

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

PIERRE CAPISTRANO LOPES

**PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM
ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS**

**RIO DO SUL
2023**

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

PIERRE CAPISTRANO LOPES

**PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM
ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS**

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí, como condição parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Prof. Orientador: M.e Jullian Hermann Creutzberg

**RIO DO SUL
2023**

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

PIERRE CAPISTRANO LOPES

**PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM
ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS**

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí- UNIDAVI, a ser apreciado pela Banca Examinadora, formada por:

Professor Orientador: M.e Jullian Hermann Creutzberg

Banca Examinadora:

Prof. M.e Fernando Andrade Bastos

Prof. Esp. Sandro Alencar Fernandes

Rio do Sul, 29 de novembro de 2023.

RESUMO

Nos últimos anos o Alto Vale do Itajaí tem sido uma região de destaque no setor agrícola, conseqüentemente, os escritórios de soluções agrícolas têm ganhado cada vez mais importância nesse contexto. No entanto, o crescimento desses escritórios demanda um gerenciamento eficiente, assim como ocorre em qualquer empresa em expansão. Este trabalho tem como objetivo desenvolver um protótipo de sistema web para gerenciamento de propostas em escritórios de soluções agrícolas. Quanto a metodologia, caracteriza-se como uma pesquisa aplicada e descritiva. Para atingir os objetivos foi realizada uma revisão da literatura sobre as tecnologias e linguagens de programação utilizadas no desenvolvimento. Visando explicar o processo atual de um escritório de soluções agrícolas, foi realizado um mapeamento de cenário. O capítulo de desenvolvimento compreendeu o detalhamento da análise, bem como contém o levantamento de requisitos e os diagramas de caso de uso, entidade de relacionamento e de classes. No capítulo de desenvolvimento também é apresentada uma seção sobre a implementação, contendo todos os detalhes do desenvolvimento e funcionamento do protótipo. O protótipo desenvolvido oferece ao usuário uma ferramenta de auxílio na organização das informações e gestão das rotinas internas aos escritórios de projetos agrícolas, disponibilizando um controle eficiente de dados e documentos. O protótipo permite o cadastro de pessoas, imóveis, propostas, laudos de acompanhamento e documentações, facilitando o acesso a essas informações em um ambiente integrado.

Palavras-Chave: Soluções agrícolas, Desenvolvimento web, Sistemas de informação.

ABSTRACT

In recent years, the Alto Vale do Itajaí region has emerged as a prominent area in the agricultural sector, consequently, agricultural solutions offices have gained increasing importance in this context. However, the growth of these offices requires efficient management, as is the case with any expanding business. This work aims to develop a prototype web system for proposal management in agricultural solutions offices. The methodology is characterized as applied and descriptive research. To achieve the objectives, a literature review on the technologies and programming languages used in development was conducted. To explain the current process of an agricultural solutions office, a scenario mapping was carried out. The development chapter includes a detailed analysis, as well as a survey of requirements and diagrams of use cases, relationship entities, and classes. The development chapter also includes a section on implementation, providing all the details of the prototype's development and operation. The developed prototype offers users a tool to assist in organizing information and managing internal routines in agricultural project offices, providing efficient control of data and documents. The prototype allows the registration of individuals, properties, proposals, follow-up reports, and documentation, facilitating access to this information in an integrated environment.

Keywords: Agricultural solutions, Web development, Information systems.

LISTA DE FIGURAS

Figura 1- Código sem boas práticas de nomeação	17
Figura 2- Código com boas práticas de nomeação	17
Figura 3- Compilação no .NET Framework	22
Figura 4- Exemplo de Json Web Token	29
Figura 5- Proposta de Financiamento Simplificado	32
Figura 6- Capacidade de pagamento	32
Figura 7- Laudo de Supervisão e Assistência Técnica	34
Figura 8- Diagrama de Atividade – Mapeamento de Cenário	35
Figura 9- Diagrama de Caso de Uso	40
Figura 10- Diagrama de Atividade – Cadastro de Proposta	41
Figura 11- Diagrama de Entidade Relacionamento	42
Figura 12- Diagrama de Entidade Relacionamento – Identity Core	43
Figura 13- Diagrama de Classe	44
Figura 14- Login (RF 01)	47
Figura 15- Página Inicial e Menu (RF 03)	48
Figura 16- Listagem de pessoas (RF 05)	49
Figura 17- Cadastro de Pessoa (RF 05)	49
Figura 18- Listagem de Documentações (RF 10)	50
Figura 19- Cadastro de Documentação (RF 10)	50
Figura 20- Download de Arquivo de Documentação (RF 10)	50
Figura 21- Listagem de Culturas (RF 11)	51
Figura 22- Cadastro de Culturas (RF 11)	51
Figura 23- Listagem de Imóveis (RF 13)	51
Figura 24- Cadastro de Imóvel (RF 13)	52

Figura 25- Listagem de Usuários (RF 04).....	52
Figura 26- Cadastro de Usuário (RF 04).....	52
Figura 27- Listagem de Permissões de Usuário (RF 14).....	53
Figura 28- Cadastro de Permissão de Usuário (RF 14).....	53
Figura 29- Listagem de Filiais (RF 08).....	54
Figura 30- Cadastro de Filial (RF 08).....	54
Figura 31- Listagem de Tipo de Documentação (RF 16).....	54
Figura 32- Cadastro de Tipo de Documentação (RF 16).....	55
Figura 33- Listagem de Tipo de Proposta (RF 07).....	55
Figura 34- Cadastro de Tipo de Proposta (RF 07).....	56
Figura 35- Listagem de Proposta (RF 06).....	56
Figura 36- Cadastro de Proposta (RF 06).....	57
Figura 37- Listagem de Imóveis da Proposta (RF 15).....	57
Figura 38- Cadastro de Imóvel da Proposta (RF 15).....	58
Figura 39- Listagem de Laudos de Acompanhamento (RF 09).....	58
Figura 40- Cadastro de Laudo de Acompanhamento (RF 09).....	59
Figura 41- Listagem de Diagnósticos do Laudo de Acompanhamento (RF 09).....	59
Figura 42- Cadastro de Diagnóstico do Laudo de Acompanhamento (RF 09).....	60
Figura 43- Formulário de Impressão de Proposta (RF 12).....	60
Figura 44- Impressão de Proposta (RF 12).....	61
Figura 45- Menu Lateral Fechado.....	61
Figura 46- Menu Lateral Aberto.....	62
Figura 47- Cadastro de Imóvel Responsivo.....	63
Figura 48- Listagem de Imóvel Responsivo.....	63
Figura 49- Tratamento de Permissão de Usuário.....	64
Figura 50- Tratamento de Permissão de Usuário no Formulário.....	64

Figura 51- Mensagem de Validação de Permissão de Usuário	65
Figura 52- Mensagem de Sucesso Alert do Material UI	65
Figura 53- Mensagem de Confirmação Alert Padrão	65

LISTA DE QUADROS

Quadro 1 - Requisitos de Software.....	16
Quadro 2 – Métodos HTTP	25
Quadro 3 - Propriedades ACID	26
Quadro 4 – Requisitos Funcionais.....	37
Quadro 5 – Requisitos Funcionais Opcionais	38
Quadro 6 – Requisitos Não Funcionais	39
Quadro 7 – Regras de Negócio.....	39

SUMÁRIO

1. INTRODUÇÃO	11
1.1 PROBLEMA DE PESQUISA	12
1.2 OBJETIVOS	12
1.2.1 Geral	12
1.2.2 Específicos	12
1.3 JUSTIFICATIVA	13
1.4 CONTEXTUALIZAÇÃO DA EMPRESA	13
2. REVISÃO DA LITERATURA	15
2.1 ENGENHARIA DE SOFTWARE	15
2.1.1 Levantamento de Requisitos	15
2.1.2 Requisitos Funcionais e Não Funcionais	15
2.1.3 Regras de Negócio	16
2.2 BOAS PRÁTICAS DE DESENVOLVIMENTO	16
2.2.1 Nomeação	17
2.2.2 Funções	18
2.3 PROGRAMAÇÃO ORIENTADA A OBJETOS	18
2.4 HTML	19
2.5 JAVASCRIPT	20
2.5.1. TypeScript	21
2.6. FRAMEWORK	21
2.6.1. .NET Framework	21
2.6.2. Entity Framework	22
2.7. REACTJS	23
2.7.1 Material UI	23
2.7.2 Axios	23
2.8 WEB SERVICES	24
2.8.1 API	24
2.8.2 Arquitetura REST	25
2.8.3 Métodos HTTP	25
2.9 BANCO DE DADOS	26
2.9.1 Banco de Dados Relacional	26
2.9.2 SQL Server	27
2.10 C#	28
2.11 SWAGGER	28

2.12 JWT.....	29
3. METODOLOGIA DA PESQUISA.....	31
3.1 MAPEAMENTO DE CENÁRIO	31
4. PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS	36
4.1 ANÁLISE	36
4.1.1 Visão Geral do Protótipo	36
4.1.2 Requisitos	37
4.1.3 Diagramas	40
4.2 Implementação.....	44
4.2.1 Técnicas e Ferramentas Utilizadas	44
4.2.2 Utilização e Funcionamento	46
5. CONSIDERAÇÕES FINAIS.....	67
5.1 RECOMENDAÇÕES DE TRABALHOS FUTUROS	68
REFERÊNCIAS	70
ANEXOS	73
ANEXO I – TERMO DE AUTORIZAÇÃO PARA USO DE NOME EMPRESARIAL E DADOS.....	73

1. INTRODUÇÃO

O Alto Vale do Itajaí vem crescendo cada vez mais no setor agrícola, se tornando uma região de destaque, com a agricultura familiar desempenhando um papel fundamental em seu desenvolvimento econômico. Conseqüentemente, os escritórios de soluções agrícolas têm ganhado cada vez mais importância nesse contexto, à medida que oferecem suporte e serviços essenciais aos agricultores locais. No entanto, o crescimento desses escritórios demanda um gerenciamento eficiente, assim como ocorre em qualquer empresa em expansão. É neste ponto que a tecnologia pode desempenhar um papel crucial, oferecendo soluções para aprimorar e otimizar esse processo de gerenciamento.

