micro-organismos.

Existe uma metodologia convencional que detecta a adição de formal no leite, metodologia recomendada pela Instrução Normativa N68/2006 [40]. Segundo [41], nesse teste o formol é separado do leite por destilação, e após um processo de aquecimento e oxidação, resulta em um composto de coloração violeta. Entretanto, esse método demanda uma quantidade grande de tempo na sua realização, não permitindo que a pesquisa de formol no leite seja rápida e simples.

2.5.3 FORMFIX®

Para resolução do problema de tempo de análise do método convencional, foi desenvolvido pela *Macofren Tecnologias Químicas LTDA* [42] o reagente FORMFIX® [7]. Esse reagente indica a presença de formaldeído no leite de uma forma mais simples e rápida, com resultados sendo obtidos em questão de minutos, por meio de uma análise colorimétrica.

O reagente tem forma líquida e transparente, mas quando em contato com leite a amostra apresenta uma coloração que varia de acordo com a presença de formol. Como

mostra a Figura 2.18, os tons de cor das reações podem variar de róseo pálido (resultado negativo) até roxo (resultado positivo), respectivamente os tubos de ensaio T0 e T7.

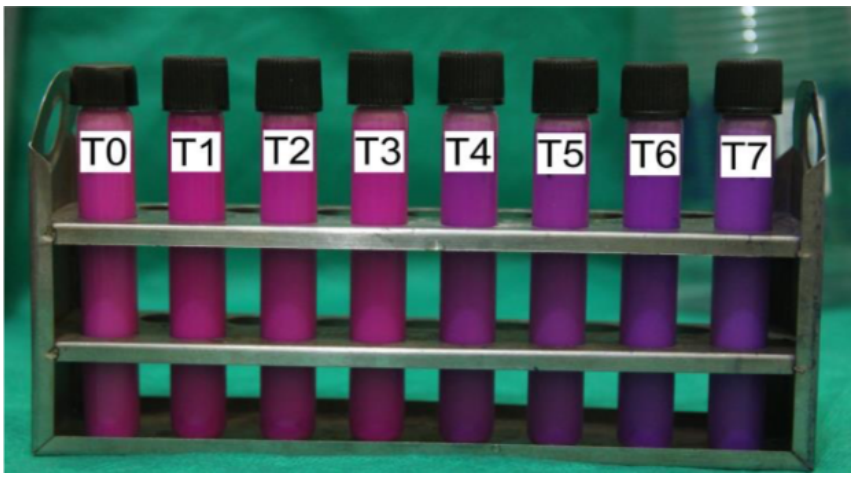

Figura 2.18: Resultados das reações colorimétricas da pesquisa [2] de formol adicionado ao leite pelo reagente FORMFIX®.

Segundo [2], sobre o desempenho do FORMFIX®, o reagente se mostrou eficiente na detecção de formol quanto adicionado ao leite cru,e quando comparado com o método convencional é capaz de detectar a substância quimica a partir de 0,05%. O método convencional é capaz de detectar a partir de 0,005%. Entretanto, o reagente pode ser considerado uma boa opção para a finalidade do teste de detecção de formol, pois a rapidez na obtenção de resultados e a facilidade na execução da análise servem de fatores compensatórios.

Além da melhora no tempo de análise, o reagente também proporciona que o teste possa ser feito à qualquer momento da cadeia produtiva, seja na coleta do leite na fazenda, na entrega do leite ao laticínio, ou até mesmo em estabelecimento de recebimento e armazenamento de leite, que estão geograficamente posicionados na rota entre laticínios e fazendas, conhecidos como entrepostos.

Segundo [41], o tempo de persistência do formol adicionado ao leite é de até 96 horas após sua adição, sendo assim, ainda é possível a detecção do mesmo por meio do reagente FORMFIX®.

Após as devidas conceituações sobre Sistemas de Informação, como são modelados, documentados e desenvolvidos e também sobre a cadeia produtiva do leite, as adulterações por formaldeído e suas ferramentas de análise pode-se passar à apresentação da proposta de solução a ser desenvolvida.

Capítulo 3

Proposta de Solução

Este trabalho foi baseado em algumas macroetapas do modelo UP, conforme proposto por [11], composto por quatro fases: concepção, elaboração, construção e transição.

3.1 Concepção

Nesta etapa foi utilizada a metodologia de pesquisa exploratória, com abordagem qualitativa, realizada por meio de entrevistas de coleta de requisitos e análise de documentos seguida pelo desenvolvimento de um artefato de software para solucionar a questão levantada.

As primeiras atividades dessa etapa são as entrevistas com os interessados da solução e com os especialistas no domínio do problema, a fim de entender o domínio de aplicação da solução e levantar os requisitos de software.

3.1.1 Entendimento do Domínio do Problema

Visando conhecer o ecossistema do Leite no Brasil e entender como funciona seu processo produtivo foram realizados os seguintes passos:

1. Reunião com participantes/especialistas do processo
2. Modelagem da cadeia produtiva de Leite antes da solução - *As-is*
3. Modelagem da cadeia produtiva de Leite depois da inserção da solução - *To-be*
4. Validação dos modelos com participantes/especialistas

Foram realizadas reuniões com a Profa. Dra. Marcia Aguiar (FAV/UnB), como especialista na área, para um melhor entendimento de como funciona toda a cadeira produtiva de Leite em suas microatividades.

Com isso foi possível modelar, a um nível de atividades, cada umas das cinco macro etapas da cadeia produtiva de leite e também os atores responsáveis por cada uma dessas atividades, antes e depois da inserção da solução. As Figuras 3.1 e 3.2 exemplificam uma modelagem feita para a Etapa 2 da cadeia produtiva, na Fazenda. As demais modelagens podem ser encontradas de forma integral no Anexo I deste trabalho.

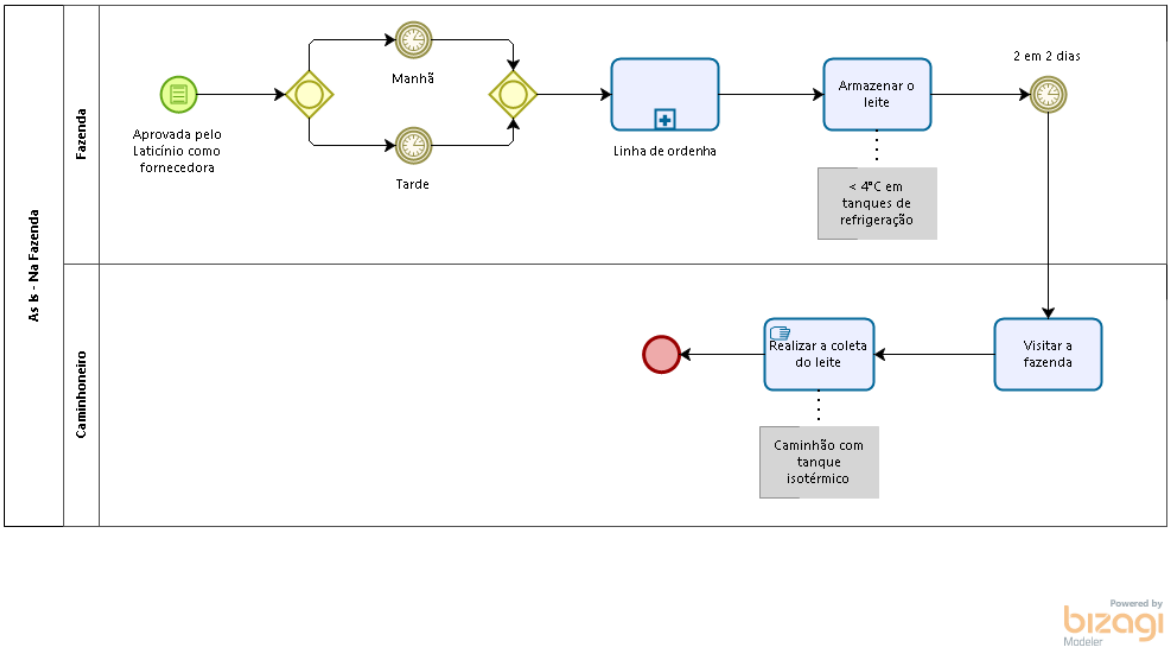

Figura 3.1: Modelagem As-Is do processo na Fazenda.

Uma visão mais holística do processo, trazida pelo seu mapeamento, possibilitou o entendimento de alguns pontos importantes.

Os problemas no processo As-Is:

- Falta de registro de realização dos testes com FORMFIX®
- Inviabilidade de rastrear o local da adulteração

As soluções no processo To-Be:

- Todos os testes feitos com FORMFIX® começarão a ser registrados via aplicativo para dispositivos móveis, armazenando informações, como, data/horário/local de realização do teste, quantidade de litros ao qual o teste se refere, resultado identificado pelo usuário, cor identificada, foto do teste.
- Realizar o registro do teste em dois ou mais pontos estratégicos do processo: na fazenda (antes da coleta do leite), no laticínio (antes da entrega) e no entreposto (na coleta e na entrega), caso exista esse ponto.

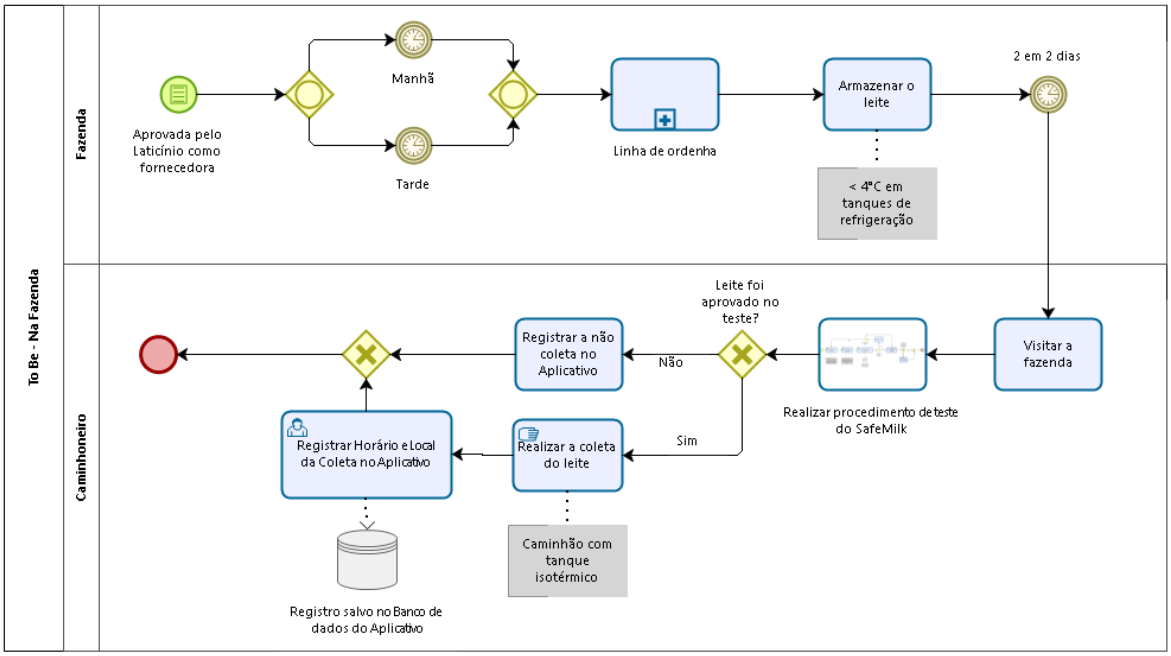

Figura 3.2: Modelagem To-Be do processo na Fazenda.

3.1.2 Levantamento de Requisitos

Visando entender melhor a solução pretendida e as respectivas funcionalidades, seus atores e as ações que seus atores realizam, foram seguidos os seguintes passos:

1. Reunião com clientes da solução/especialistas do FORMFIX®;
2. Levantamento dos requisitos da solução;
3. Listagem de casos de uso da solução;
4. Validação dos levantamentos com os clientes/especialistas.

Foram realizadas reuniões com os responsáveis pela Macofren, Sr. Renato Santana e Sr. Arilson Onesio, como especialistas do FORMFIX e clientes da solução, para um melhor entendimento das necessidades da mesma quando estiver interagindo com os testes do reagente, e com seus usuários.

Com isso foi possível criar, uma documentação mínima necessária de requisitos e também uma listagem mais simplificada dos casos de uso da solução. A seguir mostramos alguns exemplos de requisitos levantados. O documento completo pode ser encontrado no Anexo II deste trabalho.

Exemplos de Requisitos levantados:

- Usuários
    1. Produtor;
    2. Caminhoneiro;
    3. Técnico do Laticínio;
    4. Auditor do MAPA (Fiscal);
    5. Administrador do Laticínio.
- Requisitos Funcionais
    1. Todo usuário precisa/deve realizar login;
    2. Caminhoneiros e técnicos registram testes;
    3. Caminhoneiros e técnicos assistem vídeos do procedimento do FORMFIX®;
    4. Administradores dos Laticínios, Auditores do Ministério e Produtores visualizam os testes registrados;
    5. Administradores dos Laticínios gerenciam a logística de transporte.
- Requisitos não funcionais

1. Devem ser registrados até três testes por ponto de coleta/entrega;
2. A interface e usabilidade do Aplicativo deve ser simples e intuitiva para pessoas tecnologicamente limitadas.

Casos de Uso: Foi gerada uma listagem mais geral dos casos de uso, como pode ser visto na Figura 3.3.

Ao final dessa etapa foi possível compreender também que a solução tem por finalidade garantir uma segurança ao se consumir leite, por isso, a sugestão de nome pensada para o produto foi de *SafeMilk* (leite seguro).

3.2 Elaboração

Passada a fase de entendimento do problema, do ambiente que ele está inserido e dos requisitos da sua solução, inicia-se a fase de planejamento de como será o software.

Ao projetar como será o software, é exigido que se detalhe melhor tanto os requisitos quanto os casos de uso da fase anterior. Outra entrega final é uma proposta de modelo de dados, algo importantíssimo que ajuda no entendimento de todas as relações entre entidades e funcionalidades.

Também faz parte do projeto da solução, pensar no design visual e na experiência de usuário, ou seja, como o usuário interage com suas telas, que cores fazem mais sentido para ele, onde devem estar posicionados os elementos da tela. Esses são questionamentos muito importantes que podem fazer com que o usuário aprecie ou não a solução, impactando diretamente no seu sucesso.

3.2.1 Visão Geral da Solução

Modularidade

Por meio dos requisitos levantados e dos casos de uso, foi possível entender quais os atores da solução e as ações que podem realizar(funcionalidades).

Com a finalidade de agrupar todas as funcionalidades que são ou parecidas, ou referentes a um mesmo assunto, ou que podem ser resumidas por um único tema, a solução foi modulada/dividida em partes distintas.

Cada parte, ou módulo, contém seu conjunto de funcionalidades com algo em comum, e que quando juntas, fazem desse módulo uma parte o máximo possível independente do resto dos módulos. Essa técnica ajuda tanto na simplificação, de uma grande solução em pequenas menores soluções, quanto na facilitação de manutenções no futuro, na qual será possível corrigir módulos separados sem que alguma correção em um atrapalhe funcionalidades de outro módulo.

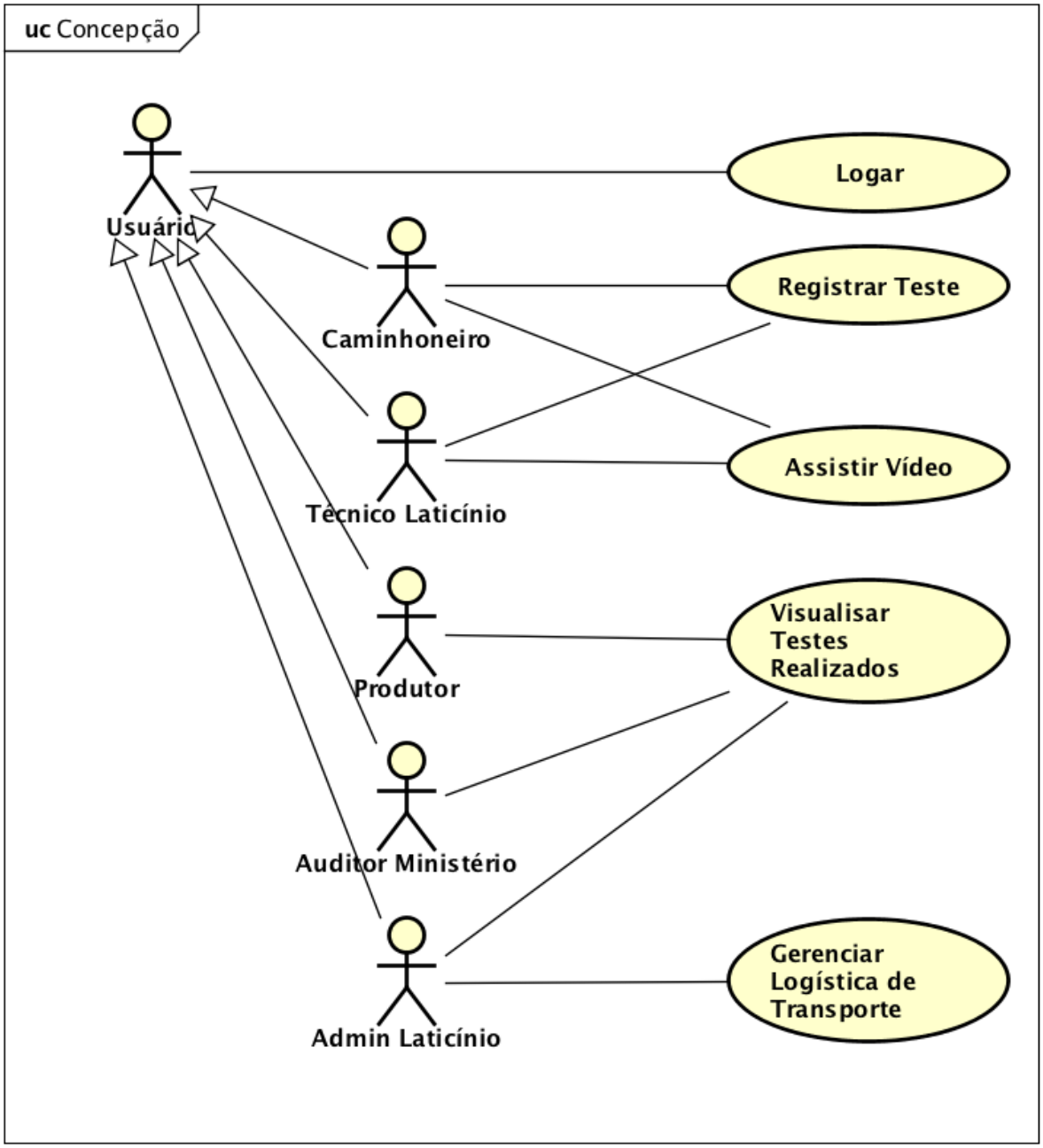

Figura 3.3: Casos de Uso da solução.