Os escritórios de soluções agrícolas oferecem diversos serviços aos agricultores, como financiamento agrícola, investimentos, análises de terras, seguro de safra, entre outros. No entanto, a prestação desses serviços requer o gerenciamento eficiente de uma ampla gama de informações. Caso essas informações não sejam devidamente administradas, podem ocorrer perdas de dados, ineficiência nos processos e, em casos extremos, problemas judiciais.

Diante dessa necessidade de gerenciamento mais eficiente, surge um problema: a escassez de sistemas específicos no mercado para o gerenciamento de escritórios de soluções agrícolas. Existem diversos sistemas ERP (*Enterprise Resource Planning*) genéricos que atendem a diferentes tipos de empresas, além de sistemas voltados para o gerenciamento de tarefas e projetos de forma mais geral. Contudo, um escritório de soluções agrícolas possui requisitos específicos que vão além do escopo das soluções existentes.

Esse tipo de sistema deve possuir as funcionalidades adequadas para gestão das demandas, suas documentações, etapas, atividades e andamento. Além disso, é essencial contar com funcionalidades para controle de clientes, gestão financeira, bem como a geração de gráficos e relatórios contendo indicadores relevantes para o acompanhamento do desempenho desses escritórios.

O sistema proposto visa suprir a lacuna existente no mercado, fornecendo uma solução personalizada e eficiente para atender às necessidades específicas desses escritórios. Por meio deste estudo, pretende-se explorar as funcionalidades essenciais e os benefícios potenciais que um sistema desse tipo pode oferecer, contribuindo assim para o aprimoramento e profissionalização dos escritórios de projetos agrícolas na região do Alto Vale do Itajaí.

No que se refere a organização deste trabalho, além deste capítulo introdutório, ele contém no segundo capítulo uma revisão da literatura, onde se descrevem as tecnologias

empregadas no desenvolvimento do protótipo, com uma explicação de cada uma. No terceiro capítulo consta a metodologia da pesquisa, que relata em detalhes o processo metodológico do trabalho, incluindo a seção de mapeamento de cenário que explica o processo atual realizado por um escritório de soluções agrícolas. O quarto capítulo apresenta o desenvolvimento da aplicação, acompanhado de uma análise do projeto, com levantamento das regras de negócio, levantamento de requisitos e os diagramas necessários para auxiliar a construção do protótipo, ainda há a seção de implementação onde são demonstradas todas as rotinas da aplicação, e são detalhadas as tecnologias utilizadas durante o desenvolvimento. Por fim, no último capítulo são apresentadas as considerações finais juntamente com as recomendações para trabalhos futuros.

1.1 PROBLEMA DE PESQUISA

Qual a melhor forma de gerenciar escritórios de soluções agrícolas?

1.2 OBJETIVOS

1.2.1 Geral

- Desenvolver um protótipo de sistema web para gerenciamento de propostas em escritórios de soluções agrícolas.

1.2.2 Específicos

- Descrever as tecnologias que serão utilizadas no desenvolvimento do protótipo.
- Explicar o processo atual realizado por um escritório de soluções agrícolas.
- Detalhar a análise do protótipo por meio do levantamento de requisitos e da elaboração de diagramas.
- Construir o protótipo com utilização das tecnologias selecionadas.

1.3 JUSTIFICATIVA

O Alto vale do Itajaí é uma região onde a agricultura representa grande parte economia, principalmente proveniente da agricultura familiar, e dessa forma os escritórios de soluções agrícolas crescem cada vez mais, necessitando de um gerenciamento melhor como qualquer empresa em crescimento. Neste contexto, entende-se que a tecnologia poderá auxiliar esse processo.

Um escritório de soluções agrícolas presta serviços para agricultores que necessitam de financiamentos, investimentos, análises de terras, seguros de safra, etc. E todos esses serviços necessitam de uma grande quantidade de informações que, caso não gerenciadas corretamente, podem ocasionar perda de dados, ineficiência dos processos e nos piores casos, até problemas judiciais.

Com um sistema próprio para essas atividades, os escritórios possuirão uma maior qualidade dos seus serviços, e um aumento significativo na eficiência, se destacando assim da concorrência que não possui um sistema específico.

A partir dessa necessidade surge o problema, não foram encontrados no mercado sistemas específicos para o gerenciamento desse tipo de escritório.

1.4 CONTEXTUALIZAÇÃO DA EMPRESA

A Lopes Projetos Agrícolas é uma empresa que atua no ramo de soluções agrícolas, focada na agricultura familiar, iniciou seus serviços no ano de 2017 na cidade de Leoberto Leal em Santa Catarina e posteriormente abriu filiais em nas cidades de Imbuia, Ituporanga e Alfredo Wagner.

Atualmente a empresa conta com quatro escritórios, oito funcionários além do proprietário, e desses funcionários seis possuem registro como técnicos agrícolas, podendo assim realizar tanto os serviços do escritório como serviços de campo.

Os serviços fornecidos são principalmente a elaboração de propostas de custeio e investimento para o PRONAF (Programa Nacional de Fortalecimento da Agricultura Familiar) juntamente com instituições financeiras, visitas técnicas nas lavouras, análises de terras,

interpretação das análises de terras, laudos de PROAGRO (Programa de Garantia da Atividade Agropecuária), além de outros serviços com menor frequência.

2. REVISÃO DA LITERATURA

Neste capítulo são apresentados os conceitos das ferramentas e linguagens utilizados para o desenvolvimento do protótipo.

2.1 ENGENHARIA DE SOFTWARE

Sommerville (2011) trata a engenharia de software como uma área de grande importância, já que a sociedade atual vem cada vez mais dependendo de softwares, e a engenharia de software faz com que os softwares possam ser desenvolvidos com uma maior velocidade, confiabilidade e qualidade.

A engenharia de software estuda todos os processos envolvidos no desenvolvimento de um software, ela envolve diversas atividades distintas, onde é necessário ter habilidades em diversas áreas. (SOMMERVILLE, 2011).

Os softwares atuais devem atender diversas demandas, por isso, deve se ter o conhecimento das funcionalidades e características do mesmo, e a engenharia de software utiliza de requisitos para manter o conhecimento necessário do mesmo. (MORAIS; ZANIN, 2020).

2.1.1 Levantamento de Requisitos

No desenvolvimento de sistemas, entender os requisitos é fundamental. Isso começa na fase de concepção do projeto, onde é essencial compreender o problema, quem precisa de uma solução e que tipo de solução é necessária. A partir daí, é crucial ter uma visão clara dos objetivos de longo prazo do projeto, pois esses objetivos influenciarão os requisitos funcionais e os não funcionais. Em resumo, o processo envolve entender a fundo o que o projeto precisa alcançar e, com base nisso, identificar os requisitos específicos que o guiarão ao longo do desenvolvimento (PRESSMAN; MAXIM, 2021).

2.1.2 Requisitos Funcionais e Não Funcionais

De acordo com Sommerville (2011) os requisitos funcionais e não funcionais são basicamente o que classifica os requisitos de software, conforme exemplifica o Quadro 1.

Quadro 1 - Requisitos de Software

Nome	Conceito
Requisito Funcional	São declarações de serviços que o sistema deve fornecer, de como o sistema deve reagir a entradas específicas e de como o sistema deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais também podem explicitar o que o sistema não deve fazer.
Requisito Não Funcional	São restrições aos serviços ou funções oferecidas pelo sistema. Incluem restrições de <i>timing</i> , restrições no processo de desenvolvimento e restrições impostas pelas normas. Ao contrário das características individuais ou serviços do sistema, os requisitos não funcionais, muitas vezes, aplicam-se ao sistema como um todo.

Fonte: Elaborado a partir de Sommerville (2011)

Requisitos funcionais definem o que o sistema deve fazer, já os requisitos não funcionais não possuem ligação direta ao que o sistema irá fazer, porém podem estar relacionados às propriedades crescentes do sistema. (SOMMERVILLE, 2011).

2.1.3 Regras de Negócio

Os objetivos das regras de negócio são de providenciar um desenvolvimento que acabe atendendo as necessidades do usuário final com maior rapidez e precisão. (DEVMEDIA, 2014).

Regras de negócio são definidas com base nas necessidades da aplicação, assim o desenvolvimento deve ocorrer seguindo-as. Desta forma, facilita para que o usuário teste a aplicação e veja que suas regras foram implementadas na aplicação. (DEVMEDIA, 2014).

2.2 BOAS PRÁTICAS DE DESENVOLVIMENTO

Conforme Martin (2009, p. 8), “Um código limpo é simples e direto. Ele é tão bem legível quanto uma prosa bem escrita. Ele jamais torna confuso o objetivo do desenvolvedor, em vez disso, ele está repleto de abstrações claras e linhas de controle objetivas”.

Um código confuso diminui a produtividade da equipe, cada nova alteração pode causar problemas em outras duas ou três partes, exigindo remendos, amarrações para que o código não quebre e com o tempo, a bagunça cresce tanto que é impossível solucionar. Com a redução da produtividade, a gerência contrata novos funcionários para aumentar a mão de obra, mas os novos programadores não conhecem o código por completo, não conseguindo diferenciar a

mudança que altera o propósito do projeto e a aquela que atrapalha. Criando mais e mais confusões, a produtividade chega cada vez mais perto de zero. (MARTIN 2009).

2.2.1 Nomeação

O início de boas práticas de desenvolvimento se dá pela nomeação de variáveis, métodos e classes, pois o nome deve dizer por que existe, o que faz e como é usado. (MARTIN, 2009). A Figura 1 mostra um código que não utiliza uma boa nomeação de suas variáveis, funções, etc.

Figura 1 – Código sem boas práticas de nomeação

```

1 public List<int[]> getThem(){
2     List<int[]> list1 = new ArrayList<int[]>();
3     for (int[] x : theList)
4         if (x[0] == 4)
5             list1.add(x);
6
7     return list1;
8 }

```

Fonte: Martin (2009, p. 18)

Esse código é simples, porém seu contexto não está implícito, dificultando o entendimento. A nomeação seguindo as boas práticas garante que em qualquer parte de um código, o seu contexto estará implícito, facilitando a leitura conforme a Figura 2.

Figura 2 – Código com boas práticas de nomeação

```

1 public List<int[]> getFlaggedCells(){
2     List<int[]> flaggedCells = new ArrayList<int[]>();
3     for (int[] cell : gameBoard)
4         if (cell[STATUS_VALUE] == FLAGGED)
5             flaggedCells.add(cell);
6
7     return flaggedCells;
8 }

```

Fonte: Martin (2009, p. 19)

Além de nomes que dão o contexto do código, é necessário utilizar nomes pronunciáveis. Como uma variável do tipo *Date* nomeada de “genymdhms” por exemplo, ela pode contextualizar a sua utilização, mas a mesma variável nomeada de

“generationTimestamp” facilita muito mais o entendimento, além de melhorar a comunicação da equipe de desenvolvimento. (MARTIN, 2009)

Martin (2009) também salienta a utilização de nomes passíveis de busca, auxiliando na procura de determinadas variáveis, constantes e métodos por todo o projeto, pois utilizar o número 7 como uma constante é comum, mas buscar por todo o código pode demorar por ser um número que pode aparecer em diversos contextos, mas utilizar uma constante com nome MAX_CLASSES_PER_STUDENT torna a pesquisa muito mais rápida.

2.2.2 Funções

A primeira regra para funções é que elas devem ser pequenas, mesmo que não possuam informações de estudos que indiquem que funções extremamente pequenas são superiores. Mas que pode afirmar que ao longo de aproximadamente quatro décadas tem desenvolvido funções de diferentes tamanhos, seja de 3000 linhas; funções de 100 a 300 linhas; e funções que apresentavam apenas 20 a 30 linhas. Essa experiência ensinou que, após inúmeras tentativas e erros, as funções devem ser muito compactas. (MARTIN, 2019).

2.3 PROGRAMAÇÃO ORIENTADA A OBJETOS

Programação Orientada a Objetos (POO) é um paradigma de programação que se concentra em representar entidades do mundo real como objetos, que possuem atributos e métodos que definem seu comportamento e funcionalidade. O objetivo da POO é facilitar a modelagem e a resolução de problemas complexos, dividindo-os em partes menores e mais gerenciáveis. Nesse paradigma, as entidades são representadas por classes, que são usadas para criar objetos. (BARNES; KÖLLING, 2012).

A POO é baseada em quatro conceitos principais: encapsulamento, herança, polimorfismo e abstração. O encapsulamento é a capacidade de esconder os detalhes internos de um objeto e protegê-lo contra acesso indevido. A herança permite que as classes herdem atributos e métodos de outras classes e estendam seu comportamento. O polimorfismo permite que os objetos sejam tratados de maneiras diferentes, dependendo do contexto em que são usados. A abstração é a capacidade de extrair as características essenciais de um objeto e criar uma classe abstrata que define a interface para essas características. Juntos, esses conceitos

permitem que os desenvolvedores criem programas orientados a objetos que são modulares, flexíveis e fáceis de manter. (BARNES; KÖLLING, 2012).

Conforme Farinelli (2007), a programação orientada a objetos “consiste em conceber um sistema computacional como um todo orgânico formado por objetos que se relacionam entre si. Esse enfoque pode ser aplicado tanto à análise de sistemas quanto à programação, e essa é uma das principais vantagens da orientação a objetos”, portanto a mesma metodologia serve tanto para a definição lógica do sistema quanto para a sua implementação.

Farinelli (2007, p. 31) diz que:

[...] dados e processos são apenas componentes, e o enfoque está em identificar quais os objetos que interagem entre si no sistema. Os dados são identificados procurando os atributos que definem os objetos, e os procedimentos pelas operações que estes objetos realizam. A interação entre os objetos é definida pelas estruturas e relacionamentos que são identificados. O resultado é que em um modelo orientado a objetos, existe total coerência entre os dados e os processos, mesmo quando há muitas pessoas trabalhando no mesmo sistema.

Portanto, a definição de dados e processos em análise de sistemas, é uma questão antiga, pois existem metodologias que definem as estruturas de dados primeiros, e após os processos que a utilizam, mas também existem metodologias que fazem o contrário, definindo primeiro os procedimentos que serão automatizados e posteriormente os dados que deverão ser utilizados. (FARINELLI, 2007).

2.4 HTML

Sobre HTML, segundo Saraiva e Barreto (2018, p.9) afirma que “HTML, do inglês *Hyper Text Markup Language*, ou linguagem de marcação de hipertexto, é uma linguagem utilizada para criar documentos para a web.”.

A linguagem HTML não é considerada uma linguagem de programação convencional, em vez disso, é mais apropriadamente classificada como uma de definição da estrutura de um documento. Ao contrário das linguagens de programação tradicionais como C++, Java e Pascal, o HTML não gera um programa executável autônomo. Em vez disso, ela resulta em um arquivo de texto, geralmente com extensão .HTM ou .HTML, que é lido e interpretado por um

navegador. Este software é responsável por exibir no monitor o conteúdo codificado no documento HTML (ALVES, 2014b).

MDN (2022a) define um Documento HTML como um simples texto estruturado com elementos, sendo que os elementos são acompanhados de aberturas e fechamentos de *tags*, onde cada *tag* começa e termina com colchetes angulares (<>). E observa que também existem algumas *tags* que não possuem conteúdo textual, mas que possuem funções específicas como para a apresentação de uma imagem, vídeo ou simplesmente para pular uma linha no texto.

2.5 JAVASCRIPT

Flanagan (2014) descreve o JavaScript como umas das três principais ferramentas para desenvolvimento web, juntamente com HTML e CSS, sendo uma tecnologia presente em praticamente todos os sites da atualidade. Oliveira (2020) complementa dizendo que é uma linguagem de programação orientada a objetos, interpretada e executada por navegadores web e seu objetivo principal é oferecer maior interatividade às páginas.

“Na verdade, o nome “JavaScript” é um pouco enganoso. A não ser pela semelhança sintática superficial, JavaScript é completamente diferente da linguagem de programação Java. E JavaScript já deixou para trás suas raízes como linguagem de script há muito tempo, tornando-se uma linguagem de uso geral robusta e eficiente. A versão mais recente da linguagem [...] define novos recursos para desenvolvimento de software em grande escala.” (FLANAGAN, 2014, p.18).

MDN (2022b) adverte sobre JavaScript de que ele não deve ser confundido com a linguagem de programação Java, mesmo tendo os nomes de suas marcas comerciais bastante semelhantes são duas linguagens de programação significativamente diferentes em sintaxes, semânticas e também bem em seus casos de uso.

Sobre a linguagem JavaScript, Oliveira (2020, p.46) certifica que “Uma característica importante da linguagem JavaScript é que ela não apresenta tipos de dados, isto é, qualquer variável definida é do tipo variante.”.

2.5.1. TypeScript

TypeScript é uma linguagem de programação fortemente tipada baseada no JavaScript, oferecendo sintaxes adicionais ao JavaScript para ofertar um melhor suporte em conjunto ao editor do código para detectar previamente os erros. Também informam que, ao ser executado, o TypeScript é convertido para JavaScript para poder ser executado em qualquer lugar, seja em um navegador, no Node.js ou em outros aplicativos. (TYPESCRIPT, 2023)

Ivanov e Bespoyasov (2020) ainda afirmam que o TypeScript possibilita especificar os tipos das informações, das variáveis no código, o que permite o desenvolvimento de aplicação com maior confiança.