Como pode ser visto na Figura 3.4, a solução foi dividida em cinco módulos:

- **Autenticação**

  Módulo responsável por gerenciar o acesso de usuários à solução e também às requisições realizadas pelos dispositivos móveis.

- **Logística**

  Conjunto de funcionalidades responsável pelo gerenciamento da logística de transporte de um laticínio. O cadastramento de todas as linhas de transporte, dos pontos de coleta/entrega, das fazendas, dos entrepostos e dos caminhões fazem parte desse módulo.

- **Testes Colorimétricos**

  Módulo responsável pelo registro dos testes colorimétricos realizados com o FORMFIX®. Compreende o cadastro dos testes com suas respectivas informações, como, data e hora, foto do teste, quantidade de litros coletada, resultado do teste, cor da amostra.

- **Estatísticas**

  Parte da solução responsável por disponibilizar visualmente um resumo geral dos resultados dos testes realizados.

- **Aprendizado**

  Módulo responsável pelo aprendizado de técnicos e caminhoneiros na realização do processo do FORMFIX®, por meio de vídeos tutoriais.

Plataformas: Web e Mobile

A existência de atores que tem necessidades diferentes e que utilizam ferramentes de trabalho diferentes evidencia a necessidade da criação de duas partes igualmente importantes para a solução: uma aplicação Web (sistema Web) e uma aplicação para dispositivos móveis (aplicativo).

O caminhoneiro, em seu dia-a-dia, coleta/entrega leite em muitas localidades, o que deixa seu dia bastante agitado, logo uma versão mobile da solução seria muito mais adequado às características do seu trabalho.

Imaginemos se ele precisasse utilizar uma máquina fotográfica digital para registrar todos os testes feitos em cada coleta/entrega e abrir o computador também para registrar outros dados. Além disso no final do dia ele teria que dedicar algum tempo para compilar as fotos da câmera, passá-las para o computador e relacioná-las aos dados dos testes, para finalmente adicionar tudo na versão Web. Claramente, um tanto custoso.

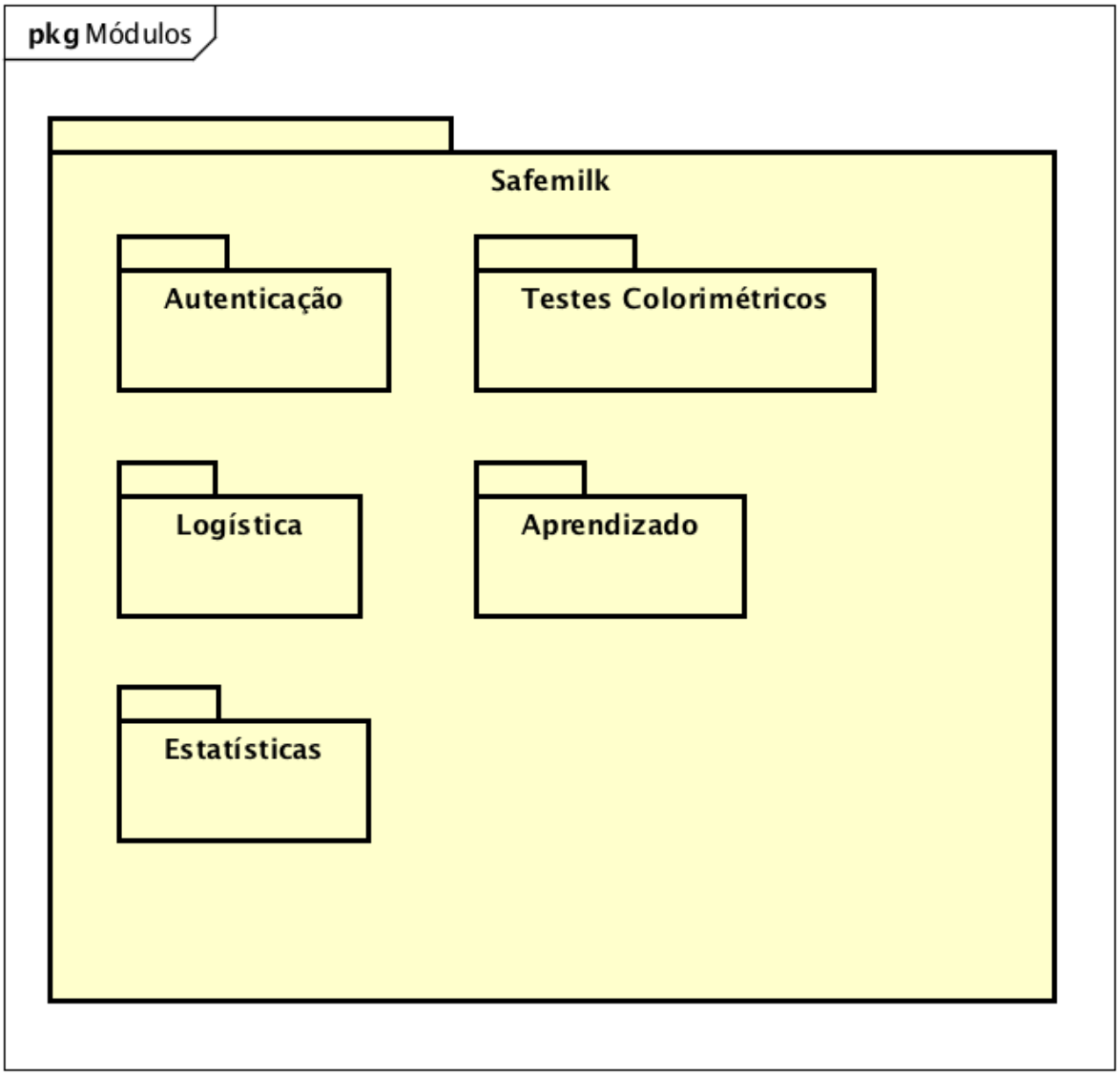

Figura 3.4: Diagrama de Módulos da solução.

Já para um técnico ou para um administrador do laticínio, que precisam, por exemplo, analisar e acompanhar painéis gerenciais. Um painel desses tenderia a ter vários gráficos e listas sobre os testes realizados no dia, por isso, talvez a tela de um dispositivo móvel não seria a mais apropriada para esse tipo de visualização. Para essa atividade uma tela de computador seria bem mais cômoda e produtiva para esses atores.

Afim de simplificar a primeira versão (Beta) da solução e também para atender, primeiramente, às necessidades que mais geram valor para cada usuário, o projeto foi pensando de modo que o sistema web e o aplicativo funcionem de forma complementar.

Ou seja, as funcionalidades foram, em sua maioria, posicionadas ou em uma plataforma ou em outra, de acordo com a necessidade do usuário.

Como pode ser visto na Figura 3.5, módulos como o de Testes Colorimétricos, por exemplo, pertencem ao escopo do aplicativo pois é mais direcionado aos caminhoneiros, os que mais a utilizam. Módulos como o de Logística, pertencem ao escopo do sistema web, pois o cadastramento das rotas que precisam ser seguidas é responsabilidade do administrador do sistema.

Em um futuro, com a evolução da solução, caso seja vista a necessidade de todas os módulos estarem nas duas plataformas e isso não dificulte o uso da aplicação, é algo que pode ocorrer pois a modularidade permite adicionar novos módulos sem interferir nos já existentes.

3.2.2 Casos de Uso Detalhados

Nos casos de uso da etapa de concepção pôde-se visualizar os atores principais do processo e conectá-los às funcionalidades que a solução necessita ter.

Com essa conexão foi possível detalhar os casos de uso para cada ator e também para cada plataforma foco, como apresentado nas Figuras 3.6 e 3.7.

3.2.3 Modelo de Dados

O modelo de dados tem grande utilidade pois será utilizado de base para a formulação do banco de dados da etapa de construção.

A Figura 3.8 mostra a Modelagem Entidade Relacionamento (MER) definida, visualmente representando uma ideia do modelo de dados a ser utilizado, com as seguintes entidades:

- **Usuário**

  Entidade que armazena as informações dos usuários da solução, como, *username*, *email* e *password*.

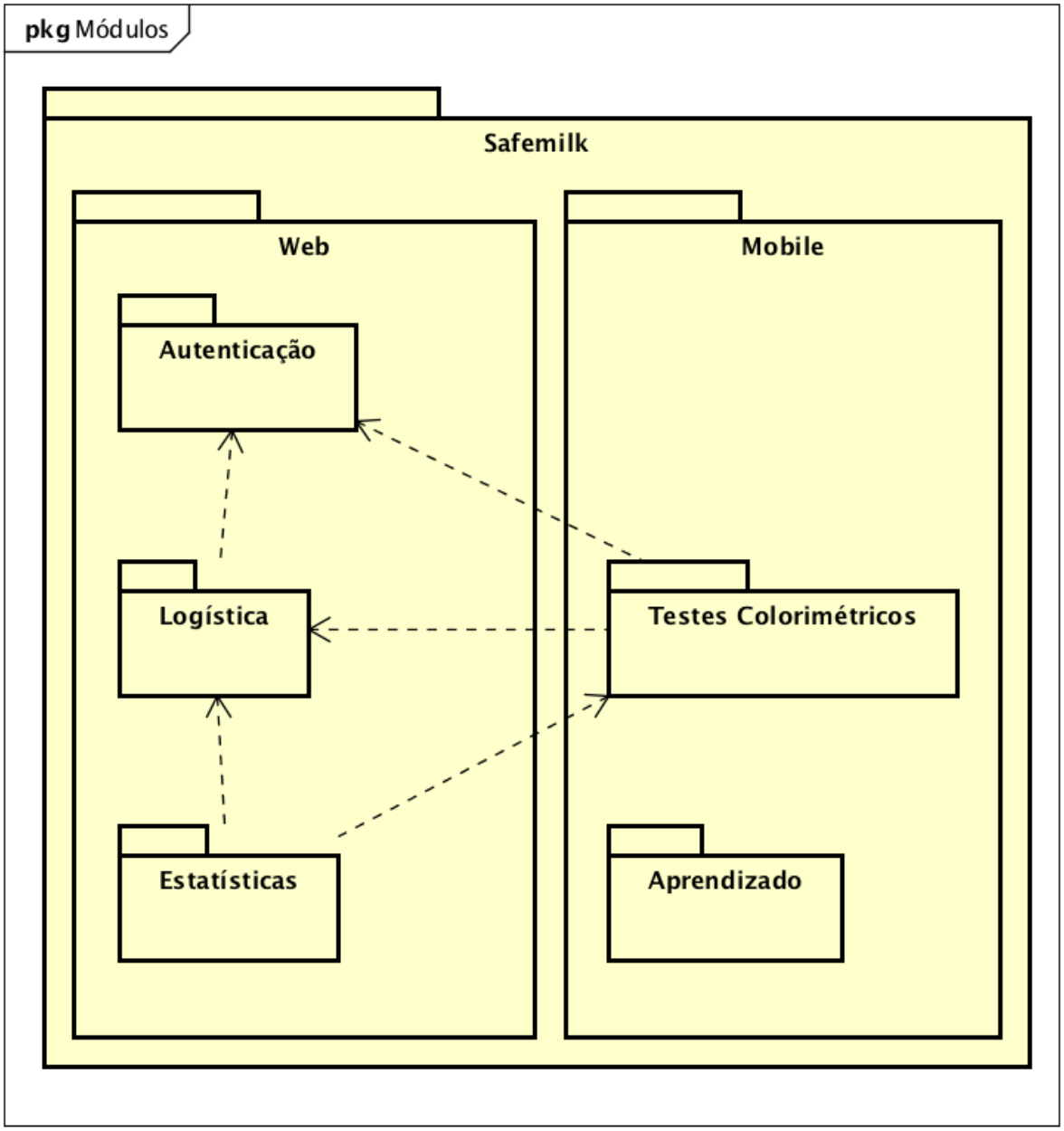

Figura 3.5: Diagrama de Módulos da solução.

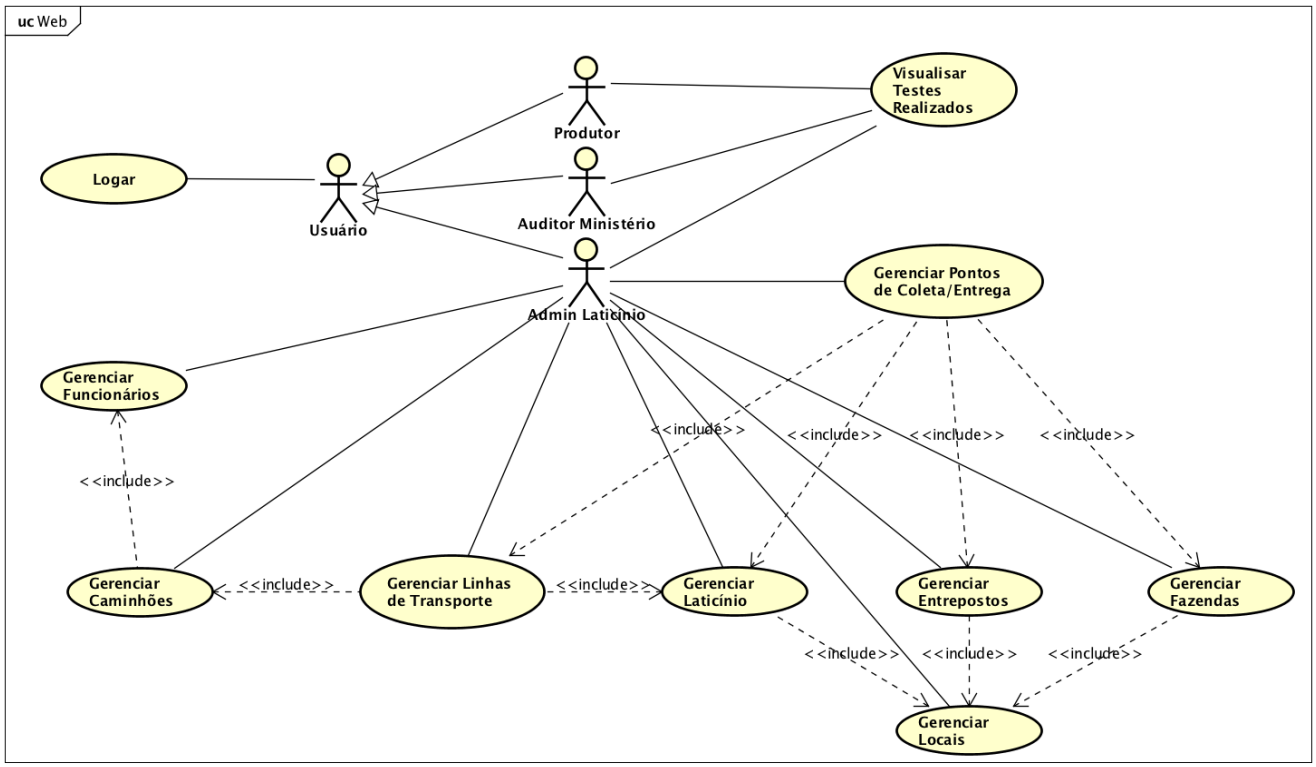

Figura 3.6: Diagrama de Casos de Uso da versão Web.

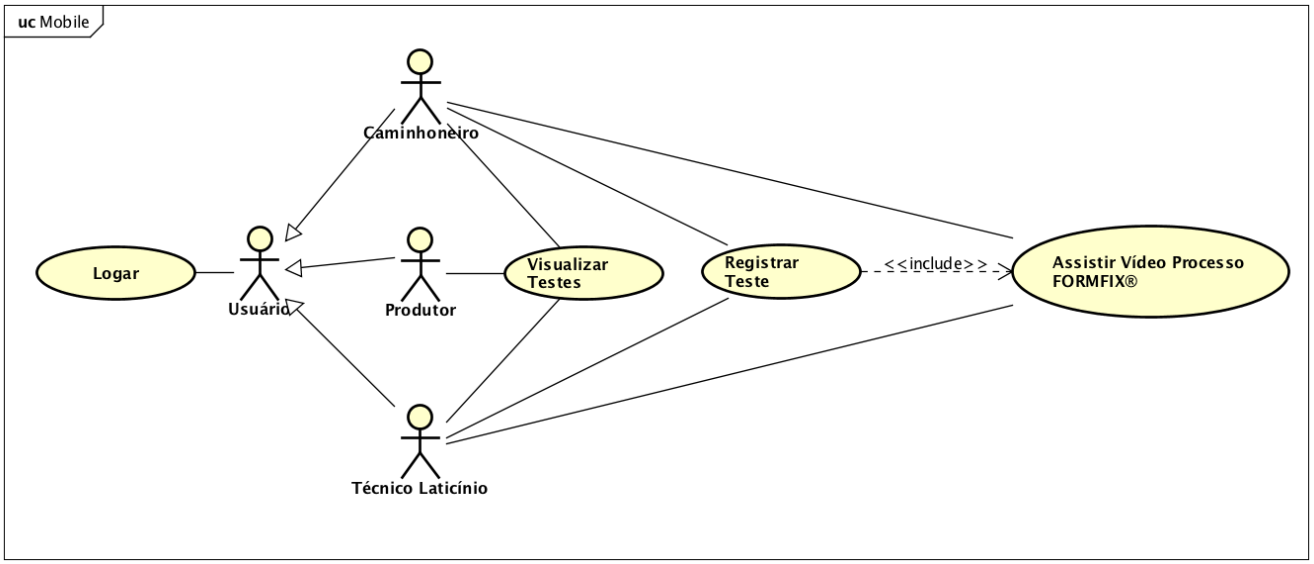

Figura 3.7: Diagrama de Casos de Uso da versão Mobile.

- **Token**

  Entidade que armazena uma chave de acesso *(key)* de cada usuário, utilizado para requisições de dados e login na solução.

- **Caminhão**

  Entidade que armazena as informações do caminhão, como, placa, chassi, marca, modelo, cor. Essa entidade também se relaciona com a entidade Usuário, visto que armazena também a informação do caminhoneiro que é motorista do um caminhão.

- **Local**

  Entidade que armazena as informações de uma localização em um mapa, como, nome, latitude, longitude, estado, cidade, endereço.

- **Fazenda**

  Entidade que armazena as informações de uma fazenda, como, nome, CNPJ, proprietário. Essa entidade se relaciona com a entidade Local, visto que armazena também a informação de sua localização. Essa entidade também se relaciona com a entidade Usuário, visto que armazena também a informação de seu proprietário.

- **Entreposto**

  Entidade que armazena as informações de um entreposto, como, nome e CNPJ. Essa entidade também se relaciona com a entidade Local, visto que armazena também a informação de sua localização.

- **Laticínio**

  Entidade que armazena as informações de um laticínio, como, nome, CNPJ e quantidade de reagente (FORMFIX®) que possui. Essa entidade também se relaciona com a entidade Local, visto que armazena a informação de sua localização e também com a entidade Usuário, já que armazena a informação de quem é o proprietário/administrador do mesmo.

- **Linha**

  Entidade que armazena as informações de uma linha de transporte, como, nome, origem, destino e se está ativa. Essa entidade também se relaciona com a entidade Laticínio, visto que armazena a informação do laticínio a qual pertence e também com a entidade Caminhão, já que armazena a informação do mesmo que percorre essa linha.

- **Processo**

Entidade que armazena as informações dos possíveis processos a serem realizados em uma linha (coleta ou entrega).

- **Tipo do Ponto**

  Entidade que armazena as informações dos possíveis tipos de pontos de coleta/entrega que existem em uma linha (fazenda, entreposto ou laticínio).

- **Ponto**

  Entidade que armazena as informações de um ponto de coleta/entrega em uma linha de transporte, como, ordem em que esse ponto será visitado, data que foi visitado, hora que foi visitado, litros a serem coletados/entregues e se essa coleta/entrega foi realizada.

  Essa entidade se relaciona com a entidade Linha, visto que armazena a informação de qual linha esse ponto pertence, e também com as entidades Processo e Tipo-Ponto, já que armazenam ,respectivamente, qual o processo a ser executado naquele ponto(coleta ou entrega) e onde será feita aquele processo (fazenda, entreposto ou laticínio).

  Além disso, essa entidade possui outros importantes relacionamentos com as entidades Laticínio, Entreposto e Fazenda. Apenas uma dessas informações é armazenada para um determinando ponto, ou ele pode ser uma fazenda, ou um laticínio, ou um entreposto. Caso o atributo que escolhe o tipo do ponto seja fazenda, é preciso armazenar qual é a exata fazenda a ser visita, e assim, respectivamente, caso o tipo do ponto seja entreposto ou laticínio.

- **Cor**

  Entidade que armazena as informações das possíveis cores a serem encontradas em um teste (rosa ou roxo).

- **Teste**

  Entidade que armazena as informações de um teste realizado em um determinado ponto, como, nome, data do teste, horário do teste, resultado do teste (seguro ou contaminado), a quantos litros aquele teste se refere, foto do teste colorimétrico.

  Essa entidade se relaciona com a entidade Cor, visto que armazena a informação de qual cor encontrada no teste, e também com as entidades Usuário e Ponto, já que armazenam ,respectivamente, qual o usuário responsável pela realização do teste e em qual ponto de coleta/entrega foi realizado o teste.

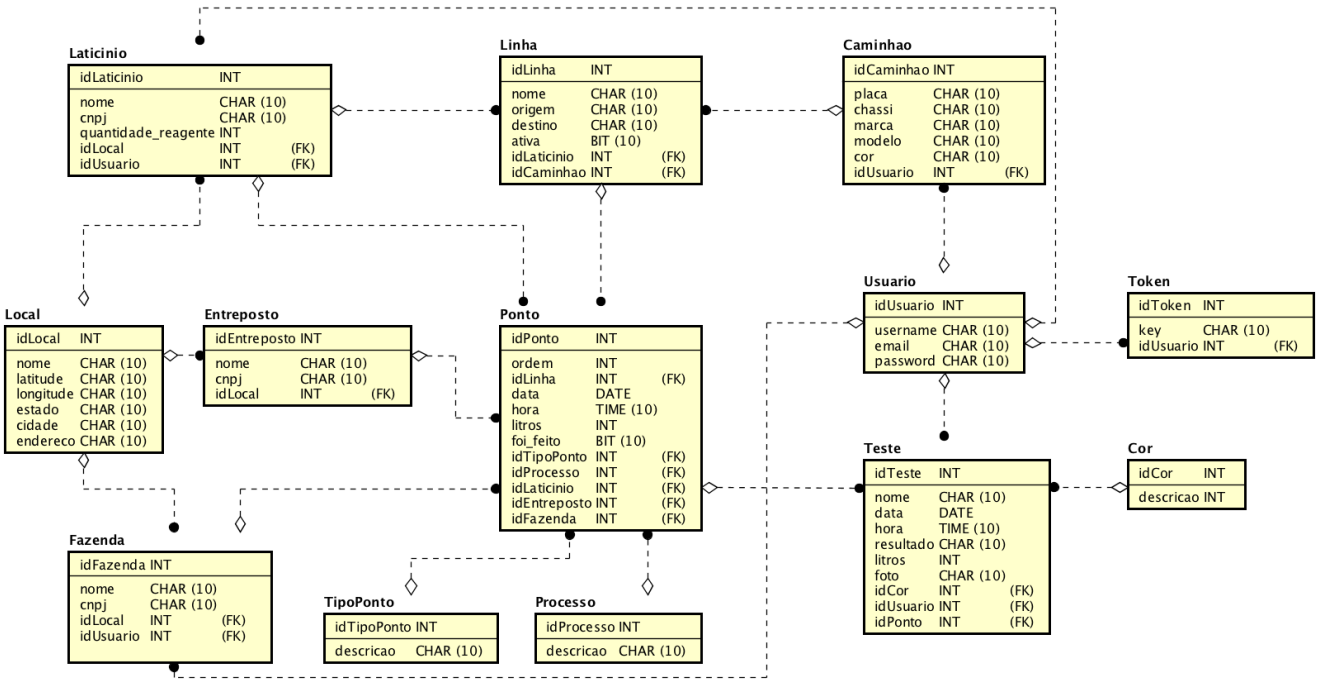

Figura 3.8: Modelo Entidade Relacionamento do Banco de dados.

3.2.4 Layout e Protótipos

Pensando no layout da aplicação e na experiência dos usuários, foram prototipadas algumas telas para a versão *mobile* e levantadas referências visuais para o desenvolvimento da versão web.

Versão Web

Para a versão web, foi utilizado um *template* de painel administrativo, o Tema Bootstrap AdminLTE [43]. Esse *template* possui várias funcionalidades visuais já implementadas, como menus, formatação de imagens, ícones, botões, formulários, gráficos, listas e tabelas, que facilitam sua utilização e permitem um visual mais agradável. A tela de referência para o visual do painel pode ser visto no Anexo III.

Versão Mobile

Focando no caminhoneiro como usuário do aplicativo, foram criadas telas para os quatro principais casos de uso desse ator:

- Realizar Login;
- Ver testes já realizados;

- Registrar um novo teste;
- Assistir vídeo tutorial do FORMFIX®.

Como pode ser visto nas Figura 3.9 e 3.10, os protótipos foram construídos para serem simples, com navegação baseada em um menu inferior com as três funcionalidades necessárias.

A tela inicial de 'Meus Testes' possui vários *cards*, cada um deles indicando um teste já realizado, com a foto e informações do teste.

A tela de 'Registrar Teste' possui campos de preenchimento básicos, a opção de adicionar foto e também o botão para salvar o teste.

Por último, a tela de 'Tutorial' com o vídeo mostrando o processo de utilização do FORMFIX®.

Figura 3.9: Telas de Login e de Meus Testes direcionadas aos Caminhoneiros/Técnicos

3.3 Construção

Ao término da fase de elaboração da solução, na qual realizou-se a modelagem de módulos, de funcionalidades, de dados, de protótipos e busca de referências visuais de *layout*, inicia-se a fase de construção do software, ou seja, a codificação.

Nessa fase de desenvolvimento de software foram utilizados alguns ambientes para alocação de código e também alguns *frameworks* que auxiliam na construção da solução.

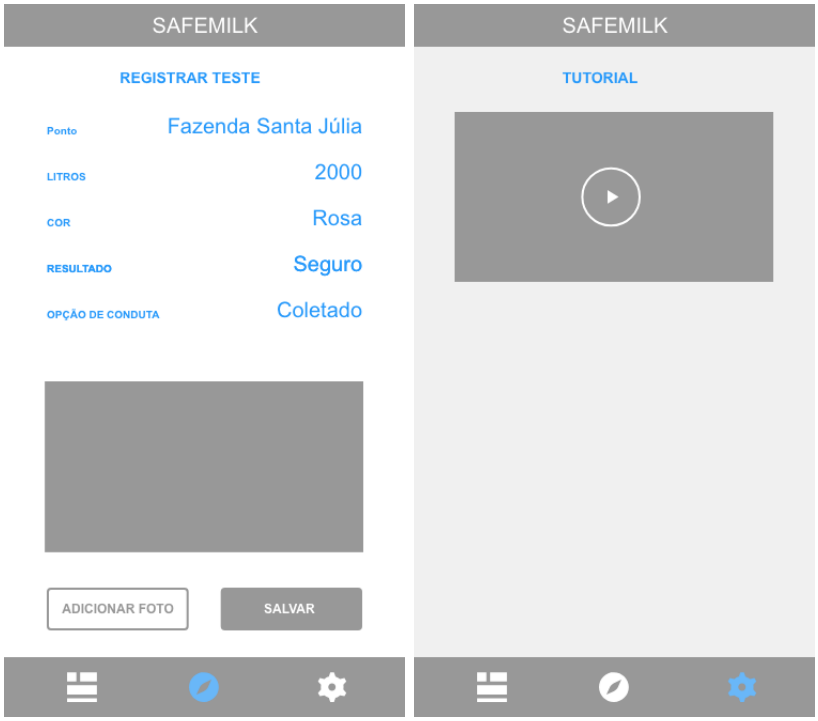

Figura 3.10: Telas de Registrar Teste e assistir Tutorial direcionadas aos Caminhoneiros/Técnicos

3.3.1 Arquitetura

Como pode ser visto na Figura 3.11, a arquitetura de alto nível da solução pode ser visualizada através dos módulos cliente e servidor. O módulo cliente contempla as plataformas utilizadas, incluindo a versão *Web* e *Mobile*. O módulo servidor apresenta o Sistema SafeMilk e o banco de dados ambos alocados em uma máquina do *Heroku*. O SafeMilk possui dois submódulos onde são processadas as requisições recebidas do módulo cliente:

- O 'Backend' para atender solicitações recebidas da versão *Web*;
- A 'REST API' para atender a requisições da versão *Mobile*.

Na sequência serão detalhados as versões *Web* e *Mobile* da arquitetura de solução.

3.3.2 Versão Web

Ambientes

Para fins de desenvolvimento foi configurado um ambiente virtual do Python versão 3.6.2, com git configurado e apontado *(heroku git:clone -a safemilk)* para um Heroku App.

Para realização de Deploy do código local do computador para o servidor do Heroku, seguem os seguintes passos:

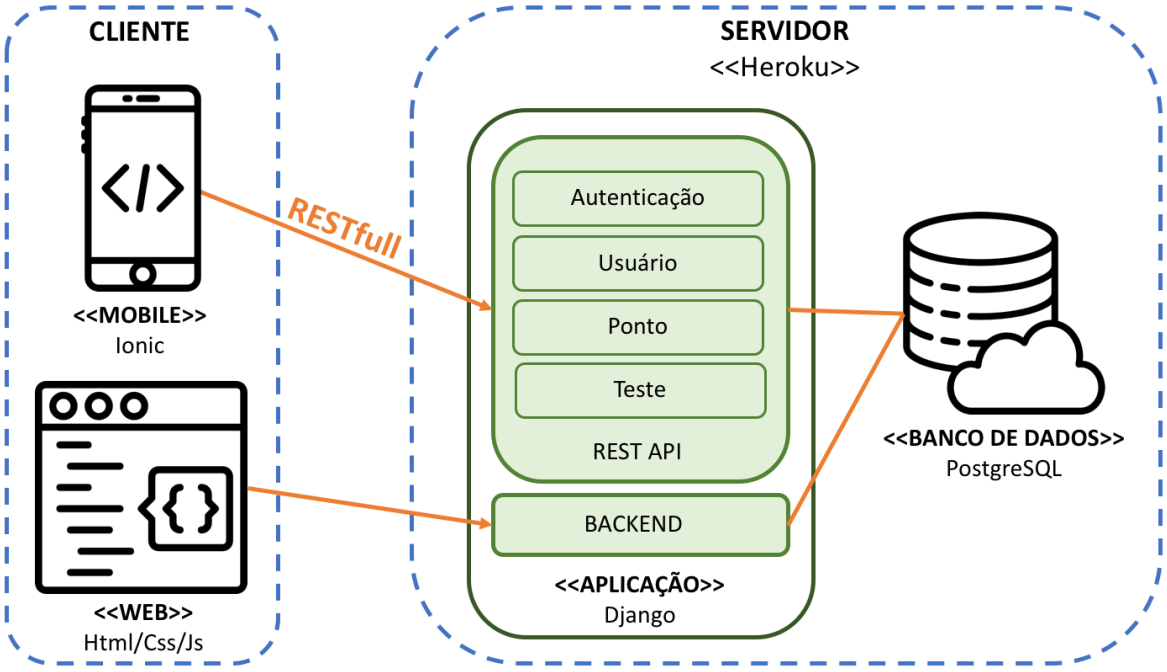

Figura 3.11: Arquitetura de alto nível do Safemilk.

1. Realizar Login

   $heroku login

2. Clonar(copiar) o repositório para a máquina local

   $ heroku git:clone -a safemilk

   $ cd safemilk

3. Subir as modificações do local para o repositório no servidor

   $ git add .

   $ git commit -am "make it better"

   $ git push heroku master

Para codificação não foi utilizado nenhum tipo de IDE (Integrated Development Environment ou Ambiente de Desenvolvimento Integrado). Foi utilizado o software *Sublime Text* [44] para edição dos códigos e o próprio aplicativo de *Terminal* do Mac para solicitações git e shell do próprio python.

Frameworks

Após o ambiente estar instalado, foi possível iniciar a programação/codificação do sistema, focando nesse primeiro momento na instalação de vários pacotes/bibliotecas correspondentes a *frameworks* utilizados, como, *Django* e *Django RESTapi*.

Segundo os módulos definidos anteriormente na etapa de Elaboração, pôde-se criar os 'Apps' da estrutura Django. Como pode ser visto na Figura 3.12, foram criados inicialmente 3 grandes Apps na arquitetura de pastas: admin (Módulo interno do Django para controle de rotas e templates), logistic (Módulo de Logística), colortest(Módulo de Testes Colorimétricos).

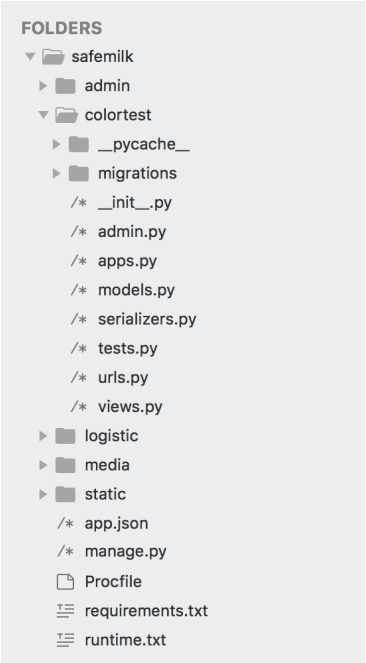

Figura 3.12: Hierarquia de Pastas com os Apps admin, logistic e colortest.

O padrão de projeto nativo do Django e utilizado no sistema foi o Model-Template-View, como já mencionado anteriormente.

Para as *Models*, o *Django* utiliza uma técnica onde os objetos/tabelas do banco de dados são automaticamente mapeadas das *Models* do sistemas. Exemplos das *models* Local e Laticínio podem ser visualizadas na Figura 3.13.

Os *Templates* basicamente são páginas HTML/CSS, que possuem código *python* embutido, na qual os dados vindos das views são renderizados.

Views utilizam as estruturas das *Models* para transportar informações entre o banco de dados e os *Templates*, tratando as antes em seu próprio código.

Para as views funcionarem, existe um listagem de rotas, na qual URL's que poderão ser solicitadas pelo usuário estão mapeadas. Esse mapeamento indica qual *views* o sistema

models.py

```python
from django.db import models
from django.utils import timezone
from django.urls import reverse
from django.contrib.auth.models import User
from admin.helper import validate_CPF, validate_CNPJ

# Create your models here.

class Local(models.Model):
    nome = models.CharField('Nome', max_length=45)
    latitude = models.CharField(max_length=45)
    longitude = models.CharField(max_length=45)
    estado = models.CharField(max_length=45)
    cidade = models.CharField(max_length=45)
    endereco = models.CharField(max_length=45)

    def __str__(self):
        return self.nome

    def get_absolute_url(self):
        return reverse('admin:list_locais')

class Laticinio(models.Model):
    nome = models.CharField(max_length=100)
    cnpj = models.CharField(unique=True, max_length=18, validators=[validate_CNPJ])
    quant_reagente = models.IntegerField(default=0)
    local = models.ForeignKey(Local, on_delete=models.CASCADE)
    proprietario = models.ForeignKey(User, on_delete=models.CASCADE)

    def __str__(self):
        return self.nome

    def get_absolute_url(self):
        return reverse('admin:list_laticinios')
        # return reverse('admin:update_laticinio', kwargs={'pk': self.pk})

```

Figura 3.13: Models do App Logistic.

deve rodar de acordo com a URL solicitada pelo usuário. Exemplos de *views* podem ser visualizadas na Figura 3.14.

views.py

```python
from django.shortcuts import render

from rest_framework import viewsets, status
from rest_framework.decorators import api_view
from rest_framework.response import Response
from rest_framework.decorators import api_view

from logistic.serializers import PontoSerializer


from django.contrib.auth.models import User
from logistic.models import Ponto, Linha, Caminhao

# Create your views here.

@api_view(['GET'])
def api_list_pontos(request):
    """
    List all pontos.
    """
    if request.method == 'GET':
        pontos = Ponto.objects.all()
        serializer = PontoSerializer(pontos, many=True)
        return Response(serializer.data)


@api_view(['POST'])
def api_list_pontos_by_user(request):
    """
    List all pontos by user.
    """
    if request.method == 'POST':

        caminhao = Caminhao.objects.filter(motorista=request.data['id']).order_by('pk')

        linhas = Linha.objects.filter(caminhao = caminhao.values('id')).order_by('pk')

        pontos = Ponto.objects.filter(linha = linhas.values_list('id')).order_by('pk')

        serializer = PontoSerializer(pontos, many=True)
        return Response(serializer.data)
```

Figura 3.14: Views do App admin.