2.6. FRAMEWORK

Conforme afirmado por Freitas, Birnfield e Saraiva (2021, p.47) “*Framework* é uma abstração que une códigos comuns entre vários projetos de *software*, provendo uma funcionalidade genérica. Portanto, trata-se de uma forma mais simples de desenvolver aplicações pelo reuso de componentes.”.

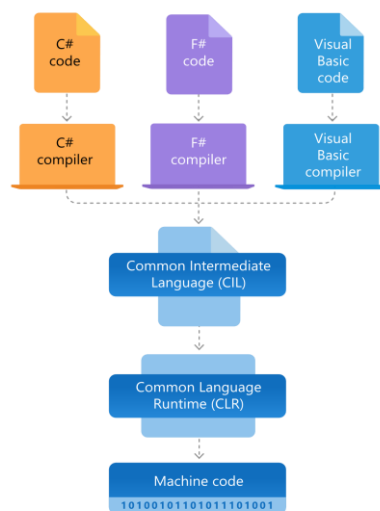
2.6.1. .NET Framework

De acordo com a Microsoft (2023b), o .NET Framework é uma plataforma de execução gerenciada para o Windows que disponibiliza uma variedade de recursos para os aplicativos em execução. Ele é composto por duas partes principais: o CLR (*Common Language Runtime*), responsável pela execução dos aplicativos, e a biblioteca de classes do .NET Framework, que oferece um conjunto de códigos testados e reutilizáveis que os desenvolvedores podem utilizar em seus próprios aplicativos.

O *Common Language Runtime* (CLR) é o mecanismo de execução que identifica aplicativos em execução. Ele fornece serviços como gerenciamento de threads, coleta de lixo, fortemente tipado, identificação de exceção e muito mais. Já a biblioteca de classes oferece um conjunto de APIs e tipos para funcionalidades comuns. Ela oferece tipos de cadeias de caracteres, datas, números, etc. A biblioteca de classes inclui APIs para leitura e gravação de arquivos, conexão a bancos de dados, desenho e muito mais. (MICROSOFT, 2023b, n.p.).

Cada linguagem de programação .NET tem um compilador que transforma seu código em *Common Intermediate Language*. Em *runtime*, o *Common Language Runtime* transforma o código compilado em código do computador e o executa, conforme a Figura 3. O código compilado é armazenado em conjuntos, seja arquivos com uma extensão de arquivo “.dll” ou “.exe”. (MICROSOFT, 2023b).

Figura 3 – Compilação no .NET Framework



Fonte: Microsoft (2023b, n.p.).

2.6.2. Entity Framework

O Entity Framework é um *framework* de mapeamento objeto-relacional (ORM), “permitindo aos desenvolvedores trabalharem com dados relacionais como objetos específicos do domínio, eliminando a necessidade de grande parte do código de acesso aos dados que os desenvolvedores geralmente precisam escrever.” (ENTITY FRAMEWORK, 2021, p.1).

Ainda conforme o Entity Framework (2021), um ORM é uma ferramenta que possibilita a armazenagem automatizada de dados de objetos de domínio em um banco de dados relacional, como o MS SQL Server, sem a necessidade de muita programação. Ela é composta por três componentes principais: objetos de classe de domínio, objetos de banco de dados relacional e informações de mapeamento que definem como os objetos de domínio são relacionados aos objetos de banco de dados (tabelas, visualizações e procedimentos armazenados). A ORM auxilia na separação do *design* do banco de dados do *design* das classes de domínio, tornando a aplicação mais fácil de ser mantida e expandida. Além disso, ela automatiza as operações

CRUD padrão (*Create, Read, Update e Delete*), evitando que o desenvolvedor tenha que escrevê-las manualmente.

2.7. REACTJS

Conforme Roldán (2021), o React é uma biblioteca JavaScript de código aberto e flexível, projetada para a criação de interfaces de usuário complexas a partir de pequenos componentes separados. Com eficiência, o React aprimora a flexibilidade, a manutenção e o desempenho das aplicações, fornecendo um impulso significativo ao fluxo de trabalho, aumentando a velocidade sem comprometer a qualidade.

Na sua documentação, React (2022) enfatiza o objetivo do ReactJS de simplificar o desenvolvimento de interfaces de usuário interativas, permitindo que as interfaces se atualizem e sejam renderizadas automaticamente à medida que os dados são alterados. Eles também destacam a estrutura baseada em componentes encapsulados do ReactJS, que possibilita a fácil combinação desses componentes para criar interfaces mais complexas e reutilizáveis.

2.7.1 Material UI

Material UI é uma biblioteca de componentes React JS de código aberto que segue os princípios de design do Material Design, uma filosofia de design criada pela Google, que enfatiza a simplicidade, a usabilidade e a consistência. (MATERIAL UI, 2023).

Essa biblioteca oferece uma ampla gama de componentes pré-construídos, desde botões e formulários até barras de navegação e cartões, todos projetados para se integrarem perfeitamente em aplicativos React JS. Ao adotar o Material UI, os desenvolvedores podem economizar tempo e esforço, pois não precisam criar componentes personalizados do zero. Em vez disso, eles podem se concentrar na lógica de negócios de suas aplicações, garantindo ao mesmo tempo uma experiência de usuário moderna e agradável. (MATERIAL UI, 2023).

2.7.2 Axios

Axios pode ser definido como “[...] um cliente HTTP baseado-em-promessas para o node.js e para o navegador. É isomórfico (pode rodar no navegador e no node.js com a mesma

base de código). No lado do servidor usa o código nativo do node.js - o módulo http, enquanto no lado do cliente usa XMLHttpRequests”. (AXIOS, 2023, n.p.)

O Axios facilita que desenvolvedores façam requisições HTTP, utilizando todos os verbos, como POST, PUT, GET, DELETE, etc. Permitindo a manipulação da requisição, seja pela rota, *headers*, *params*, *body*, *interceptors*, etc. Garantindo que todas as necessidades durante o desenvolvimento sejam atendidas por ferramentas presentes no Axios. (AXIOS, 2023).

2.8 WEB SERVICES

Web Services são conjuntos de programas que podem ser publicados, buscados e chamados por meio da internet. Esses programas podem realizar um simples processo de troca de mensagens, como também complexas transações comerciais ou industriais, por exemplo, um processo de compra de produtos. Uma vez que um Web Service é publicado em um servidor web, vários programas e até mesmo outros Web Services podem acessá-lo e chamá-lo, tanto para obtenção de dados como para a interação com serviços que uma organização oferece. (TAMAE, 2004).

2.8.1 API

Freitas, Birnfield e Saraiva (2021, p.43) definem API (*Application Programming Interface*, ou em português, Interface de Programação de Aplicação) como, “[...] construções de aplicações que permitem que os desenvolvedores criem funcionalidades complexas mais facilmente. Tais construções abstraem o código mais complexo, proporcionando o uso de sintaxes de forma mais simples.”.

Em relação à definição de API, Rodrigues e Neumann (2020) acrescentam que ela também pode ser descrita como um conjunto de procedimentos e diretrizes de um sistema, permitindo o acesso externo às funcionalidades sem a necessidade de acessar diretamente o código ou a implementação, mas de maneira abstrata e completamente independente da programação.

De acordo com MDN (2022a), uma API é comumente constituída por um conjunto de métodos, propriedades, eventos e URLs padronizados, que podem ser utilizados no

desenvolvimento de aplicações para facilitar a interação entre a aplicação, o servidor e serviços de terceiros.

2.8.2 Arquitetura REST

Segundo as afirmações de Rodrigues et al. (2020), o conceito de REST (*Representational State Transfer*) surgiu a partir de uma dissertação com o intuito de propor uma padronização para a integração entre aplicações baseadas no protocolo HTTP (*Hypertext Transfer Protocol*). Para complementar, Red Hat (2020) esclarece que REST não é um protocolo ou um padrão em si, mas sim um conjunto de restrições arquiteturais que oferece aos desenvolvedores a flexibilidade de utilizar a arquitetura de várias maneiras. Essas possibilidades vão desde comunicações simples para adquirir dados até uma aplicação que não armazena dados, mas obtém todas as informações diretamente do servidor através das comunicações estabelecidas pela arquitetura REST.

2.8.3 Métodos HTTP

Conforme apresentado por MDN (2023), o protocolo HTTP define um conjunto de métodos de requisição que indicam o tipo de ação que se deseja processar. Cada método possui características próprias, mas podem compartilhar algumas de suas características com outros. Como podem ser observados os métodos e suas funções no Quadro 2.

Quadro 2 – Métodos HTTP

Método	Função
GET	O método GET solicita a representação de um recurso específico. Requisições utilizando o método GET devem retornar apenas dados.
HEAD	O método HEAD solicita uma resposta de forma idêntica ao método GET, porém sem conter o corpo da resposta.
POST	O método POST é utilizado para submeter uma entidade a um recurso específico, frequentemente causando uma mudança no estado do recurso ou efeitos colaterais no servidor.
PUT	O método PUT substitui todas as atuais representações do recurso de destino pela carga de dados da requisição.
DELETE	O método DELETE remove um recurso específico.
CONNECT	O método CONNECT estabelece um túnel para o servidor identificado pelo recurso de destino.

OPTIONS	O método OPTIONS é usado para descrever as opções de comunicação com o recurso de destino.
TRACE	O método TRACE executa um teste de chamada <i>loop-back</i> junto com o caminho para o recurso de destino.
PATCH	O método PATCH é utilizado para aplicar modificações parciais em um recurso.