A Figura 3.15 mostra a Home da Versão Web implementada, esta e outras telas do sistema podem ser vistas no Anexo IV deste trabalho.

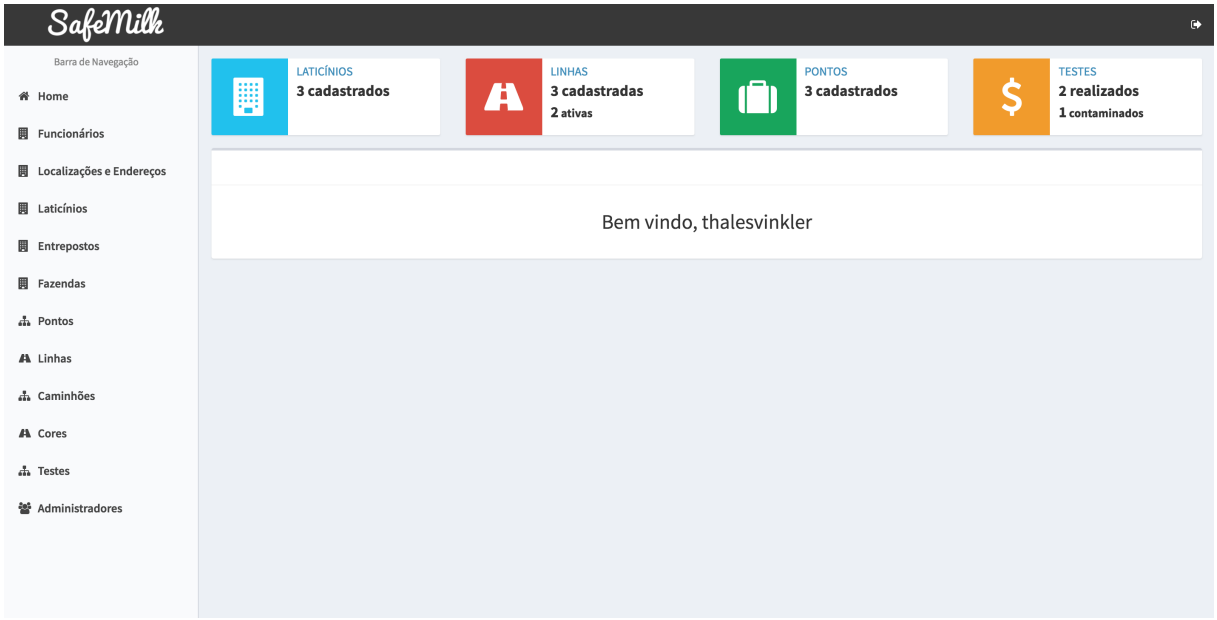

Figura 3.15: Home da versão Web implementada.

3.3.3 Versão Mobile

Ambientes

O Ionic tem seu código já pré configurado, apenas necessitando compilá-lo via terminal com diretivas do próprio sistema.

A estrutura do Ionic é baseada em *Pages*. Cada *Page* representa uma tela do aplicativo. Como pode ser visto na Figura 3.16, cada *Page* possui 3 arquivos que a descrevem: 'page.html', 'page.scss' e 'page.ts'.

No arquivo *HTML*, fica a estrutura visual da sua página, construída basicamente em *Angular2*.

O arquivo *SCSS*, representa todas as marcações de estilo do padrão *CSS* que estão presentes no *HTML*.

A principal página é a 'tela.ts', em código *TypeScrip*, por isso '.ts'. Essa página é a base para o funcionamento das outras duas estruturas. Ela funciona como uma *Controladora* onde se inicia a execução, e a partir dela saem todos os direcionamentos de fluxos, busca de dados, renderizações em tela e redirecionamentos.

A Figura 3.17 mostra as telas da Versão Mobile implementadas.

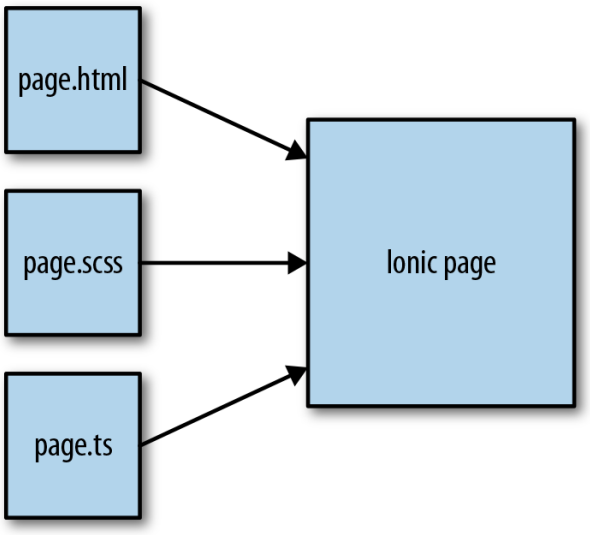

Figura 3.16: Estrutura de Tela do Ionic.

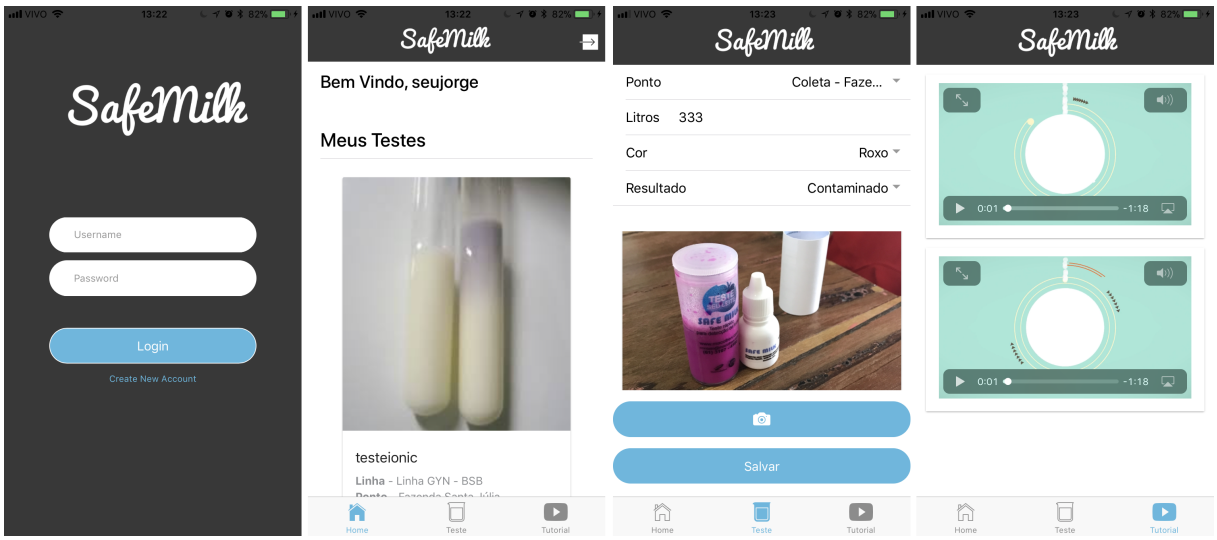

Figura 3.17: Telas da Versão Mobile implementadas.

3.4 Transição

Passada a fase de construção do solução, na qual realizou-se a programação da solução tanto na parte Web quanto na Mobile, inicia-se a fase de transição do software, ou seja, a disponibilização da solução para o usuário final.

Nessa fase de transição, foram feitos alguns movimentos de *Deploy* e de *Build* dos códigos, visando a disponibilização da versão *Web* na rede e da versão *Mobile* já no celular.

3.4.1 Versão Web

O Deploy para versão web acontece via Git, como mencionado anteriormente, sendo o código colocado diretamente no servidor do Heroku. O mesmo pode ser acessado a partir da url: https://safemilk.herokuapp.com/admin/.

3.4.2 Versão Mobile

O Ionic transforma um único código em linguagens web em dois códigos em linguagens dos sistemas operacionais Android e iOS, apenas a partir da compilação do código único.

Android

Para a versão Android, é gerado um arquivo em extenção '.apk', característico do SO por meio da diretiva:

$ ionic cordova build android –prod –release

Para instalação em um dispositivo, é necessário apenas adicionar esse arquivo à memória interna do mesmo e solicitar via Aplicativo de Pastas do próprio Android a abertura do mesmo. Com isso o aplicativo é instalado no celular e fica pronto para uso.

iOS

Para versão iOS, é gerado um projeto próprio para o software 'Xcode' do Mac, a partir da diretiva:

$ ionic cordova build ios –prod

Esse projeto, quando aberto, gera a instalação do aplicativo, ou em um software de Simulação de Iphones dentro do Mac, ou em um celular conectado ao computador. Com isso, a instação em um celular está garantida e disponível para uso e teste.

3.5 Trabalhos Correlatos

Durante a realização desse trabalho foram pesquisados outros trabalhos relacionados. Neste sentido, vão ser detalhados dois trabalhos, [45] e [46].

Em [45], foi realizado um diagnóstico da rastreabilidade no setor agroalimentar, verificando se o fato de existir rastreabilidade na cadeia produtiva gera ou não vantagem competitiva para seus produtos.

Como conclusão, entendeu-se que a rastreabilidade alinhada à uma ferramenta de gestão da qualidade e à estratégias de marketing, pode gerar vantagem competitiva e com isso melhorar a imagem que o consumidor final tem do produto. Isso vem a corroborar com o intuito desse trabalho, de gerar um solução que permita rastreabilidade e assim melhore a cadeia produtiva e a consequente melhora do produto que chega ao consumidor final.

Na área de tecnologia também já existem outras soluções com o intuito de garantir a rastreabilidade na cadeia produtiva de leite. Em [46], é proposta a resolução do problema de rastreabilidade também por meio de dispositivos móveis.

Segundo os autores, o sistema criado permite acompanhar e gerenciar a coleta e o transporte de leite entre propriedade rural e laticínio. Esse acompanhamento é feito por dados geográficos da rota percorrida.

Como podemos ver na Tabela 3.1, existem algumas diferenças e semelhanças entre os 3 trabalhos. Com relação ao trabalho [45], existe uma diferença de paradigmas, devido ao primeiro ter um foco grande em teorizar a escolha de um sistema de rastreabilidade e mostrar que sua implementação trás benefícios à organização. Já esse trabalho foca mais em colocar em prática a construção desse sistema de rastreabilidade. Sendo assim, apesar das diferenças, ambos se complementam na resolução do problema.

Existe uma semelhança muito grande entre o trabalho [46] e esse trabalho, em se tratando de objetivo e de problema a ser resolvido. A diferença está apenas na maneira como o rastreamento é feito. Enquanto [46] foca no registro via geolocalização, este trabalho foca no registros de testes colorimétricos realizados. Claramente ambas as perspectivas poderiam ser utilizadas em conjunto para resolução do problema, aumentando as funcionalidades da solução.

Tabela 3.1: Tabela Comparativa dos trabalhos correlatos

	Rastreabilidade	**Tecnologia**	**Finalidade**
Rastreabilidade da Cadeia Produtiva do Leite como Ferramenta de Diferenciação Mercadológica [45]	Incentiva rastreabilidade em toda a cadeia produtiva	Não define qual tecnologia precisa ser (Web / Mobile / Desktop), apenas determina que o sistema precisa ser acoplado à uma gestão de custos do produto	Melhorar a imagem do Leite frente ao consumidor Final, informando características relacionadas à qualidade e segurança alimentar
Uma solução para a rastreabilidade no processo de coleta e transporte do leite [46]	Via registro de geolocalização em toda cadeia produtiva	Web e Mobile	Rastrear a cadeia produtiva de leite, a fim de inibir possíveis fraudes e elevar o nível de confiabilidade deste processo
Soluções Tecnológicas na Rastreabilidade de Adulterações por Formol no Leite	Via registro de testes colorimétricos em toda cadeia produtiva	Web e Mobile	Rastreamento do leite e identificação de Adulterações por Formol em sua cadeia produtiva

Capítulo 4

Validação da Solução

Este capítulo tem como objetivo principal a realização de experimentos/testes de uso simulado e de uso real da solução. Tais testes visam examinar o comportamento do *software* através de sua execução. Para fins de processo, foram realizados testes de validação de software.

Segundo [13], a validação de um *software* pode ser definida de várias maneiras. Entretanto, uma maneira simples de se definir é: quando um *software* funciona razoavelmente como o cliente final espera, então a validação teve sucesso.

Assim, a validação visa assegurar que o software corresponda aos requisitos estabelecidos anteriormente no projeto da solução, ou seja, a validação nos permite responde à seguinte pergunta: **"Estamos construindo o produto certo?"**.

4.1 Objetivo dos Testes

Como apontado anteriormente, um problema que permeia a cadeia produtiva de leite é a falta de controle, tanto de rastreabilidade, quanto das ocorrências de adulteração do produto por formol.

Com isso, o objetivo principal dos testes é mostrar que a solução implementada atende de forma específica e favorável aos requisitos de rastreabilidade e de registro de ocorrências de adulteração por formol.

4.2 Metodologia

Conforme [13], quando um software é desenvolvido para ser utilizado como um produto por uma quantidade grande de clientes, a execução de teste formais para aceitação de cada cliente àquela solução é inviável. Nesses casos, é comum a utilização de um processo conhecido como teste *Alfa* e *Beta*.

Segundo o autor, o teste *Alfa* consiste na realização de testes conduzidos pelo desenvolvedor com um grupo de usuários finais. Assim, os testes são realizados em um ambiente controlado e natural ao desenvolvedor, que pode visualizar o cliente enquanto o mesmo utiliza a solução.

Já o teste *Beta* é conduzido em um ambiente real de utilização e mais natural ao usuário final. Ou seja, é um teste "ao vivo", ao qual o desenvolvedor não tem controle sob o ambiente.

Com o intuito de testar a solução *web* e *mobile* do *SafeMilk*, ambos os processos de testes supracitados foram utilizados e fizeram parte das etapas a seguir:

1. Planejamento
    - Revisar os requisitos (casos de uso, diagramas e modelos) criados nas fases de concepção e elaboração do Software;
    - Definir um Plano de Teste, ou seja, identificar quais são os requisitos a serem validados, por quem os testes serão realizados, em que ordem serão realizados e quais métricas/critérios serão usados para definir o sucesso de um teste.
2. Especificação
    - Definir quais serão os Casos de Teste, ou seja, para cada requisito, identificar quais são seus possíveis caminhos/cenários a serem percorridos e seus respectivos valores de entrada, restrições de execução e resultados esperados;
    - Verificar se os Casos de Teste cobrem todos os requisitos escolhidos no Plano de Teste, ou seja, nenhum cenário deixou de ser identificado.
3. Execução
    - Realizar os testes;
    - Registrar os resultados.
4. Análise
    - Analisar os resultados obtidos;
    - Verificar se os requisitos escolhidos no Plano de Teste foram validados.

4.3 Processo de Teste

Nessa seção apresentamos a realização das etapas de planejamento, especificação, execução e análise descritos na metodologia, bem como seus resultados.

4.3.1 Planejamento

Após ter sido feita a revisão de todos os requisitos do projeto, foi elaborado um Plano de Teste, com as seguintes definições:

- Requisitos a serem validados

  1. Usuários conseguem realizar login;
  2. Administradores dos Laticínios conseguem gerenciar a logística de transporte;
  3. Administradores dos Laticínios conseguem visualizar testes realizados.
  4. Caminhoneiros e Técnicos do Laticínio conseguem registrar testes;
  5. Caminhoneiros e Técnicos do Laticínio conseguem visualizar testes;
  6. Caminhoneiros e Técnicos do Laticínio conseguem visualizar tutorial do FORMFIX®;

- Responsáveis por realizar os testes

  Para os testes *Alfa* cinco usuários foram convidados a realizarem os testes em ambiente preparado pelo desenvolvedor da plataforma. Para os testes *Beta* um usuário, administrador de um laticínio familiar, utilizou a plataforma em ambiente final.

- Ordem de realização

  A realização dos testes inicia-se com o público de teste *Alfa*. Depois dessa primeira bateria de testes, seguem os testes para o público de teste *Beta*. Com ambos os públicos, foi seguido a seguinte ordem:

  1. Primeiro os Administradores dos Laticínios realizam os testes na versão *web*, pois sem o gerenciamento da logística de transporte, os Caminhoneiros e Técnicos não conseguem registrar os devidos testes.
  2. Caminhoneiros e técnicos realizam os testes na versão *mobile*.
  3. Administradores dos Laticínios testam a visualização dos dados registrados anteriormente pelos Caminhoneiros e Técnicos.

- Métricas/critérios de sucesso

  Após a realização de cada caso de teste, é verificado se o resultado do teste é igual ao resultado esperado pela especificação, de acordo com o critério de opinião de satisfação do usuário que está testando. Assim, o mesmo pode atribuir uma das duas condições ao caso de teste:

  1. A funcionalidade realiza o que ele esperava, segundo o teste. Ou seja, foi implementada de acordo com a especificação e é aceita.

2. Ou a funcionalidade não respondeu de acordo com o teste. Sendo assim, sofreu desvios no resultado, não correspondendo exatamente como a especificação.

4.3.2 Especificação

Após o planejamento, foram redigidos os casos de testes relacionados a cada funcionalidade (Caso de Uso) definida como alvo no plano de teste.

Foi construído, assim, um conjunto mínimo de cenários de teste, ou seja, nesse conjunto caso algum dos testes não for satisfeito, já não será possível a solução garantir a cobertura dos requisitos. Isso acontece pois existem algumas funcionalidades que não são essenciais ao funcionamento da solução, mas que foram implementadas com o objetivo de melhorar a experiência do usuário.

Ao total foram construídos 16 casos de teste, sendo os 11 primeiros direcionados à admnistradores do laticínio e os cinco restantes aos caminhoneiros/técnicos. Seguem alguns exemplos de casos de teste que foram redigidos:

- Caso de Uso: Usuários conseguem realizar login

  CT01 - Realizar Login na Web

  Objetivo: Realizar login no sistema web

  Pré-condições: Já estar cadastrado na plataforma

  Entrada: Nome de usuário e senha válidos

  Passos:

  1. Usuário acessa tela de login no sistema;
  2. Usuário insere o campo "Usuário";
  3. Usuário insere o campo "Senha";
  4. Usuário clica em "Entrar".

  Saída:

  1. Sistema valida os campos usuário e senha;
  2. O sistema abre a página "Home".

  CT12 - Realizar Login no App

  Objetivo: Realizar login no aplicativo mobile

  Pré-condições: Já estar cadastrado na plataforma

  Entrada: Nome de usuário e senha válidos

  Passos:

1. Usuário abre o aplicativo "SafeMilk";
2. Usuário insere o campo "Usuário";
3. Usuário insere o campo "Senha";
4. Usuário clica em "Entrar".

Saída:

1. Sistema valida os campos usuário e senha;
2. O sistema abre a página "Home".

- Caso de Uso: Administrador do Laticínio consegue gerenciar a logística de transporte

  CT03 - Adicionar Funcionário

  Objetivo: Adicionar um funcionário ao laticínio

  Pré-condições: Estar logado e na página Home

  Entrada: Informações de um funcionário

  Passos:

  1. Usuário clica no item "Funcionários"do menu lateral;
  2. Usuário clica no botão "Novo Funcionário";
  3. Usuário insere o campo "Nome do Usuário";
  4. Usuário insere o campo "Email";
  5. Usuário insere uma senha padrão no campo "Senha";
  6. Usuário insere novamente uma senha no campo "Confirmar Senha";
  7. Usuário clica em "Salvar".

  Saída:

  1. Sistema valida os campos;
  2. O sistema abre a página de listagem de funcionário já com o funcionário aparecendo na listagem.

O documento com a descrição dos casos de teste pode ser encontrado de forma integral no Anexo V deste trabalho.

4.3.3 Execução

Os testes foram executados por sete usuários diferentes, sendo cinco deles pertencentes ao grupo Alfa e dois ao grupo Beta. Cada um dos usuários do grupo Alfa percorreu os 16 casos de teste, simulando tanto o uso de um administrador de laticínio via *Web* quanto de um caminhoneiro/técnico via *App*.

Já no grupo Beta, os dois usuários reais da solução percorreram os casos de teste cada um em seu papel. O administrador do laticínio percorreu do Caso de Teste 01 (CT01) ao Caso de Teste 11 (CT11) e o caminhoneiro/técnico do CT12 ao CT16.

Os testes foram executados na ordem indicada no planejamento. Primeiro todos os testes com o público Alfa e depois os mesmos com o público Beta.

Para avaliar um determinado caso de teste, cada usuário seguiu a sequencia de passos descrita a seguir:

1. Ler o Caso de Teste, para entendimento de seu objetivo e de como alcançá-lo;
2. Realizar o passo a passo indicado no caso;
3. Verificar se a saída realizada pela solução é igual a saída requisitada pelo caso;
4. Avaliar se o teste é aceito ou não, a partir da verificação da saída ser igual ao esperado ou não.

As avalizações e comentários realizadas por cada usuário foram anotadas e quando compiladas geraram a Tabela 4.1 e a Tabela 4.2[1], respectivamente, para os públicos Alfa e Beta.

4.3.4 Análise

Após ter sido realizado o planejamento dos testes, sua especificação e também sua simulação de fato, é possível iniciar uma etapa de análise dos dados coletados durante sua execução.

Foram escolhidos, na fase de planejamento, cinco casos de uso como alvos para os testes. Como mostrado na Figura 4.1, esses casos de uso foram desdobrados em 16 casos de teste, representando cada um, uma parte da verificação de cobertura de seu requisito (caso de uso) relacionado.

O objetivo das simulações era mostrar que com a aceitação de 100% dos casos de teste redigidos, os mesmos representavam a validação dos cinco requisitos/casos de uso alvo.

[1]Os símbolos * existentes nas células simbolizam a existência de comentários do usuário durante execução do caso de teste e podem ser vistos no Anexo VI

Tabela 4.1: Resultados dos testes com usuários do grupo Alfa

		Usuário 1	**Usuário 2**	**Usuário 3**	**Usuário 4**	**Usuário 5**
Admin Laticínio	**CT01**	Aceito	Aceito	Aceito	Aceito	Aceito
	CT02	Aceito*	Aceito	Aceito*	Aceito	Aceito
	CT03	Aceito	Aceito	Aceito	Aceito	Aceito
	CT04	Aceito*	Aceito	Aceito	Aceito	Aceito*
	CT05	Aceito	Aceito*	Aceito	Aceito	Aceito
	CT06	Aceito	Aceito	Aceito	Aceito	Aceito
	CT07	Aceito	Aceito	Aceito	Aceito	Aceito
	CT08	Aceito	Aceito	Aceito*	Aceito	Aceito*
	CT09	Aceito	Aceito	Aceito	Aceito*	Aceito
	CT10	Aceito*	Aceito*	Aceito	Aceito*	Aceito*
	CT11	Aceito*	Aceito*	Aceito*	Aceito	Aceito
		Usuário 1	**Usuário 2**	**Usuário 3**	**Usuário 4**	**Usuário 5**
Caminhoneiro	**CT12**	Aceito	Aceito	Aceito	Aceito	Aceito
	CT13	Aceito	Aceito	Aceito	Aceito	Aceito
	CT14	Aceito*	Aceito*	Aceito	Aceito*	Aceito
	CT15	Aceito*	Aceito*	Aceito*	Aceito	Aceito*
	CT16	Aceito	Aceito*	Aceito	Aceito	Aceito

Tabela 4.2: Resultados dos testes com usuários do grupo Beta

		Usuário 1
Admin Laticínio	**CT01**	Aceito
	CT02	Aceito*
	CT03	Aceito
	CT04	Aceito*
	CT05	Aceito*
	CT06	Aceito
	CT07	Aceito
	CT08	Aceito*
	CT09	Aceito*
	CT10	Aceito*
	CT11	Aceito*
		Usuário 2
Caminhoneiro	**CT12**	Aceito
	CT13	Aceito
	CT14	Aceito*
	CT15	Aceito*
	CT16	Aceito*

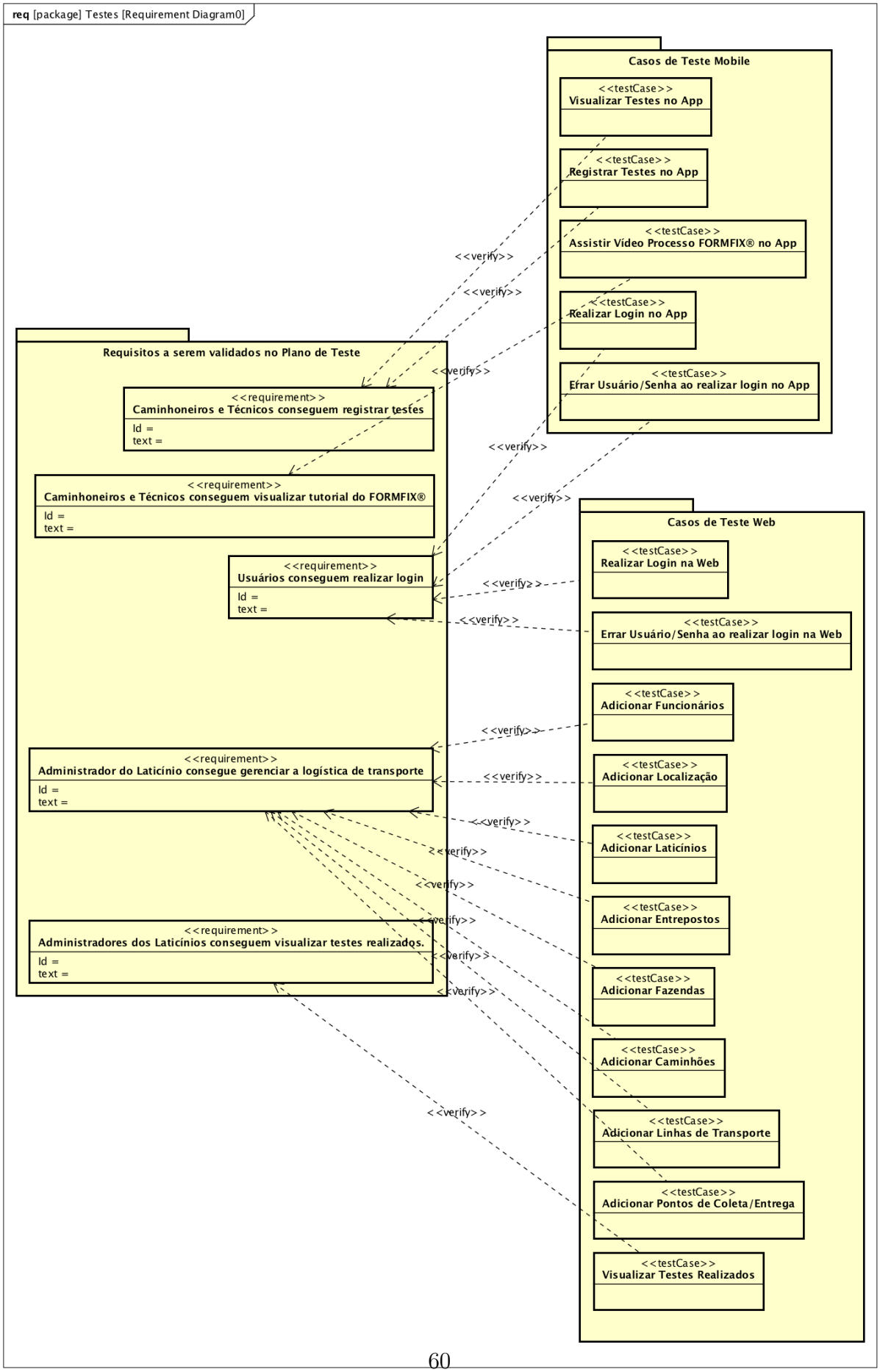

Figura 4.1: Diagrama de Requisitos e Casos de Teste.

Como mostrado na Tabela 4.1 e na Tabela 4.2, 100% dos casos de teste foram aceitos pelos usuários, tanto do grupo Alfa quanto do grupo Beta. Com isso confirmamos que os cinco requisitos alvo foram implementados.

4.4 Análise dos Resultados

O objetivo principal dos testes, era mostrar que a solução implementada atendia aos requisitos, ou seja, um produto que segue os requisitos estabelecidos e, que por consequência, resolve os problemas de rastreabilidade e de registro de ocorrências de adulteração por formol.

Como mostra a Figura 4.2, para alcançar tal objetivo, era preciso mostrar que os requisitos dos sistemas tinham sido implementados e, que por sua vez, precisava que seus casos de teste fossem aceitos.

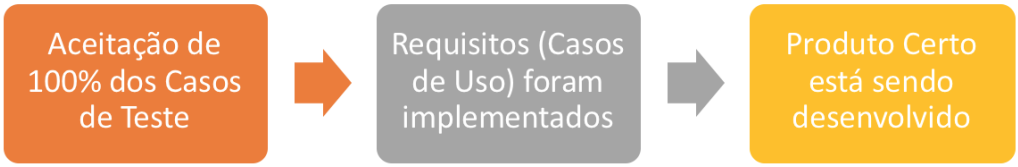

Figura 4.2: Diagrama Causal de Validação do SafeMilk.

Como 100% dos casos de teste foram aceitos pelos usuários dos grupos Alfa e Beta e, por consequência, os cinco requisitos alvos foram validados com sucesso, pode-se dizer que a solução atende aos requisitos estabelecidos.

Apesar de seu desenvolvimento estar sendo correto, existem ainda muitas evoluções que podem ser feitas, principalmente a partir das observações coletadas durante os testes com os usuários. Evoluções estas que são tanto melhorias de usabilidade para o usuário quanto novas funcionalidades para aumentar a robustez da solução.

Capítulo 5

Conclusões

Este trabalho tratou do projeto, construção e validação de uma solução *Web* e *Mobile*, chamada SafeMilk, para solucionar o problema de controle de rastreabilidade e de adulterações por formol na cadeia produtiva de leite. A solução foi projetada como ferramenta de auxilio junto ao uso de um reagente químico detector de adulterações por formol via teste colorimétrico, chamado FORMFIX®. O levantamento de requisitos foi realizado junto a especialistas nas áreas de leite e também da empresa responsável por esse reagente.

A solução foi pensada para dois públicos alvos principais da cadeia produtiva, o administrador do Laticínio, responsável por gerenciar toda a logística de transporte do laticínio, e o caminhoneiro/técnico, responsáveis pela realização dos testes colorimétricos com o FORMFIX®.

Para os administradores do laticínio, foi desenvolvido um sistema *Web* que permite gerenciar a logística de transporte do laticínio inserindo, por exemplo, quais rotas são percorridas, quais fazendas e entrepostos são visitados nessas rotas, qual caminhão e qual motorista é responsável por cada rota. Foi utilizado como linguagem de desenvolvimento, o *Python* com *framework Django* e banco de dados *PostgreSQL*.

Para os caminhoneiros e técnicos dos laticínio, foi desenvolvido um aplicativo *mobile*. Esse aplicativo tinha o objetivo de permitir que, após realização do processo de teste do leite para detecção de formol com o reagente FORMFIX®, fosse feito o registro desse teste inserindo informações, como, quantidade de litros testada, hora e data do teste, local de realização do teste, resultado obtido e também foto do resultado colorimétrico.

Foi possível, com a realização dos testes de validação da solução, compreender que o software, apesar das evoluções sugeridas pelos usuários testadores, atende às suas expectativas e desempenha a funções primordiais que eram necessárias:

- Gerenciar a logística de transporte do laticínio;
- Registrar testes colorimétricos;

- Permitir visualização do tutorial de utilização do FORMFIX®.

Por fim, entendemos que a implementação dessas funções primordiais já foram suficientes para reduzir o problema de controle da rastreabilidade do leite e das adulterações por formol, pois:

- O registro dos testes colorimétricos via App permite que os testes sejam gravados em uma base de dados e futuramente fiscalizados, caso tenha sido encontrada uma adulteração.
- Uma logística de transporte definida permite a rastreabilidade do local onde o teste com resultado contaminado foi realizado.

Trabalhos Futuros

A solução desenvolvida permite tanto evoluções relacionadas a adição/manutenção de funcionalidades quanto na criação de novas estruturas/aplicações que a utilizem como ponto de partida.

Focando inicialmente em funcionalidades, foram desenvolvidas as primordiais para que a solução pudesse resolver o problema atacado. Entretanto outras funcionalidades que permitem impulsionar o desempenho da solução podem ser implementadas de maneira evolutiva.

Na atual solução, a escolha do resultado do teste colorimétrico é realizado pelo próprio usuário do aplicativo no momento do registro do teste. Entretanto, seria de grande valia se o software, ao tirar a foto, conseguisse analisar a imagem e apontar qual a coloração do teste presente na mesma e assim o resultado daquele teste colorimétrico. Além de ajudar o usuário na escolha do resultado dos casos mais duvidosos também pode ser utilizado como uma maneira de avisar os administradores do laticínio em possíveis tentativas de fralde do caminhoneiro/técnico, ao colocar um resultado diferente do encontrado no teste colorimétrico.

Outra funcionalidade atual que pode ser evoluída é a de localização das propriedades, sejam elas fazendas, entrepostos ou os próprios laticínios. Na atual solução, a localização é feita por meio de campos de texto contendo o endereço e latitude/longitude das mesmas. Entretanto, uma integração com algum sistema de geolocalização tanto para a *Web* quanto para o *App* seriam de grande valia, visto que poderia permitir a visualização da localização em um mapa na tela do usuário e também a identificação automática via *GPS* do celular ou do navegador do computador.

Pensando em novas soluções relacionadas a este trabalho, podemos idealizar a criação de um portal integrado à *API* do sistema *Web* que tenha como objetivo conectar ambos à

alguma ferramenta do Ministério da Agricultura. Isso permitiria que atores do Ministério pudessem ter informações mais operacionais, como, relatório de todos os testes realizados e também algumas informações mais estratégicas, como, propriedades com mais focos de adulteração, principais linhas em que essas adulterações ocorrem, entre outras.