Fonte: Elaborado a partir de MDN (2023).

2.9 BANCO DE DADOS

De acordo com Alves (2014a), um banco de dados é um conjunto de valores e arquivos que se relacionam entre si e demonstram um cadastro de pessoas, vendas, produtos, agendas, etc. Por exemplo, uma planilha ou uma ficha cadastral podem ser exemplos de cadastros. O banco de dados guarda todos esses cadastros em formato virtual e os disponibiliza para as aplicações consultarem, emitirem relatórios, realizarem vendas, etc.

2.9.1 Banco de Dados Relacional

De acordo com Alves (2014a, p. 20), banco de dados relacional “se caracteriza pelo fato de organizar os dados em tabelas (ou relações), formadas por linhas e colunas. Assim, essas tabelas são similares a conjuntos de elementos ou objetos, uma vez que relacionam as informações referentes a um mesmo assunto de modo organizado.”.

Conforme Silva (2021), bancos de dados relacionais são os mais utilizados no mercado atual, mesmo que tenha surgido a partir de modelos de 1970. Eles são conhecidos por serem sistemas de propósito geral, utilizarem a linguagem SQL (*Structured Query Language*; ou Linguagem de Consulta Estruturada, em português) para consultar e manipular os dados, além das propriedades ACID para transações, conforme apresentado no Quadro 3.

Quadro 3 - Propriedades ACID

Atomicidade	As transações são atômicas, executam completamente ou não executam.
Consistência	As transações criam novos estados válidos, ou, em caso de falha, o banco de dados volta para o último estado válido.
Isolamento	Garante que uma transação em andamento não será interferida por outras transações.
Durabilidade	Garante que as alterações de dados feitas por transações executadas com sucesso sejam preservadas, mesmo em caso de falha do sistema.

Fonte: Elaborado a partir de Pritchett (2008)

Ainda de acordo com Silva (2021, p. 14):

Outra característica marcante dos bancos de dados relacionais é a integridade referencial, que garante a acurácia e a consistência dos dados dentro de um relacionamento entre tabelas. Isso é feito por meio de uma chave estrangeira, que faz referência a um valor de uma chave primária em outra tabela, e a integridade referencial garantirá que esse relacionamento é íntegro (o registro referenciado existe).

Codd (1985) destacou os princípios básicos do sistema de banco de dados relacional em 1968, baseando-se na teoria dos conjuntos e da álgebra relacional. Segundo ele, certos conceitos da matemática podiam ser aplicados ao gerenciamento de bancos de dados, provavelmente por ter sido um brilhante matemático.

Ainda segundo Codd (1985), existem um conjunto de doze regras para que um banco de dados relacional fosse admitido como tal:

- Regra de informações;
- Regra de acesso garantido;
- Tratamento de valores nulos;
- Catálogo relacional ativo;
- Inserção, exclusão e alteração em bloco;
- Linguagem de manipulação de dados abrangente;
- Independência física dos dados;
- Independência lógica dos dados;
- Regra de atualização de visões;
- Independência de integridade;
- Independência de distribuição;
- Regra não subversiva.

2.9.2 SQL Server

Para Microsoft (2023d, n.p.), “O Microsoft SQL Server é um sistema de gerenciamento de banco de dados relacional (RDBMS). Aplicativos e ferramentas se conectam a uma instância ou banco de dados do SQL Server e se comunicam usando Transact-SQL (T-SQL)”.

Assim como outras tecnologias de SGBD (Sistema de Gerenciamento de Banco de Dados) SQL Server é principalmente construído em torno de uma estrutura de tabela baseada em linhas que conecta elementos de dados relacionados em diferentes tabelas, evitando a necessidade de armazenar dados de forma redundante em múltiplos locais dentro de um banco de dados. O modelo relacional também oferece integridade referencial e outras restrições de integridade para manter a precisão dos dados. Essas verificações fazem parte de uma adesão mais ampla aos princípios de atomicidade, consistência, isolamento e durabilidade, conhecidos coletivamente como as propriedades ACID, e são projetadas para garantir que as transações de banco de dados sejam processadas de forma confiável. (MICROSOFT, 2023d).

O SQL Server funcionou exclusivamente no ambiente Windows por mais de 20 anos. Em 2016, a Microsoft o tornou disponível no Linux. O SQL Server 2017 se tornou amplamente disponível em outubro de 2016 e passou a ser executado tanto no Windows quanto no Linux. (MICROSOFT, 2023d).

2.10 C#

O C# (pronunciado como "C Sharp") é uma linguagem de programação contemporânea, orientada a objetos e altamente tipada. Esta linguagem possibilita que desenvolvedores criem uma variedade de aplicativos seguros e resistentes que operam no ambiente .NET. O C# possui suas origens na família de linguagens C, sendo imediatamente reconhecível por programadores familiarizados com C, C++, Java e JavaScript. (MICROSOFT, 2023c).

Ainda conforme a Microsoft (2023c), o C# é uma linguagem de programação que se baseia nos princípios da programação orientada a objetos e na criação de componentes. Incorporando elementos da linguagem que diretamente respaldam esses conceitos, o que a torna uma escolha natural para o desenvolvimento e utilização de componentes de software. Desde o seu surgimento, o C# tem continuamente integrado funcionalidades para dar suporte a novos cenários de trabalho e abraçar práticas de design de software emergentes.

2.11 SWAGGER

A abordagem fornecida pelo Swagger, também conhecido como OpenAPI, desempenha um papel fundamental na documentação e desenvolvimento de APIs. Conforme ressaltado pela própria documentação do Swagger, essa especificação oferece um padrão para descrever APIs

RESTful, permitindo, assim, a colaboração eficaz entre equipes multidisciplinares. Através da definição padronizada de *endpoints*, parâmetros, respostas e autenticação, o Swagger promove uma compreensão clara da API, facilitando a geração automatizada de código em diversas linguagens (SWAGGER, 2023).

A capacidade do Swagger de manter uma fonte única de documentações sobre a API e melhorar a clareza da comunicação entre as partes interessadas é o que o torna muito utilizado. Além disso, destaca que a ferramenta possibilita a realização de testes automatizados para garantir a qualidade da API. Ao adotar o Swagger, as equipes podem alcançar uma documentação consistente e uma comunicação fluida, contribuindo para a qualidade e sucesso do desenvolvimento de APIs. (SWAGGER, 2023).

2.12 JWT

O JSON Web Token (JWT) é um padrão aberto, com o objetivo de estabelecer um método compacto e independente para transmitir com segurança informações do usuário entre cliente e servidor. Essas informações são encapsuladas em um objeto JSON e podem ser enviadas através de um cabeçalho HTTP. O token JWT resultante é armazenado no dispositivo do usuário, e suas informações são protegidas por meio de criptografia usando um segredo conforme a Figura 4. Isso é realizado por meio de algoritmos HMAC (*Hash-based Message Authentication Code*) ou por meio de pares de chaves pública e privada. A autenticidade do *token* é garantida, permitindo que suas informações sejam verificadas em cada solicitação (JONES et al., 2015).

Figura 4 – Exemplo de Json Web Token

O diagrama ilustra a estrutura de um JWT token, dividido em três partes principais:

- HEADER: ALGORITHM & TOKEN TYPE:** Contém um objeto JSON com os campos "alg" (algoritmo) e "typ" (tipo). Exemplo: `{ "alg": "HS256", "typ": "JWT" }`
- PAYLOAD: DATA:** Contém um objeto JSON com as informações do usuário. Exemplo: `{ "sub": "1234567890", "name": "John Doe", "iat": 1516239022 }`
- VERIFY SIGNATURE:** Mostra a fórmula para gerar a assinatura HMACSHA256: `HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), your-256-bit-secret)`. O texto "secret base64 encoded" indica que o segredo deve ser codificado em base64.