Referências

[1] Milk, Ideas for: *Oportunidades para startups*. Publicacao Online. Disponivel em: http://www.ideasformilk.com.br/desafio-startups/conteudo/oportunidades-para-startups Acessado em: 17/11/2017. ix, 24, 25

[2] Sousa, Rander Esteves: *Avaliação de um teste rápido para a detecção de formol no leite cru*. Trabalho de conclusão de curso, faculdade de agronomia e medicina veterinária, Universidade de Brasília - UnB, Brasília, DF, 2016. ix, 27

[3] Estatística, IBGE Instituto Brasileiro de Geografia e: *Indicadores ibge - estatística da produção pecuária*. Publicacao Online. Disponivel em: https://biblioteca.ibge.gov.br/visualizacao/periodicos/2380/epp_2017_dez.pdf Acessado em: 06/02/2018. 1

[4] Zoccal, Rosângela: *Alguns números do leite*. Publicacao Online. Disponivel em: http://www.baldebranco.com.br/alguns-numeros-do-leite/ Acessado em: 06/02/2018. 1

[5] Rural, Canal: *Operação leite compen$ado cumpre mandados de prisão no rs*. Publicacao Online. Disponivel em: http://www.canalrural.com.br/noticias/leite/operacao-leite-compenado-cumpre-mandados-prisao-66498 Acessado em: 06/02/2018. 1

[6] Vigilância Sanitária, ANVISA Agência Nacional de: *Esclarecimentos sobre os riscos à saúde das substâncias ureia e formol e sua adição ao leite*. Publicacao Online. Disponivel em: http://www.ebc.com.br/noticias/saude/2013/05/consumo-de-leite-com-formol-nao-e-seguro-alerta-anvisa Acessado em: 06/02/2018. 1

[7] Tecnologias Químicas, Macofren: *Formfix*. Publicacao Online. Disponivel em: http://macofren.com/laticinios/formfix/ Acessado em: 17/11/2017. 5, 26

[8] LAUDON, K. e LAUDON, J: *Sistemas de Informação Gerenciais*, volume 9ª ed. Pearson, São Paulo, 2011. 5, 6

[9] Management Group, Object: *Bpmn resource page*. Publicacao Online. Disponivel em: http://www.omg.org/bpmn/ Acessado em: 17/11/2017. 6

[10] Guedes, G.T.A.: *UML 2 - Uma Abordagem Prática - 1ª Edição:*. NOVATEC, 2011, ISBN 9788575221938. 9

[11] Wazlawick, Raul Sidnei: *Análise e Projeto de Sistemas de Informação Orientados a Objetos*, volume 2.ed. Elsevier, Rio de Janeiro, 2011, ISBN 9788535239164. 10, 15, 16, 28

[12] Grady Booch, James Rumbaugh, Ivar Jacobson: *UML: guia do usuário*, volume 2a edição. Elsevier, Rio de Janeiro, 2005, ISBN 9788535217841. 10, 11, 12, 15

[13] Pressman, Roger S.: *Engenharia de Software: Uma abordagem profissional*, volume 7.ed. AMGH, Porto Alegre, 2011, ISBN 9788580550443. 13, 53

[14] Siqueira, Professor Fernando De: *4 - modelo entidade e relacionamentos*. Publicacao Online. Disponivel em: `https://sites.google.com/site/uniplibancodedados1/aulas/aula-4---modelo-entidade-e-relacionamentos` Acessado em: 18/11/2017. 13

[15] Lutz, Mark: *Learning Python*. 2003, ISBN 0596002815. 17

[16] Brito, Thiago Guimarães: *Porque aprender python pode te levar mais longe na carreira!* Publicacao Online. Disponivel em: `https://becode.com.br/porque-aprender-python/` Acessado em: 18/11/2017. 17

[17] Chun, Wesley: *Core Python Programming*, volume 2nd ed. Pearson Education, Inc, Upper Saddle River, NJ, 2007, ISBN 0132269937. 17

[18] Software Foundation, Python: *Our documentation*. Publicacao Online. Disponivel em: `https://www.python.org/doc/` Acessado em: 17/11/2017. 17

[19] Software BV, TIOBE: *The python programming language*. Publicacao Online. Disponivel em: `https://www.tiobe.com/tiobe-index/python/` Acessado em: 17/11/2017. 17

[20] Programming Language, PYPL PopularitY of: *Pypl index*. Publicacao Online. Disponivel em: `http://pypl.github.io/PYPL.html` Acessado em: 17/11/2017. 18

[21] TechBeacon: *Programming language rankings: Which ones matter?* Publicacao Online. Disponivel em: `https://techbeacon.com/programming-language-rankings-which-ones-matter` Acessado em: 17/11/2017. 18

[22] Software Foundation, Django: *Django*. Publicacao Online. Disponivel em: `https://www.djangoproject.com/` Acessado em: 17/11/2017. 18

[23] Programming Language, Python: *Django*. Publicacao Online. Disponivel em: `https://pythonprogramminglanguage.com/django/` Acessado em: 17/11/2017. 19

[24] Book, The Django: *The model-view-controller design pattern*. Publicacao Online. Disponivel em: `https://djangobook.com/model-view-controller-design-pattern/` Acessado em: 20/12/2017. 19

[25] Global Development Group, The PostgreSQL: *About*. Publicacao Online. Disponivel em: `https://www.postgresql.org/about/` Acessado em: 17/11/2017. 20

[26] Python, Full Stack: *Postgresql*. Publicacao Online. Disponivel em: `https://www.fullstackpython.com/postgresql.html` Acessado em: 17/11/2017. 20

[27] Heroku: *What*. Publicacao Online. Disponivel em: `https://www.heroku.com/what` Acessado em: 17/11/2017. 20

[28] Conservancy, Software Freedom: *Git*. Publicacao Online. Disponivel em: `https://git-scm.com/` Acessado em: 17/11/2017. 21

[29] Atlassian: *What is git*. Publicacao Online. Disponivel em: `https://www.atlassian.com/git/tutorials/what-is-git` Acessado em: 17/11/2017. 21

[30] Rezener, Beatriz e Britto, João Gabriel: *Estudo comparativo entre o desenvolvimento de aplicativos móveis utilizando abordagem nativa e multiplataforma*. Trabalho de conclusão de curso, faculdade unb gama - fga, Universidade de Brasília - UnB, Brasília, DF, 2016. 21, 22

[31] Framework, Ionic: *About us*. Publicacao Online. Disponivel em: `https://ionicframework.com/` Acessado em: 17/11/2017. 22

[32] Angular: *One framework. mobile & desktop.* Publicacao Online. Disponivel em: `https://angular.io/` Acessado em: 27/12/2017. 22

[33] Network, Mozilla Developers: *O que é javascript?* Publicacao Online. Disponivel em: `https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/About_JavaScript` Acessado em: 27/12/2017. 22

[34] Microsoft: *Typescript*. Publicacao Online. Disponivel em: `https://www.typescriptlang.org/` Acessado em: 27/12/2017. 22

[35] Consortium, World Wide Web: *Html*. Publicacao Online. Disponivel em: `https://www.w3.org/html/` Acessado em: 27/12/2017. 22

[36] Consortium, World Wide Web: *Cascading style sheets*. Publicacao Online. Disponivel em: `https://www.w3.org/Style/CSS/` Acessado em: 27/12/2017. 22

[37] Valsechi, Prof. Dr. Octávio Antônio: *O leite e seus derivados*. Tese de Mestrado, Universidade Federal de São Carlos, Araras, SP, 2001. 24

[38] Tonaco, Adriano Scarpa, Antônio Augusto Melo Malard e Arthur Tôrres Filho: *Guia Técnico Ambiental da Indústria de Laticínios*. FIEMG - Federação das Indústrias do Estado de Minas Gerais, Belo Horizonte, 2014. 24

[39] BRASIL, Instrução Normativa nº62 de 29 de dezembro de 2011: *Aprova os regulamentos técnicos de produção, identidade e qualidade do leite*. Diário Oficial da União, Brasília, 2011. 26

[40] BRASIL, Instrução Normativa nº68 de 14 de dezembro de 2006: *Métodos analíticos oficiais físico-químicos para controle de leite e produtos lácteos*. Diário Oficial da União, Brasília, 2006. 26

[41] Pinheiro, Priscila Souza: *Estudo sobre a influência do tempo de armazenamento e da temperatura na detecção de formol adicionado ao leite*. Trabalho de conclusão de curso, faculdade de agronomia e medicina veterinária, Universidade de Brasília - UnB, Brasília, DF, 2016. 26, 27

[42] Tecnologias Químicas, Macofren: *Macofren*. Publicacao Online. Disponivel em: `http://macofren.com/` Acessado em: 17/11/2017. 26

[43] Almsaeed, Abdullah: *Adminlte control panel template*. Publicacao Online. Disponivel em: `https://adminlte.io/` Acessado em: 06/01/2018. 41

[44] HQ Pty Ltd, Sublime: *A sophisticated text editor for code, markup and prose*. Publicacao Online. Disponivel em: `https://www.sublimetext.com/` Acessado em: 06/01/2018. 44

[45] Winck, César e Morgan, Andressa: *Rastreabilidade da cadeia produtiva do leite como ferramenta de diferenciação mercadológica*. Revista de Administração de Roraima - UFRR, 6(2):430–449, 2016. 51, 52

[46] Rambo, Fernando e Maciel, José Maurício Carré: *Uma solução para a rastreabilidade no processo de coleta e transporte do leite*. Instituto de ciências exatas e geociências, Universidade de Passo Fundo, Passo Fundo, RS, 2015. 51, 52

Anexo I

Modelagem BPMN

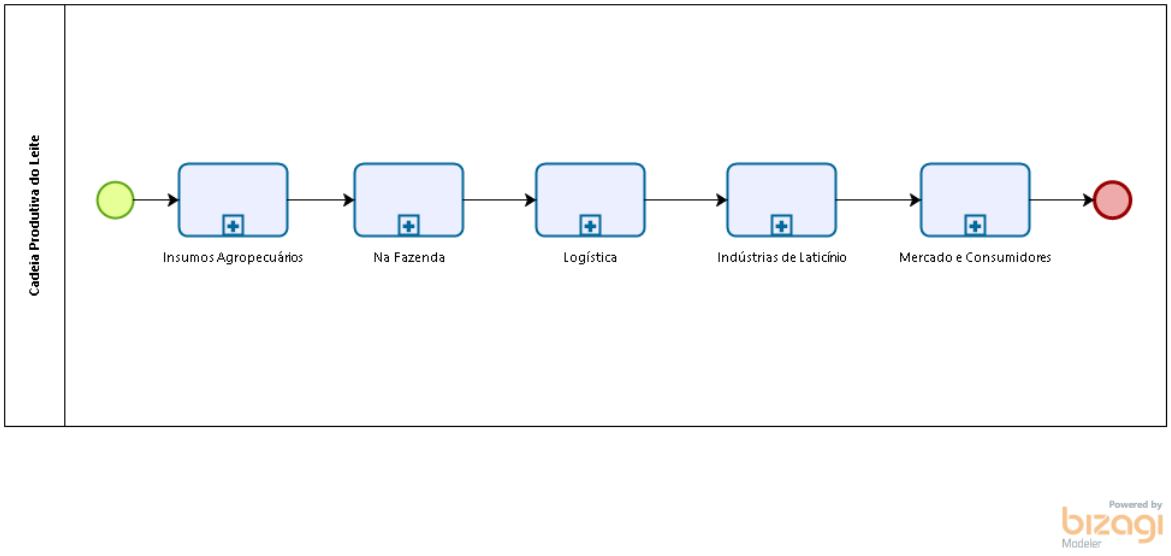

Figura I.1: Modelagem da Cadeia Produtiva de Leite.

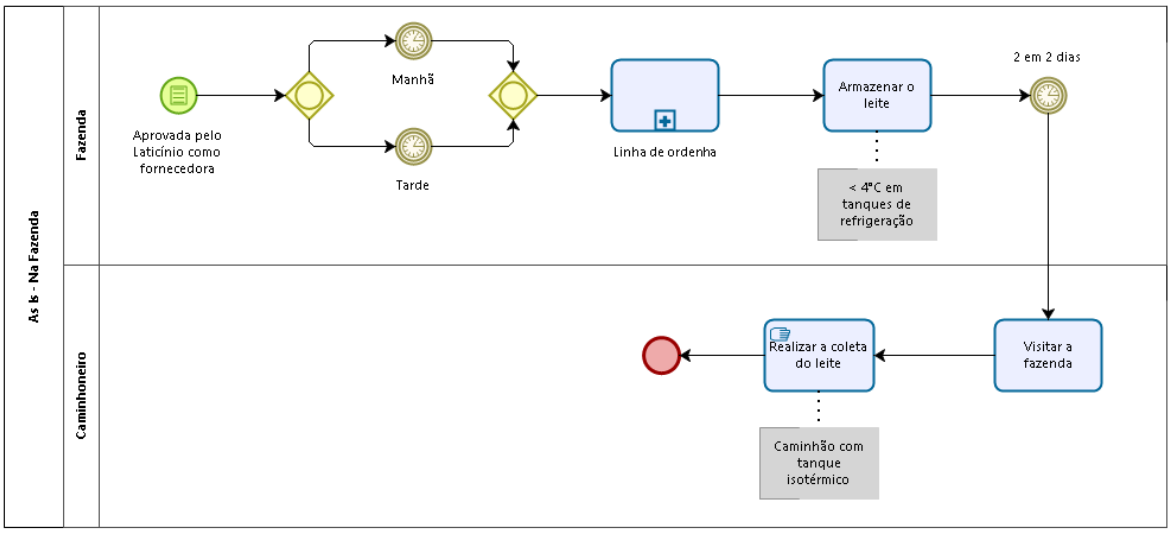

Figura I.2: Modelagem As-Is do processo na Fazenda.

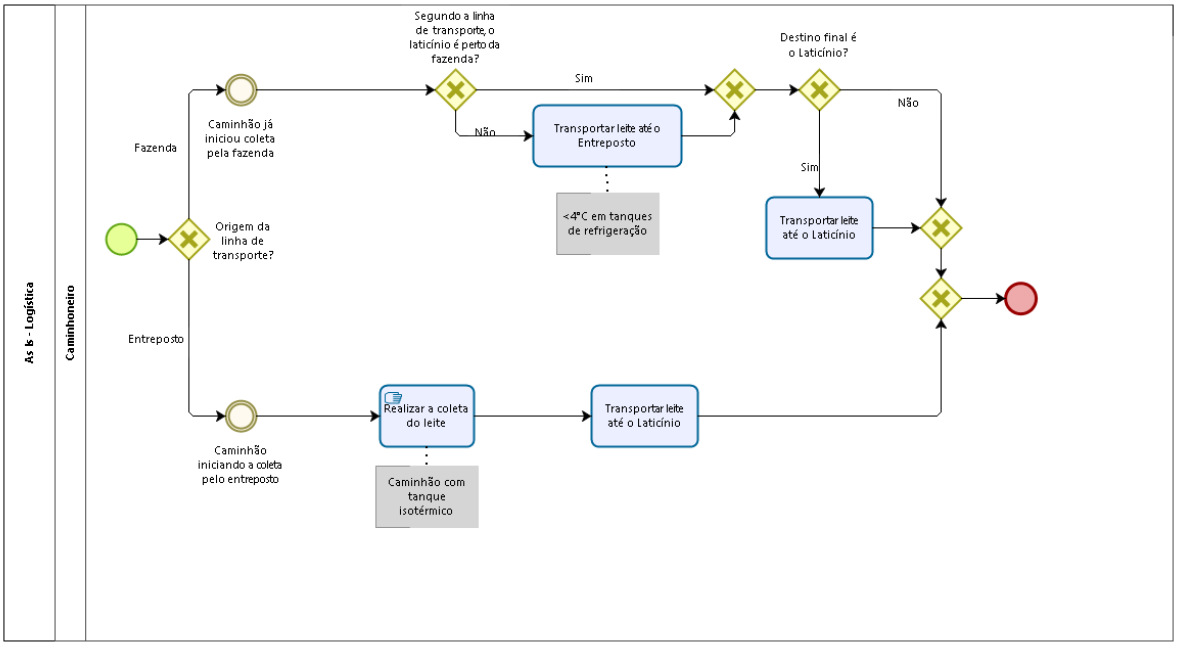

Figura I.3: Modelagem As-Is do processo na Logística.

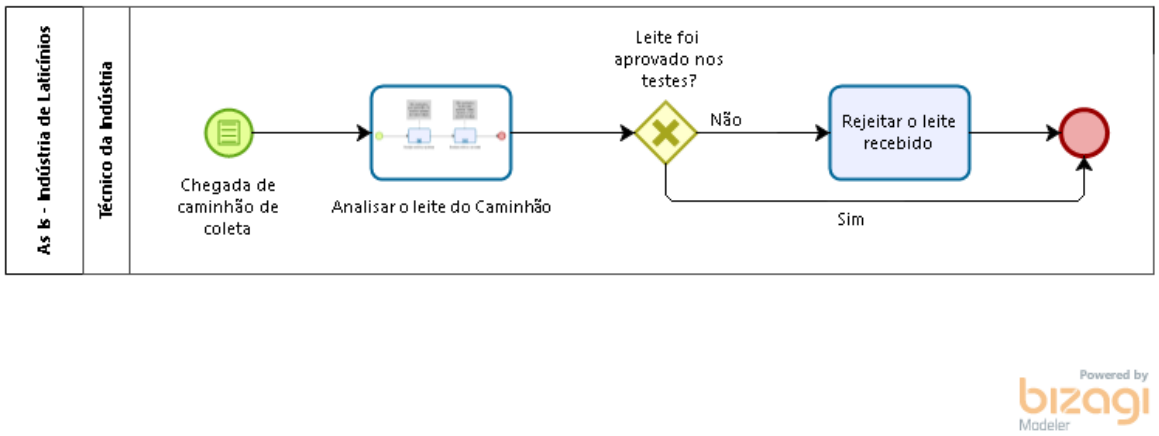

Figura I.4: Modelagem As-Is do processo na Indústria de Laticínios.

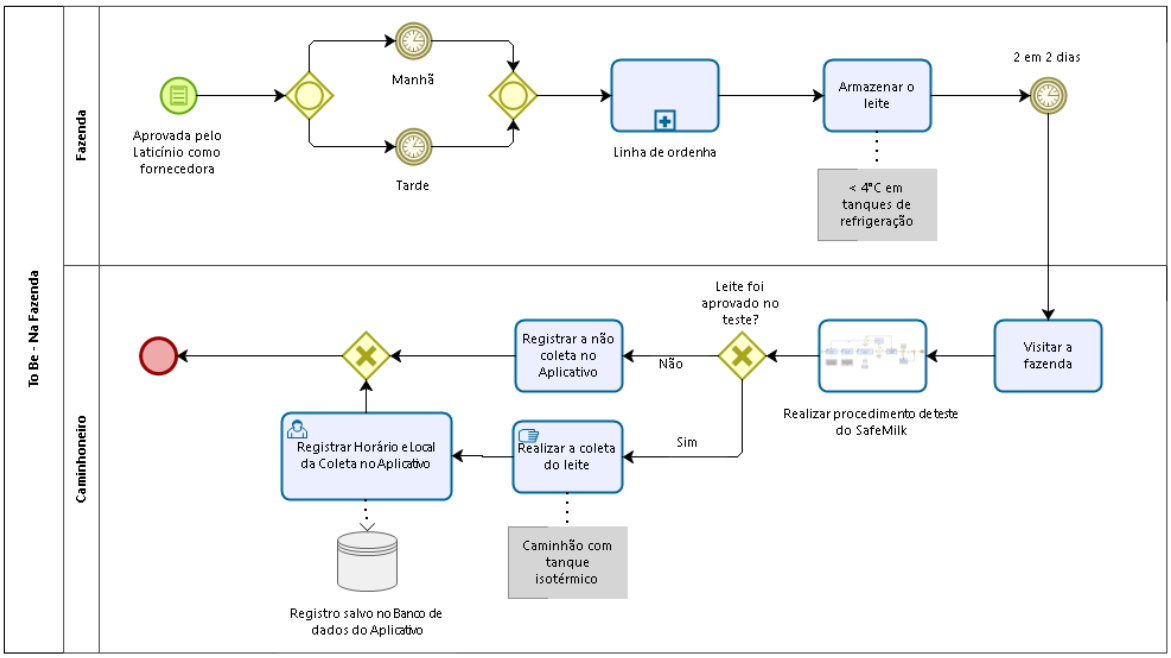

Figura I.5: Modelagem To-Be do processo na Fazenda.