À esquerda do diagrama, um exemplo de token JWT completo é exibido em uma caixa de código:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MzkwMjQ5Lm51LnR5cCI6IkpXVCJ9.eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MzkwMjQ5Lm51LnR5cCI6IkpXVCJ9.
```

Fonte: JWT (2023, n.p.).

Diversas bibliotecas que implementam o padrão JWT estão disponíveis para várias linguagens de programação, incluindo .NET, Python, NodeJs, Java, JavaScript, Perl, Ruby e PHP. Essas bibliotecas são utilizadas tanto para autenticação de usuários, onde o *token* é enviado a cada solicitação para permitir ou negar o acesso a determinadas rotas, serviços ou recursos, quanto para garantir a transmissão segura de informações. Isso ocorre porque o conteúdo do *token*, devido à sua assinatura criptografada, pode ser verificado quanto a adulterações (JWT, 2023).

3. METODOLOGIA DA PESQUISA

O presente trabalho caracteriza-se como pesquisa aplicada e descritiva, pois foi desenvolvido um protótipo de sistema web.

O trabalho buscou desenvolver um sistema para otimizar os processos de um escritório de soluções agrícolas.

Após a finalização do levantamento bibliográfico, foi realizado um estudo mais aprofundado sobre o desenvolvimento de aplicações com a arquitetura cliente servidor, utilizando as ferramentas e padrões de desenvolvimentos modernos, visando deixar a aplicação relevante por mais tempo, evitando futuros retrabalhos, migrações para novos padrões e facilitando possíveis adições futuras ao trabalho. Na revisão da literatura estão detalhadas as tecnologias, linguagens de programação e *frameworks* que são utilizadas para desenvolvimento do protótipo.

3.1 MAPEAMENTO DE CENÁRIO

O principal objetivo do mapeamento de cenário foi de identificar o atual fluxo de trabalho do escritório de soluções agrícolas ao iniciar uma nova proposta. após o cliente entrar em contato são coletadas informações pessoais, como nome, telefone, endereço, documentos pessoais, geolocalização do terreno que será cultivado, etc. E também são coletados documentos necessários para a elaboração da proposta, como matrícula do terreno, análise física e química de solo, CAR (Cadastro Ambiental Rural), contrato de arrendamento/comodato, DAP (Declaração de Aptidão ao Pronaf), ITR (Imposto sobre a Propriedade Territorial Rural), etc. Após isso são elaborados outros documentos necessários para cada proposta, como Croqui de localização, CND (Certidão Negativa de Débitos), Recomendação de fertilizantes e corretivos com base na análise do solo, Laudo de Ater (Assistência Técnica e Extensão Rural), além da própria proposta.

A proposta é elaborada detalhando todas as informações acerca do custeio agrícola, seja a área de plantio, a cultura que será cultivada, a capacidade de pagamento do cliente, o avalista do financiamento, etc. Na Figura 5 é apresentado o modelo atual utilizado para as propostas, onde são cadastradas todas as informações para a realização do financiamento, como Proponente, Avalista, Imóveis, Orçamento, etc. E na Figura 6 pode ser observado o modelo de

capacidade de pagamento utilizado atualmente, onde é descrito todo o fluxo de caixa do proponente e a capacidade do mesmo arcar com esse novo financiamento.

Figura 5 – Proposta de Financiamento Simplificado

Agência:		Município:		Data:					
Proponentes - Nome(s):		Telefone:	CPF:	Conta:	Proposta				
Escritório de:		Fone:							
Responsável pelo Projeto:				CFTA -					
Finalidade: Custeio para o Plantio de MILHO 1º SAFRA									
1. Orçamento de aplicação									
Especificação		Valor Unitário		Total em R\$					
MILHO 1º SAFRA		3,90 ha		R\$ 4.263,21					
Total em R\$				16.626,52					
Tipo de Agricultura:		Produtividade Média (kg/ha):		8.000					
Sistema de Plantio:		Produtividade Esperada (kg/ha):		10.000					
2. Usos e Fontes (Valores em Reais):									
Total do Orçamento		16.626,52		Valor Astec Finan. = R\$ 332,53 Valor Total Financiado = R\$ 16.959,05					
Do financiamento:		16.626,52		100,0% Linha de crédito PRONAF CUSTEIO					
Recurso próprio:		0,00		Origem rec. Próprio => Não se aplica					
Prazo:		12 Meses n°.parcel.		1 Carência: 0 Meses					
Prestações:		Taxa de juros => 3,0%		Vencido: 7/27/2023					
3. Garantias:									
AVALISTA:				CPF:					
4. Imóveis beneficiados									
Nome Imóvel		Proprietário		Usos da Terra					
Identificação do Imóvel		Área Total		Usos da Terra					
Matrícula		Utilizada		Agrícola					
Localização		Própria		Pecuário (pastagem)					
Município		Arrendada		Discriminação das Matrículas					
Distrito		% de área própria		Latitude:					
		0%		Longitude:					
5. Histórico Agrícola									
Produtos		Antepenúltima		Penúltima		Última Safra		Resultado Última Safra	
unidade		Produtivid. kg/ha		Área ha		Produtivid. kg/ha		Área ha	
Beterraba		#ERRO!							
Tomate		#ERRO!							
Feijão		#ERRO!							
Milho		8.500		3,9		8.500		3,9	
Mand. Salsa		kg							
Cebola		#ERRO!							
		35.000		5,0		35.000		5,0	
		total ha		8,90		total ha		8,90	
		total R\$				total R\$		244.596,75	
6. Produção Prevista									
Produtos		Ano 1		Ano 2		Ano 3		seguintes (n)	
unidade		Produtivid. kg/ha		Área ha		Produtivid. kg/ha		Área ha	
Beterraba		#ERRO!							
Tomate		#ERRO!							
Feijão		#ERRO!							
Milho		10.000		3,9		10.000		3,9	
Cebola		#ERRO!							
Mand. Salsa		kg							
		40.000		5,0		40.000		5,0	
		Total ha		8,90		Total ha		8,90	

Fonte: acervo do autor (2023).

Figura 6 – Capacidade de pagamento

FLUXO DE CAIXA/CAPACIDADE DE PAGAMENTO													
1. VALOR A FINANCIAR											Ano 1		
2. RECURSOS PRÓPRIOS											0,00		
3. Valor total do orçamento											16.626,52		
4. RECEITAS AGRÍCOLAS													
	1g ano	produto	Valor	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5	Ano 6	Ano 7	Ano 8	Ano 9	Ano 10
Beterraba	0	kg	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Tomate	0	kg	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Feijão	0	kg	1,493	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Milho	3,9	kg	0,00	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40	20.849,40
5. OUTRAS RECEITAS AGROPECUÁRIAS													
6. TOTAL DAS RECEITAS											#ERRO!		
7. DESPESAS													
Valor do orçamento proposto (financiamento + rec. prop)											16.626,52		
8. DESPESAS AGRÍCOLAS													
	1g ano	unidade	V. unit.	Ano 1	Ano 2	Ano 3	Ano 4	Ano 5	Ano 6	Ano 7	Ano 8	Ano 9	Ano 10
Beterraba	0	ha	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Tomate	0	ha	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Feijão	0	ha	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Milho	3,9	ha	1.111,00	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90	5.902,90
	0	ha	2.207,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00	11.395,00
9. DÍVIDAS (no Banco do Brasil e terceiros)													
10. CUSTOS FIXOS											#ERRO!		
11. TOTAL DAS DESPESAS											#ERRO!		
12. CAPACIDADE DE PAGAMENTO													
MARGEM DISPONÍVEL TOTAL - (MDT)											#ERRO!		
MARGEM DISPONÍVEL - MDI - % de MDT											70,00%		
13. COMPROMISSOS COM O FINANCIAMENTO													
AMORTIZAÇÃO DO CAPITAL - AC											#ERRO!		
PAGAMENTO DOS JUROS											#ERRO!		
14. SALDO DO EXERCÍCIO											#ERRO!		
15. SALDO ACUMULADO											#ERRO!		

Fonte: acervo do autor (2023).

Após a elaboração de toda a documentação e cadastros necessários nos sistemas dos bancos/cooperativas, é solicitada a assinatura do cliente em alguns documentos e enviado à instituição financeira para análise. Neste momento a cobrança pelos serviços pode ser realizada de algumas formas, onde o cliente pode pagar diretamente para o escritório, ou solicitar à instituição financeira que desconte o valor da ASTEC (Assistência Técnica) do valor do projeto, dizendo que a ASTEC será financiada. Porém o controle de contas a receber é realizado por meio de planilhas e livro caixa, sem um sistema específico para o controle financeiro.

Com a proposta finalizada e aprovada junto à instituição financeira, existe a possibilidade da realização de laudos de acompanhamento da lavoura em que foi realizado o financiamento. No caso de custeios são três laudos e no caso de investimentos é apenas um laudo. Esses laudos podem ou não ser obrigatórios, pois algumas instituições financeiras obrigam a realização para que posteriormente fique facilitado o acionamento do PROAGRO caso necessário. Mas mesmo sem obrigatoriedade o cliente poderá solicitar os laudos, pagando um valor maior pelos serviços.

Esses laudos de acompanhamento são realizados em três períodos da safra, um no início, um no meio, e um no final, antes da colheita. No laudo são coletadas informações sobre o estado da lavoura, previsões de produtividade, insumos utilizados, pragas e doenças identificadas, etc. Como pode ser identificado na Figura 7 da planilha de laudo utilizada atualmente pela empresa.

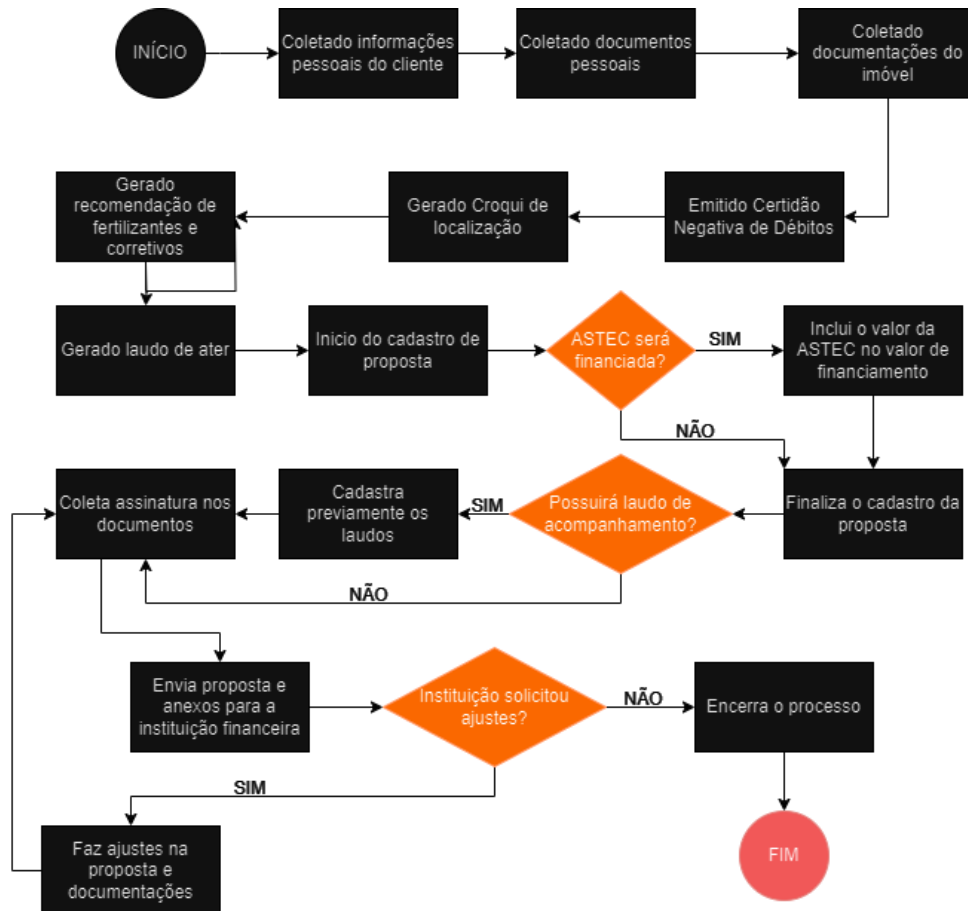
Figura 7 – Laudo de Supervisão e Assistência Técnica

LAUDO DE SUPERVISÃO E ASSISTÊNCIA TÉCNICA						Nº 01 (Emergência)
Identificação						
Agente	Banco Grana Forte	Data da Vistoria:	12/08/2006			
Agência	1234 - Casca Dura	Data do Laudo:	15/08/2006			
Cliente	Fulando de tal ciclano beltrano	CPF	111.222.333-44			
Cultura	Soja (em grãos)	Município	Cascavel/PR			
Imóvel 1	12.345					
Imóvel 2	0					
Imóvel 3	0					
Imóvel 4	0					
Aplicação do crédito e recomendações						
Varied/Híbrido	Área ha	Época plantio	Estágio	Stand	Desenvolvimento	Uniformidade
				BOM	bla lba	Bom
						Colheita fev mar/06
Observações:						
Área cultivada corresponde a financiada?	sim					
Lavoura plantada corresponde a financiada?	sim		bla lba			
Croqui identifica a área?	sim		bla lba			
Possui área com recursos próprios?	não		bla lba			
Época de plantio atendeu ao Zoneamento agrícola?	sim		bla lba			
Operações realizadas, insumos aplicados e manifestação quanto à eficiência técnica						
Insumos				Serviços		
Produto utilizado	Quantidade	Unidade	Data	Operação realizada	Quantidade	Unidade
Matatudo (Inseticida)	2,00	l	18/03/06	Pulverização (Inseticida)	1	h/maq
Data: 18/03/2006						
OBS.:						
Aspectos técnicos e recomendações						
Diagnósticos	Nível	Área afetada	Especificações	Fazer controle?	Alterou a produtividade?	
Erva daninha	baixo	20,00	Leiteiro xxxxxxxxxxx	não	não	
Pragas	ausente			não	não	
Doenças	moderado	20,00	Doença Brava xxx	não	sim	
Def. mineral	ausente			não	não	
Erosão	ausente			não	não	
OBS.:						
Conservação de solos						
Existente?	Sim	Análise de Solo	Atualizada			
Adequada?	Sim	Condições Climáticas	Satisfatórias			
Sinistros						
Ocorrência de sinistros	Não	Evento ocorrido apresenta cobertura pelo Proagro?			Sim	
Evento ocorrido	Chuva na colheita	Cliente deve acionar o Proagro?			Sim	
Data de ocorrência		Cliente ciente que deve aguardar pericia antes da colheita?			Sim	
Conclusão						
Crédito aplicado corretamente	sim		Produtividade do plano (kg/ha)	2.800		
Mutuário vem atendendo as recomendações	sim		Produtividade obtida (kg/ha)	2.850		
Liberação da próxima parcela	sim		Local de entrega do produto			
Situação do empreendimento	normal		Qualidade do produto			
<p style="text-align: center;">Fulando de tal ciclano beltrano 111.222.333-44</p> <p style="text-align: center;">Fulando de Tal UF-CREA PR-12345/D 0</p>						

Fonte: acervo do autor (2023).

A Figura 8 resume o mapeamento de cenário para auxiliar na visualização de todo o processo que foi descrito anteriormente. Inicialmente são coletadas as informações do cliente e seus documentos, bem como as documentações do imóvel que será beneficiado com o financiamento. Após isso são gerados alguns documentos como Certidão Negativa de Débitos, Croqui, Recomendação de Fertilizantes e Corretivos e Laudo de Ater. Com isso será iniciado o cadastro da proposta, onde deverá ser verificado se a ASTEC será financiada e se possuirá laudo de acompanhamento, caso possua laudo, deverá ser cadastrado os laudos previamente a realização da vistoria em si. Após finalizar será coletada a assinatura do cliente nos documentos e na proposta e todo o dossiê será encaminhado a instituição financeira, que poderá solicitar ajustes na proposta.

Figura 8 – Diagrama de Atividade – Mapeamento de Cenário.



Fonte: acervo do autor (2023).

Com base nessa pesquisa foi definido que a aplicação seria feita na IDE Visual Studio, com a linguagem C#, integrando com o banco de dados SQL Server por meio do Entity Framework e design baseado no conjunto de boas práticas definidos pela Microsoft na documentação mais atualizada das tecnologias. Além disso, será utilizado o ReactJS para desenvolver a parte visual da aplicação. Essas escolhas foram feitas por serem tecnologias de grandes empresas que atualizar frequentemente as ferramentas com melhorias.

Considerando que não foram encontrados sistemas com características similares ao protótipo proposto, e garantir que o protótipo possua os recursos adequados, foi realizado um mapeamento de cenário na empresa “Lopes Projetos Agrícolas”, visando entendimento do fluxo de trabalho e das suas necessidades. O mapeamento do cenário foi realizado no dia 27 de agosto de 2023, por meio de uma entrevista em profundidade não estruturada com o Sr. Alan Fernando Lopes, proprietário da Lopes Projetos Agrícolas. No Anexo I consta o termo de autorização para uso de nome empresarial e dados da empresa.

4. PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS

Neste capítulo são apresentados todos os aspectos técnicos do processo de análise e implementação do protótipo.

4.1 ANÁLISE

Condizente ao apontado na revisão da literatura entender as necessidades do cliente é um dos objetivos dos processos previstos na engenharia de software. Tendo isso como base a análise é importante para definir o que será desenvolvido e como será. Uma análise adequada, acarretará uma compreensão clara de todos os processos, assim tornando mais fácil o desenvolvimento e futuras manutenções.

4.1.1 Visão Geral do Protótipo

O protótipo tem como objetivo principal auxiliar no gerenciamento de propostas em escritórios agrícolas, disponibilizando um sistema para substituir as planilhas que são atualmente utilizadas, além de facilitar o compartilhamento de informações entre as filiais.

Ao utilizar o sistema, o usuário poderá criar as propostas agrícolas para seus clientes, cadastrando e reutilizando todas as informações necessárias como o cadastro do proponente, avalistas, imóveis, culturas, etc. E posteriormente controlar os laudos de acompanhamento da proposta.

Também poderá ser acompanhado o fluxo da proposta, já que após o cadastro ela ficará com o status “Em elaboração”, até seja finalizada. Em seguida a proposta poderá ser atualizada para “Aguardando laudo de acompanhamento” caso existam laudos pendentes ou “Encerrada” caso não existam laudos pendentes. Ao finalizar todos os laudos, a proposta será encerrada automaticamente. Quando a proposta estiver encerrada ela não poderá mais ser alterada. Contudo, será possível retornar o status para “Em elaboração”, caso seja necessário ajustar algum registro. Ao finalizar a proposta ela poderá ser impressa no sistema para ser anexada junto com os outros documentos e encaminhada a instituição financeira que dará prosseguimento no financiamento do cliente.

Todos os registros possuirão registros de auditoria (logs) de quando incluído, alterado ou excluído, bem como informações sobre o usuário que realizou a operação.

4.1.2 Requisitos

Nesta seção estão elencados os requisitos para o funcionamento do protótipo. Esses requisitos serão desenvolvidos para que seja atendida a necessidade do cliente, e também, cumprindo aquilo que foi proposto. O Quadro 4, apresenta os requisitos funcionais necessários para o desenvolvimento do protótipo e as respectivas regras de negócio identificadas na última coluna estão detalhadas no Quadro 7.