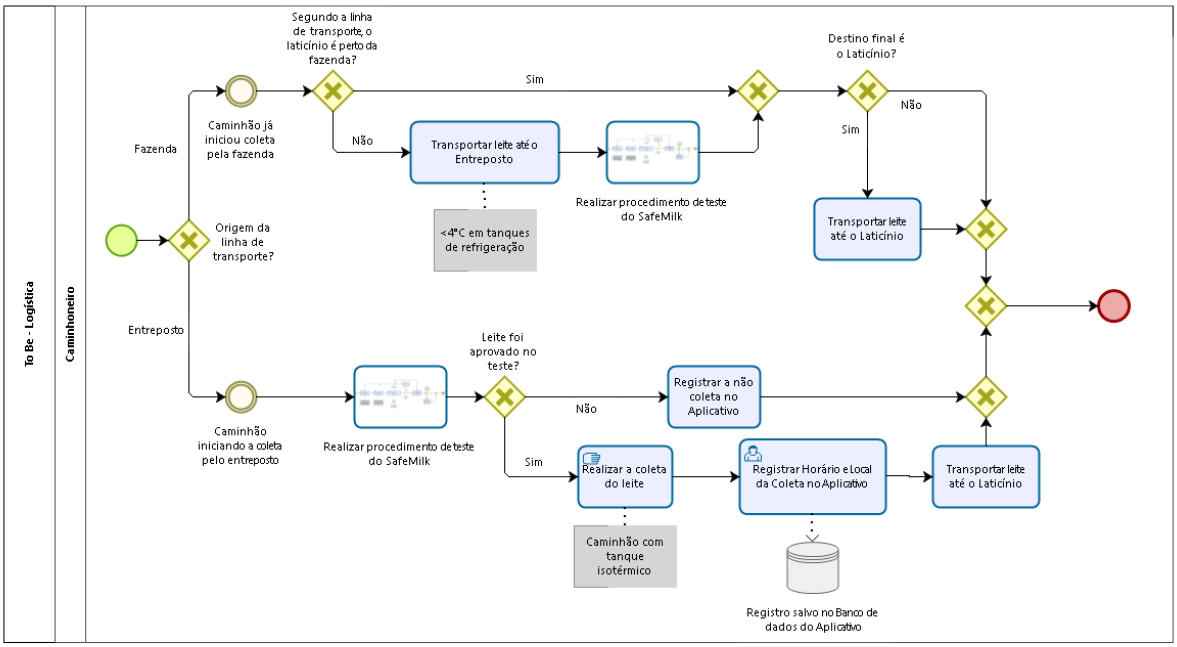

Figura I.6: Modelagem To-Be do processo na Logística.

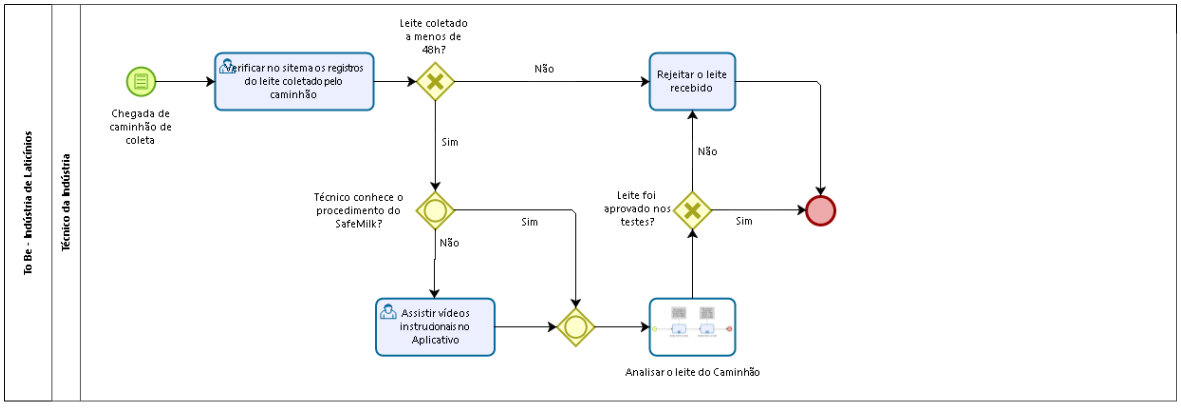

Figura I.7: Modelagem To-Be do processo na Indústria de Laticínios.

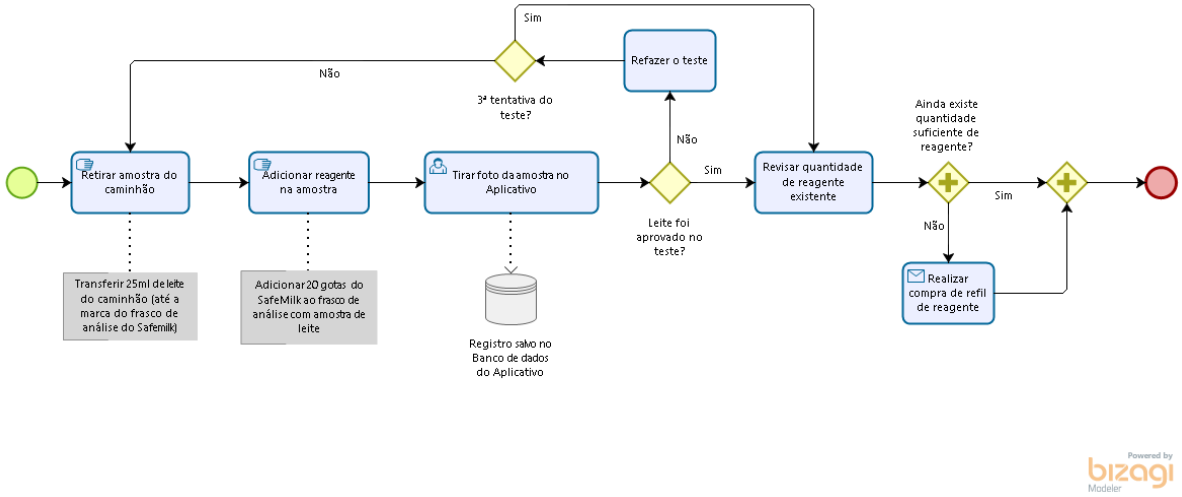

Figura I.8: Modelagem do procedimento de teste do FORMFIX®.

Anexo II

Documento de Requisitos

Soluções Tecnológicas para Controle de Qualidade e Rastreabilidade na Cadeia Produtiva de Leite

Safemilk
Levantamento de Requisitos

Versão: Versão 01.00
Data: 27 de janeiro, 2017

Histórico de revisões do modelo

Versão	Data	Autor	Descrição	Localização
00.01	15/AGO/2016	Thales Vinkler	Versão inicial: entrevista com especialistas do FORMFIX®	
00.02	07/DEZ/2016	Thales Vinkler	Entrevista com especialista em leite	
01.00	27/JAN/2017	Thales Vinkler	Versão Final: Ajustes para finalização do documento	

Aprovadores

Nome	**Função**
Thales Vinkler	Gerente de Projeto, Arquiteto de Software e Desenvolvedor
Profª Drª Marcia Aguiar Ferreira	Faculdade de Agronomia e Medicina Veterinária
Renato Santana	Responsável Comercial da Macofren Tecnologias Químicas
Arilson Onesio	Responsável Técnico da Macofren Tecnologias Químicas

Índice

1. Introdução

O objetivo da solução é permitir o registro de testes colorimétricos realizados via FORMFIX®, fazendo com que seja possível gerenciar o caminho percorrido pelo leite até chegar ao laticínio e encontrar onde estão ocorrendo as adulterações por formol.

2. Stakeholders

2.1. Produtor

Proprietário da fazenda e responsáveis pela produção leiteira.

2.2. Caminhoneiros

Responsáveis por coletar o leite nas fazendas e entrega-lo em entrepostos ou nos laticínios.

2.3. Técnico do Laticínio

Responsável por receber o leite dos caminhões no laticínio e realizar alguns testes para verificação da qualidade do mesmo.

2.4. Auditor do Ministério (Fiscal)

Responsável por fiscalizar os laticínios.

2.5. Administrador do Laticínio

Responsável pelo laticínio e por seu funcionamento.

3. Requisitos Funcionais

3.1. <RF01> Todo usuário precisa/deve realizar login

Usuário realiza login na solução com dados de usuário e senha.

3.2. <RF02> Produtores visualizam os testes registrados

Produtores conseguem visualizar os testes que foram registrados pelos Caminhoneiros em sua propriedade

3.3. <RF03> Caminhoneiros registram testes

Caminhoneiros conseguem realizar os testes do FORMFIX® na coleta/entrega de algum caminhão com leite e registrar esse teste na solução

3.4. <RF04> Caminhoneiros assistem vídeos tutoriais do procedimento do FORMFIX®

Caminhoneiros conseguem assistir pela solução vídeos tutoriais indicando o processo do FORMFIX®

3.5. <RF05> Técnicos registram testes

Técnicos conseguem realizar os testes do FORMFIX® na chegada de algum caminhão com leite ao laticínio e registrar esse teste na solução.

3.6. <RF06> Técnicos assistem vídeos tutoriais do procedimento do FORMFIX®

Técnicos conseguem assistir pela solução vídeos tutoriais indicando o processo do FORMFIX®

3.7. <RF07> Auditores do Ministério visualizam os testes registrados

Auditores conseguem visualizar os testes que foram registrados pelos Caminhoneiros e Técnicos

3.8. <RF08> Administradores dos Laticínios visualizam os testes registrados

Administradores dos Laticínios conseguem visualizar os testes que foram registrados pelos Caminhoneiros e Técnicos

3.9. <RF09> Administradores dos Laticínios adicionam funcionários

Administradores dos Laticínios conseguem adicionar novos funcionários.

3.10. <RF10> Administradores dos Laticínios adicionam localizações

Administradores dos Laticínios conseguem adicionar novas localizações.

3.11. <RF11> Administradores dos Laticínios adicionam caminhões

Administradores dos Laticínios conseguem adicionar novos caminhões.

3.12. <RF12> Administradores dos Laticínios adicionam fazendas

Administradores dos Laticínios conseguem adicionar novas fazendas.

3.13. <RF13> Administradores dos Laticínios adicionam entrepostos

Administradores dos Laticínios conseguem adicionar novos entrepostos.

3.14. <RF14> Administradores dos Laticínios adicionam laticínios

Administradores dos Laticínios conseguem adicionar novos laticínios.

3.15. <RF15> Administradores dos Laticínios adicionam linhas de transporte

Administradores dos Laticínios conseguem adicionar novas linhas de transporte.

3.16. <RF16> Administradores dos Laticínios adicionam pontos de coleta/entrega

Administradores dos Laticínios conseguem adicionar novos pontos de coleta/entrega.

4. Requisitos Não Funcionais

4.1. <RNF01> Devem ser registrados até três testes por ponto de coleta/entrega

Caso o primeiro teste colorimétrico acuse presença de formol, o usuário poderá realizar mais até mais 2 outros testes para verificar o resultado obtido no primeiro teste.

4.2. <RNF02> A interface e usabilidade do Aplicativo deve ser simples e intuitiva para pessoas tecnologicamente limitadas

Devido ao perfil dos usuários serem de pessoas com mais idade e tecnologicamente pouco familiarizadas com celulares, aplicativos e sistemas web, se faz necessário a criação de uma interface simplificada, com fluxos pequenos e de fácil manipulação.

4.3. <RNF03> Produtores só podem visualizar testes registrados em sua fazenda

Por questões de Segurança, um produtor não pode ter acesso à dados de testes realizados em outras fazendas.

4.4. <RNF04> Caminhoneiros e Técnicos só podem visualizar testes registrados pelo próprio usuário

Por questões de Segurança, um caminhoneiro não pode ter acesso à dados de testes realizados por outros usuários.

4.5. <RNF05> Administradores dos Laticínios e Auditores podem visualizar testes registrados por todos os usuários do seu laticínio

Por questões de Segurança, o administrador pode visualizar todos os testes de seu laticínio.

4.6. <RNF06> Cada caminhão possui apenas um motorista

4.7. <RNF07> Cada Laticínio possui apenas uma localização

4.8. <RNF08> Cada Entreposto possui apenas uma localização

4.9. <RNF09> Cada Fazenda possui apenas uma localização

4.10. <RNF10> Uma Linha de transporte pode conter vários pontos de coleta/entrega

4.11. <RNF11> Em uma Linha de transporte existe uma ordem de pontos de coleta/entrega que deve ser seguida pelo caminhoneiro

4.12. <RNF12> Um ponto de coleta/entrega pode estar presente em mais de uma linha de transporte

4.13. <RNF13> Cada teste está relacionado a apenas um ponto de coleta/entrega

Anexo III

Referência de Layout para o Painel Administrativo

Figura III.1: Tela de referência de layout do AdminLTE.

Anexo IV

Telas do Sistema Web

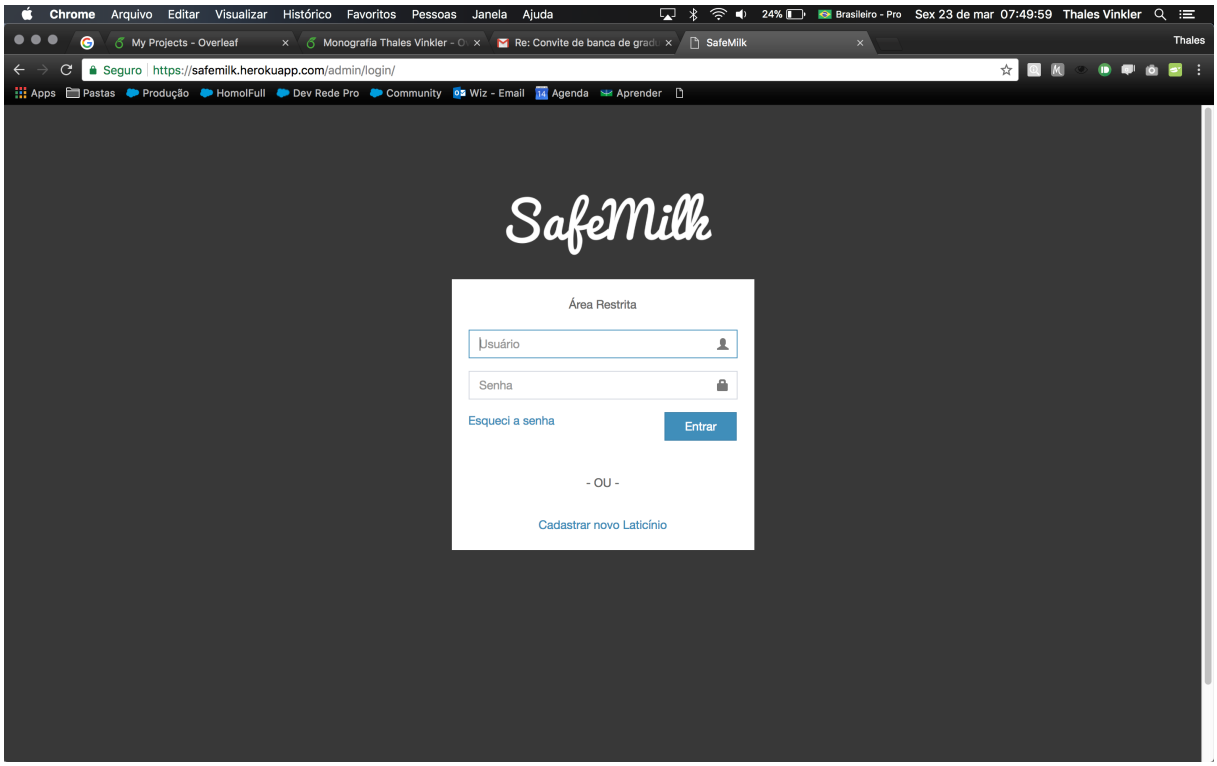

Figura IV.1: Tela de login do sistema web.

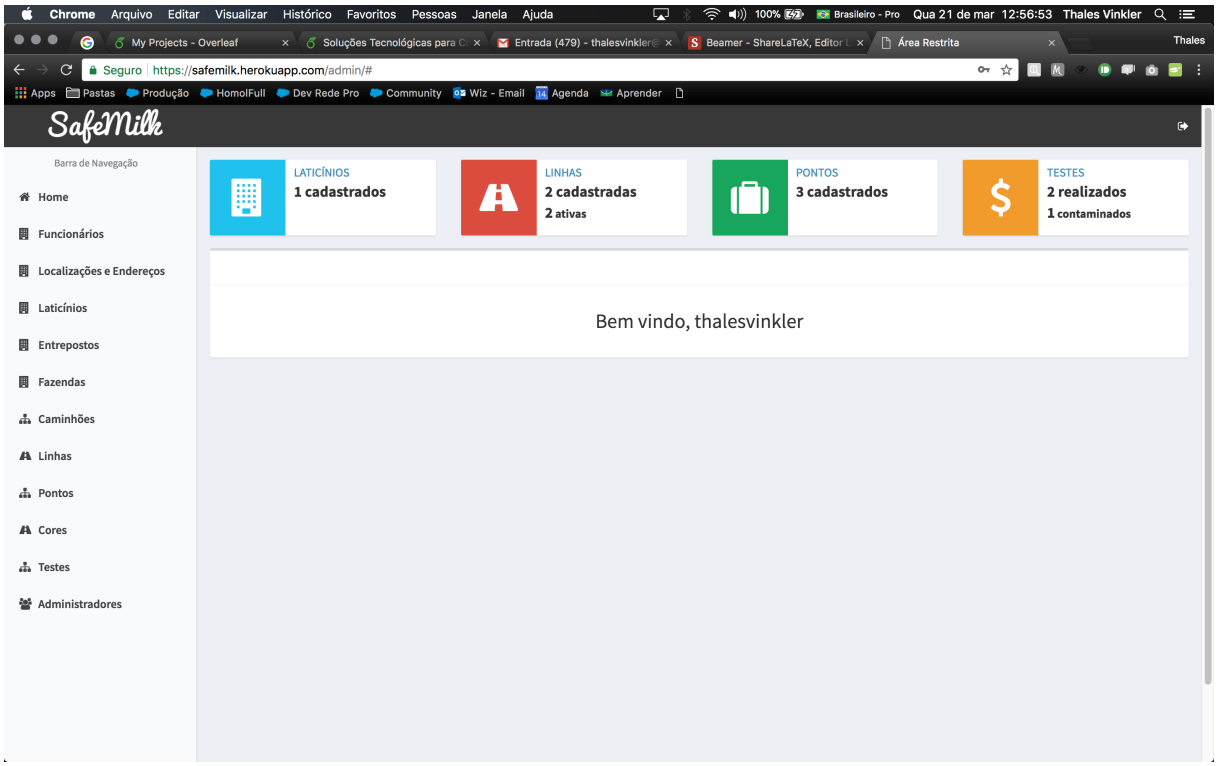

Figura IV.2: Tela principal do sistema web.

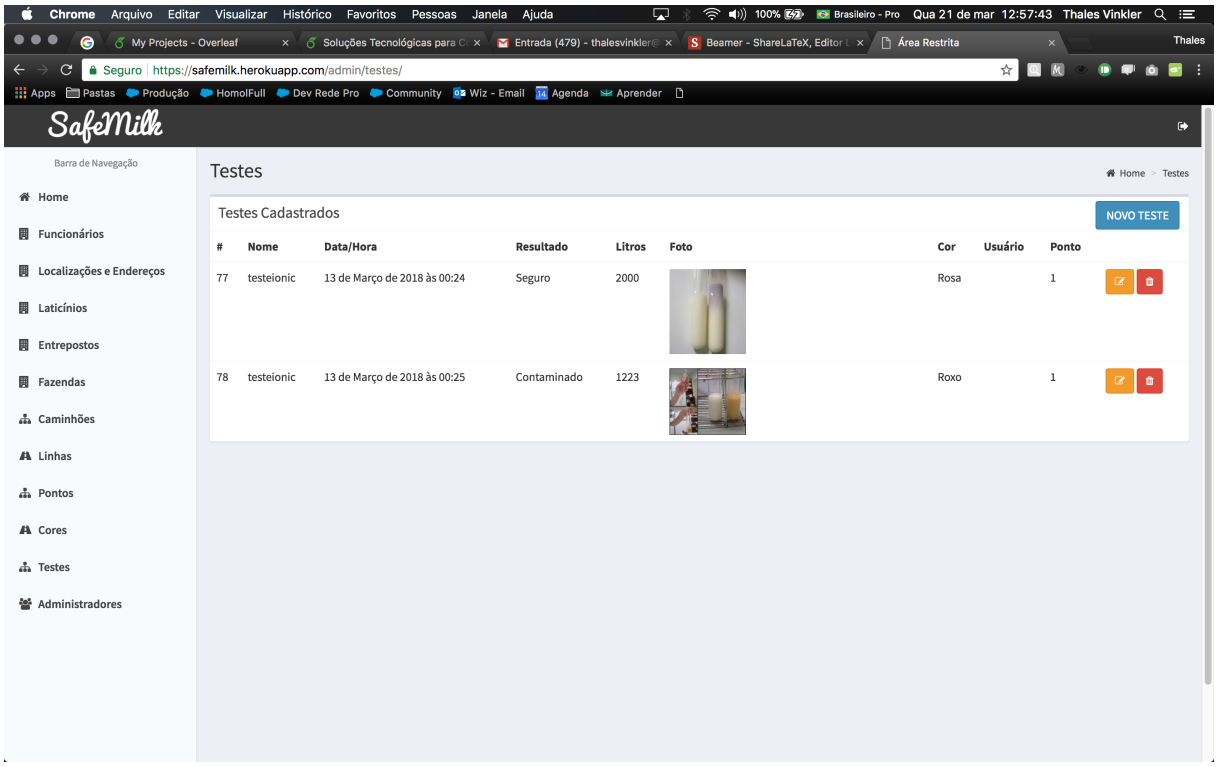

Figura IV.3: Tela de visualização de testes do sistema web.

Anexo V

Especificação dos Casos de Teste

Casos de Teste

Funcionalidades (Casos de uso) a serem testadas, e seus respectivos casos de teste:

- Usuários conseguem realizar login
    - CT01 - Realizar Login na Web
    - CT02 - Errar Usuário/Senha ao realizar login na Web
    - CT12 - Realizar Login no App
    - CT13 – Errar Usuário/Senha ao realizar login no App

- Administrador do Laticínio consegue gerenciar a logística de transporte
    - CT03 - Adicionar Funcionários
    - CT04 - Adicionar Localização
    - CT05 - Adicionar Laticínios
    - CT06 - Adicionar Entrepostos
    - CT07 - Adicionar Fazendas
    - CT08 - Adicionar Caminhões
    - CT09 - Adicionar Linhas de Transporte
    - CT10 - Adicionar Pontos de Coleta/Entrega
- Administradores dos Laticínios conseguem visualizar testes realizados.
    - CT11 - Visualizar Testes Realizados

- Caminhoneiros e Técnicos conseguem visualizar testes
    - CT14 - Visualizar Testes no App
- Caminhoneiros e Técnicos conseguem registrar testes
    - CT15 - Registrar Testes no App
- Caminhoneiros e Técnicos conseguem visualizar tutorial do FORMFIX®
    - CT16 - Assistir Vídeo Processo FORMFIX® no App

Casos de teste detalhados:

CT01 – Realizar Login na Web
Objetivo: *Realizar login no sistema web*
Pré-condições: *Já estar cadastrado na plataforma*
Entrada: *Nome de Usuário e Senha válidos*
Passos:

1. *Usuário acessa tela de login no sistema;*
2. *Usuário insere o campo Usuário;*
3. *Usuário insere o campo Senha;*
4. *Usuário clica em "Entrar".*

Saída:

1. *Sistema valida os campos usuário e senha;*
2. *O sistema abre a página Home*

CT02 – Errar Usuário/Senha ao realizar login na Web
Objetivo: *Bloquear login no sistema web/mobile*
Pré-condições: *Não estar cadastrado na plataforma*

Entrada: *Nome de Usuário e Senha inválidos*
Passos:

1. *Usuário acessa tela de login no sistema;*
2. *Usuário insere o campo Usuário;*
3. *Usuário insere o campo Senha;*
4. *Usuário clica em Entrar.*

Saída:

1. *Sistema valida os campos usuário e senha;*
2. *O sistema abre novamente a página de login para repetir a ação.*

CT03 – Adicionar funcionário
Objetivo: *Adicionar um funcionário ao laticínio*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de um funcionário*
Passos:

1. *Usuário clica no item "Funcionários" do menu lateral;*
2. *Usuário clica no botão "Novo Funcionário";*
3. *Usuário insere o campo "Nome do Usuário";*
4. *Usuário insere o campo "Email";*
5. *Usuário insere uma senha padrão no campo "Senha";*
6. *Usuário insere novamente uma senha no campo "Confirmar Senha";*
7. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de funcionário já com o funcionário aparecendo na listagem.*

CT04 – Adicionar Localização
Objetivo: *Adicionar uma localização existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de uma localização*
Passos:

1. *Usuário clica no item "Localizações e Endereços" do menu lateral;*
2. *Usuário clica no botão "Novo Local";*
3. *Usuário insere o campo "Nome";*
4. *Usuário insere o campo "Latitude";*
5. *Usuário insere o campo "Longitude";*
6. *Usuário insere o campo "Estado";*
7. *Usuário insere o campo "Cidade";*
8. *Usuário insere o campo "Endereço";*
9. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de localizações já com a localização aparecendo na listagem.*

CT05 – Adicionar Laticínio
Objetivo: *Adicionar um laticínio existente*

Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de um laticínio*
Passos:

1. *Usuário clica no item "Laticínios" do menu lateral;*
2. *Usuário clica no botão "Novo Laticínio";*
3. *Usuário insere o campo "Nome";*
4. *Usuário insere o campo "CNPJ";*
5. *Usuário insere o campo "Quantidade de reagente" que o laticínio possui;*
6. *Usuário escolhe no campo "Localização" onde fica o laticínio;*
7. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de laticínios já com o laticínio aparecendo na listagem.*

CT06 – Adicionar Entreposto
Objetivo: *Adicionar um entreposto existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de um entreposto*
Passos:

1. *Usuário clica no item "Entrepostos" do menu lateral;*
2. *Usuário clica no botão "Novo Entreposto";*
3. *Usuário insere o campo "Nome";*
4. *Usuário insere o campo "CNPJ";*
5. *Usuário escolhe no campo "Localização" onde fica o entreposto;*
6. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de entreposto já com o entreposto aparecendo na listagem.*

CT07 – Adicionar Fazenda
Objetivo: *Adicionar uma fazenda existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de uma fazenda*
Passos:

1. *Usuário clica no item "Fazendas" do menu lateral;*
2. *Usuário clica no botão "Nova Fazenda";*
3. *Usuário insere o campo "Nome";*
4. *Usuário insere o campo "CNPJ";*
5. *Usuário insere o campo "Proprietário";*
6. *Usuário escolhe no campo "Localização" onde fica a fazenda;*
7. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de fazendas já com a fazenda aparecendo na listagem.*

CT08 – Adicionar Caminhão

Objetivo: *Adicionar um caminhão existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de um caminhão*
Passos:

1. *Usuário clica no item "Caminhões" do menu lateral;*
2. *Usuário clica no botão "Novo Caminhão";*
3. *Usuário escolhe no campo "Motorista", qual usuário irá dirigir o caminhão;*
4. *Usuário insere o campo "Placa";*
5. *Usuário insere o campo "Chassi";*
6. *Usuário insere o campo "Marca";*
7. *Usuário insere o campo "Modelo";*
8. *Usuário insere o campo "Cor";*
9. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de caminhões já com o caminhão aparecendo na listagem.*

CT09 – Adicionar Linha de Transporte
Objetivo: *Adicionar uma linha de transporte existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de uma linha de transporte*
Passos:

1. *Usuário clica no item "Linhas" do menu lateral;*
2. *Usuário clica no botão "Nova Linha";*
3. *Usuário insere o campo "Nome";*
4. *Usuário insere o campo "Origem";*
5. *Usuário insere o campo "Destino";*
6. *Usuário escolhe no campo "Laticínio", a qual laticínio pertence a linha;*
7. *Usuário escolhe no campo "Caminhão", qual o caminhão que irá percorrer aquela linha;*
8. *Usuário escolhe no campo "Ativa", se a linha está ou não ativa;*
9. *Usuário clica em "Salvar".*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de linhas já com a linha aparecendo na listagem.*

CT10 – Adicionar Ponto de Coleta/Entrega
Objetivo: *Adicionar um ponto de coleta/entrega existente*
Pré-condições: *Estar logado e na página Home*
Entrada: *Informações de um ponto*
Passos:

1. *Usuário clica no item "Pontos" do menu lateral;*
2. *Usuário clica no botão "Novo Ponto";*
3. *Usuário escolhe no campo "Ordem", qual a ordem de visita daquele ponto dentro de uma linha;*

4. *Usuário escolhe no campo “Linha”, a qual linha de transporte esse ponto pertence.*
5. *Usuário insere o campo “Data”;*
6. *Usuário insere o campo “Hora”;*
7. *Usuário escolhe no campo “Processo”, qual o processo será realizado naquele ponto (coleta ou entrega);*
8. *Usuário insere o campo “Litros”;*
9. *Usuário escolhe no campo “Tipo do Ponto”, se o ponto é uma fazenda, um entreposto ou um laticínio;*
    a. *Se o Usuário escolhe “Fazenda”, ele escolhe no campo “Fazenda”, qual fazenda será esse ponto.*
    b. *Se o Usuário escolhe “Entreposto”, ele escolhe no campo “Entreposto”, qual entreposto será esse ponto.*
    c. *Se o Usuário escolhe “Laticínio”, ele escolhe no campo “Laticínio”, qual laticínio será esse ponto.*
10. *Usuário clica em “Salvar”.*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem de pontos já com o ponto aparecendo na listagem.*

CT11 – Visualizar Testes Realizados
Objetivo: *Visualizar os testes realizados*
Pré-condições: *Estar logado e na página Home*
Entrada: -
Passos:

1. *Usuário clica no item “Testes” do menu lateral;*

Saída:

1. *O sistema abre a página de listagem de testes com todos os testes já realizados.*

CT12 – Realizar Login no App
Objetivo: *Realizar login no aplicativo mobile*
Pré-condições: *Já estar cadastrado na plataforma*
Entrada: *Nome de Usuário e Senha válidos*
Passos:

1. *Usuário abre o aplicativo “SafeMilk”;*
2. *Usuário insere o campo “Usuário”;*
3. *Usuário insere o campo “Senha”;*
4. *Usuário clica em “Entrar”.*

Saída:

1. *Sistema valida os campos usuário e senha;*
2. *O sistema abre a página “Home”;*

CT13 – Errar Usuário/Senha ao realizar login no App
Objetivo: *Realizar login no aplicativo mobile*
Pré-condições: *Não estar cadastrado na plataforma*
Entrada: *Nome de Usuário e Senha inválidos*

Passos:

1. *Usuário abre o aplicativo “SafeMilk”;*
2. *Usuário insere o campo “Usuário”;*
3. *Usuário insere o campo “Senha”;*
4. *Usuário clica em Entrar.*

Saída:

1. *Sistema valida os campos usuário e senha;*
2. *Sistema informa por meio de PopUp que houve Falha e o acesso foi negado.*
3. *Sistema espera pela confirmação de “Ok” do usuário;*
4. *Sistema volta à página de “Login”;*

CT14 – Visualizar Testes no App
Objetivo: *Visualizar os testes realizados pelo próprio usuário*
Pré-condições: *Estar logado*
Entrada: -
Passos:

1. *Usuário clica na opção “Home”, do menu inferior.*

Saída:

2. *O sistema abre a página de listagem em cards de todos os testes já realizados pelo usuário.*

CT15 – Registrar Testes no App
Objetivo: *Adicionar um teste realizado*
Pré-condições: *Estar logado*
Entrada: *IthaInformações de um teste*
Passos:

1. *Usuário clica no item “Teste” do menu inferior;*
2. *Usuário escolhe no campo “Ponto”, em qual ponto o teste está sendo realizado;*
3. *Usuário insere o campo “Litros”;*
4. *Usuário escolhe no campo “Cor”, qual a cor apresentada no teste colorimétrico;*
5. *Usuário escolhe no campo “Resultado”, qual o resultado do teste como base na cor apresentada no teste colorimétrico;*
6. *Usuário clica no botão de “Câmera”, para tirar uma foto do teste realizado;*
7. *Usuário tira foto com o aplicativo de câmera integrada que irá abrir, e pode escolher duas opções depois;*
    a. *Usuário clica em “Use Photo”, caso queira utilizar a foto tirada;*
    b. *Usuário clica em “Retake”, caso queira retirar outra foto;*
8. *Usuário clica em “Salvar”.*

Saída:

1. *Sistema valida os campos;*
2. *O sistema abre a página de listagem em cards dos testes já com o teste aparecendo na listagem.*

CT16 – Assistir Tutorial FORMFIX® no App
Objetivo: *Assistir o vídeo tutorial do processo do FORMFIX®*
Pré-condições: *Estar logado*
Entrada: -
Passos:

1. *Usuário clica na opção “Tutorial”, do menu inferior.*
2. *Usuário clica no vídeo para iniciar a execução.*

Saída:

1. *O sistema abre o vídeo em uma modal para o usuário assitir.*

Anexo VI

Comentários dos Usuários durante os Teste

Comentário dos usuários durante a execução dos testes

- Testes Alfa
    - Usuário 1
        - CT02: “Podia existir uma informação pro usuário que ele errou o usuário/senha.”
        - CT04: “Poderia existir uma seleção de Latitude e Longitude diretamente via mapa do Google.”
        - CT10: “A data e hora seriam referentes ao momento que eu quero que o ponto seja visitado ou a data/hora de agora que estou adicionando?”
        - CT11: “Poderia existir filtros para selecionar testes específicos.”
        - CT14: “Poderia existir filtros para selecionar testes específicos.”
        - CT15: “Podia identificar a cor da amostra sozinho pela foto.”

    - Usuário 2
        - CT05: “É preciso realmente do CNPJ do Laticínio?”
        - CT10: “A data e hora seriam referentes ao momento que eu quero que o ponto seja visitado ou a data/hora de agora que estou adicionando?”
        - CT11: “Poderia dar pra clicar no teste e ver mais detalhes dele.”
        - CT14: “Poderia existir filtros para selecionar testes específicos.”
        - CT15: “Identificação do resultado pela foto.”
        - CT16: “Poderia voltar para o App sozinho sem precisar fechar o vídeo.”

    - Usuário 3
        - CT02: “Poderia existir uma mensagem de feedback para o usuário.”
        - CT08: “É preciso preencher o Chassi mesmo?”
        - CT11: “Poderia existir filtros para selecionar testes específicos.”
        - CT15: “Poderia dar pra adicionar foto do celular.”

    - Usuário 4
        - CT09: “Poderia existir uma seleção de Origem e Destino diretamente via mapa do Google.”
        - CT10: “A data e hora seriam referentes ao momento que eu quero que o ponto seja visitado ou a data/hora de agora que estou adicionando?”
        - CT14: “Poderia existir filtros para selecionar testes específicos.”

    - Usuário 5
        - CT04: “Poderia existir uma seleção de Lagitude e Longitude diretamente via mapa do Google.”
        - CT08: “É preciso preencher o Chassi mesmo?”

        - CT10: “A data e hora seriam referentes ao momento que eu quero que o ponto seja visitado ou a data/hora de agora que estou adicionando?”
        - CT15: “Identificar a cor da amostra automático ao tirar a foto.”

- Testes Beta
    - Usuário 1
        - CT02: “Podia existir uma informação pro usuário que ele errou o usuário/senha.”
        - CT04: “Poderia existir uma seleção de Latitude e Longitude diretamente via mapa do Google.”
        - CT05: “É preciso realmente do CNPJ do Laticínio?”
        - CT08: “É preciso preencher o Chassi mesmo?”
        - CT09: “Poderia existir uma seleção de Origem e Destino diretamente via mapa do Google.”
        - CT10: “A data e hora seriam referentes ao momento que eu quero que o ponto seja visitado ou a data/hora de agora que estou adicionando?”
        - CT11: “Poderia existir filtros para selecionar testes específicos.”

    - Usuário 2
        - CT14: “Poderia existir filtros para selecionar testes específicos.”
        - CT15: “Identificar a cor da amostra automático ao tirar a foto.”
        - CT16: “Poderia voltar para o App sozinho sem precisar fechar o vídeo.”