Quadro 4 – Requisitos Funcionais

Número	Nome	Descrição	Regras de Negócio
RF01	Login	O protótipo deverá permitir que o usuário realize o login para acessar determinados recursos da aplicação. Para realizar o acesso será necessário informar o e-mail e senha.	
RF02	Logout	O protótipo deverá permitir o usuário encerrar seu acesso.	
RF03	Menu	O protótipo deverá possuir um menu lateral à esquerda que possibilite o acesso aos recursos da aplicação a qualquer momento.	
RF04	Cadastro de Usuários	O protótipo deve permitir o cadastro de usuários. Sendo necessário informar o e-mail, senha e confirmação da senha.	RN01
RF05	Cadastro de Pessoas	O protótipo deve permitir que o usuário cadastre as pessoas, que poderão ser utilizadas nos demais processos, sendo clientes, donos de terrenos, etc.	RN01
RF06	Cadastro de Proposta	O protótipo deve permitir que o usuário cadastre suas propostas, definindo o tipo, proponente, avalista, imóveis, cultura, e se irá ou não possuir laudo de acompanhamento.	RN02 RN05 RN06 RN07
RF07	Cadastro de Tipo de Proposta	O protótipo deve permitir o cadastro de Tipo de proposta, onde poderá ser definido o nome, observação e tipos de documentos obrigatórios para a proposta.	
RF08	Cadastro de Filial	O protótipo deve permitir que o usuário cadastre as filiais da empresa, para que seja selecionada a filial ao cadastrar uma nova proposta.	RN02
RF09	Cadastro de Laudo de acompanhamento.	O protótipo deve permitir o cadastro de Laudos de acompanhamento para as propostas, onde será obrigatório o cadastro caso a proposta esteja com a opção “Possui laudo de acompanhamento” marcada.	RN01 RN07

RF10	Cadastro de Documentos	O protótipo deve permitir o cadastro de documentos, que possuirá um Tipo de documentação, e poderá estar vinculada a uma proposta, cliente ou imóvel.	RN08
RF11	Cadastro de Culturas	O protótipo deve permitir o cadastro de culturas, como Milho, Cebola, Beterraba, etc. Onde poderá ser indicado o nome e preço de mercado atual, sendo o preço por Kg.	RN03
RF12	Impressão de Proposta	O protótipo deve permitir a impressão da proposta em modelo padronizado, trazendo todas as informações em relatório que poderá ser impresso posteriormente. Deverá também trazer espaços reservados para a assinatura do cliente e do técnico responsável pela proposta.	
RF13	Cadastro de Imóveis	O protótipo deverá permitir o cadastro de imóveis, onde deverá ser indicado o nome, dono do terreno, endereço, roteiro de acesso, documentos, arquivo KML (<i>Keyhole Markup Language</i>) com dados geográficos.	RN04 RN05
RF14	Cadastro de Permissões de Usuário	O protótipo deverá permitir o cadastro de permissões de usuário, sendo possível informar qual a rotina e quais ações o usuário possui permissão para realizar.	RN01
RF15	Cadastro de Imóvel da Proposta	O protótipo deverá permitir o cadastro de Imóvel do projeto, para definir quais imóveis e qual a área do imóvel que será utilizada no projeto.	RN04 RN05
RF16	Cadastro de Tipo de Documentação	O protótipo deverá permitir o cadastro de tipo de documentação, como Certidão de casamento, RG, etc. Apenas definindo o nome do tipo.	RN08
RF17	Documentação Obrigatória do Tipo de Proposta	O protótipo deverá permitir o cadastro de tipos de documentação no tipo de proposta para obrigar o vínculo ao criar uma nova proposta.	RN08

Fonte: acerto do autor (2023).

O Quadro 5, apresenta os requisitos funcionais opcionais para o desenvolvimento do protótipo. Estes requisitos poderão ser implementados em versões futuras da aplicação. Todos os requisitos listados neste Quadro não fazem parte da versão inicial.

Quadro 5 – Requisitos Funcionais Opcionais

Número	Nome	Descrição
RN18	Controle Financeiro	O protótipo poderá permitir o controle financeiro de contas a receber das propostas, gerando parcelas para os clientes ao finalizar cada proposta. Será controlado a baixa, juros, formas e condições de pagamento.
RN19	Emissão de NFSe	O protótipo poderá permitir a geração e emissão via integração de NFSe (Nota Fiscal de Serviço eletrônica) juntamente com o <i>webservice</i> da prefeitura, para pagamento dos tributos relacionados a prestação de serviços.

RN20	Cadastro de Croqui	O protótipo poderá permitir o cadastro de croqui de localização, onde será cadastrado a imagem da área do croqui e glebas geográficas, bem como o vínculo ao imóvel referenciado.
RN21	Cadastro de Laudo de Proagro	O protótipo poderá permitir o cadastro de laudo de PROAGRO, onde será cadastrado toda a vistoria realizada pelo técnico a lavoura bem como imagens, gerando uma impressão do laudo por completo.
RN22	Área do Cliente	O protótipo poderá possuir uma área do cliente, onde o cliente poderá acessar para verificar a situação da sua proposta, apenas consultando pelo CPF, sem necessidade de contato com o escritório.
RN23	Cadastro de Capacidade de Pagamento da Proposta	O protótipo poderá permitir o cadastro de capacidade de pagamento da proposta, onde poderá informado as receitas do proponente para garantir que o mesmo possua a capacidade de arcar com o financiamento.
RN24	Visualização de trilha de auditoria	O protótipo poderá dispor de uma aba para visualização da trilha de auditoria dos registros, onde possuirá logs de ações realizadas nos registros.

Fonte: acerto do autor (2023).

O Quadro 6, tem como finalidade apresentar os requisitos não funcionais, que são características gerais da aplicação, sendo requisitos também que envolvem desempenho, usabilidade e segurança.

Quadro 6 – Requisitos Não Funcionais

Número	Descrição
RNF01	O protótipo deve ser desenvolvido com C# e React JS.
RNF02	As telas de cadastros devem possuir tratamento para valores nulos no <i>front-end</i> .
RNF03	Todos os processos devem possuir autenticação e autorização de usuário.
RNF04	O protótipo deverá implementar JWT para autenticação e autorização, expirando em 2 horas.
RNF05	Para o armazenamento de dados deverá ser utilizado SQL Server.
RNF06	Para gerenciamento do banco de dados deverá ser utilizado o Entity Framework.
RNF07	O controle de auditoria deverá ser padronizado, indicando o usuário de cadastro/alteração do registro e a data/hora.
RNF08	O <i>front-end</i> deve tratar as permissões do usuário nos componentes visuais
RNF09	O protótipo deverá possuir responsividade em suas telas, garantindo a possibilidade de utilização por aparelhos móveis.

Fonte: acerto do autor (2023).

Quadro 7 – Regras de Negócio

Número	Descrição
RN01	Apenas usuários que sejam técnicos agrícolas podem criar/alterar laudos de acompanhamento.
RN02	Todos os dados devem ser compartilhados entre as filiais, porém cada proposta deve ter uma filial de origem para controle.
RN03	Toda cultura (cebola, milho, beterraba, etc.) precisa ter seu valor de mercado em kilograma atualizado.

RN04	Um imóvel pode ser utilizado em diversas propostas diferentes, mas somente com datas de plantio e colheita que não se sobrepõem.
RN05	Uma proposta poderá ter mais de um imóvel.
RN06	Uma proposta poderá ter apenas uma cultura.
RN07	Uma proposta só poderá ficar encerrada caso não exista nenhum laudo de acompanhamento pendente.
RN08	Uma documentação poderá ter origem de uma proposta, pessoa ou imóvel.

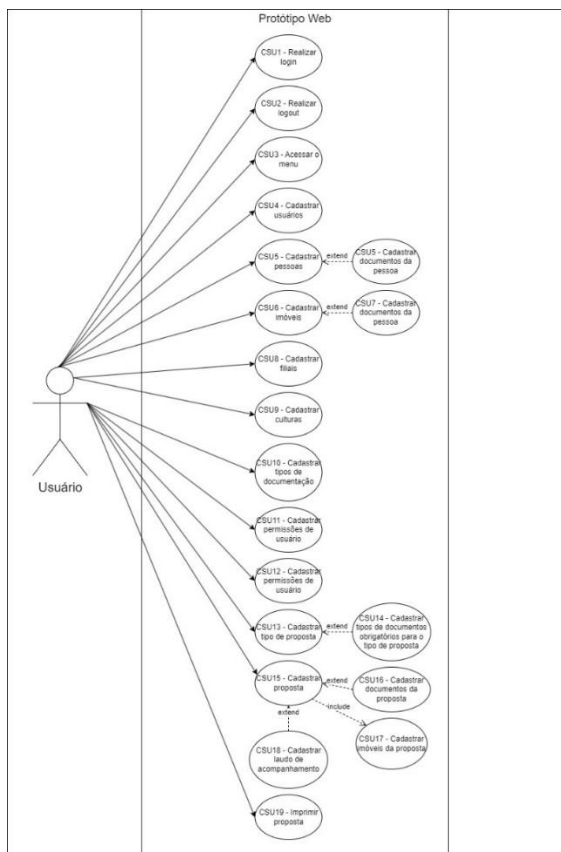
Fonte: acerto do autor (2023).

4.1.3 Diagramas

Para facilitar a compreensão dos requisitos elencados e das funcionalidades do protótipo foram elaborados alguns diagramas. Os diagramas representam os recursos que estão disponíveis para os usuários, bem como o passo a passo para executar a atividade principal proposta por esse protótipo.

A Figura 9 representa um diagrama de caso de usos, onde apresenta as funcionalidades do protótipo, bem como demonstra sua disponibilidade aos atores do sistema, neste caso apenas o usuário.

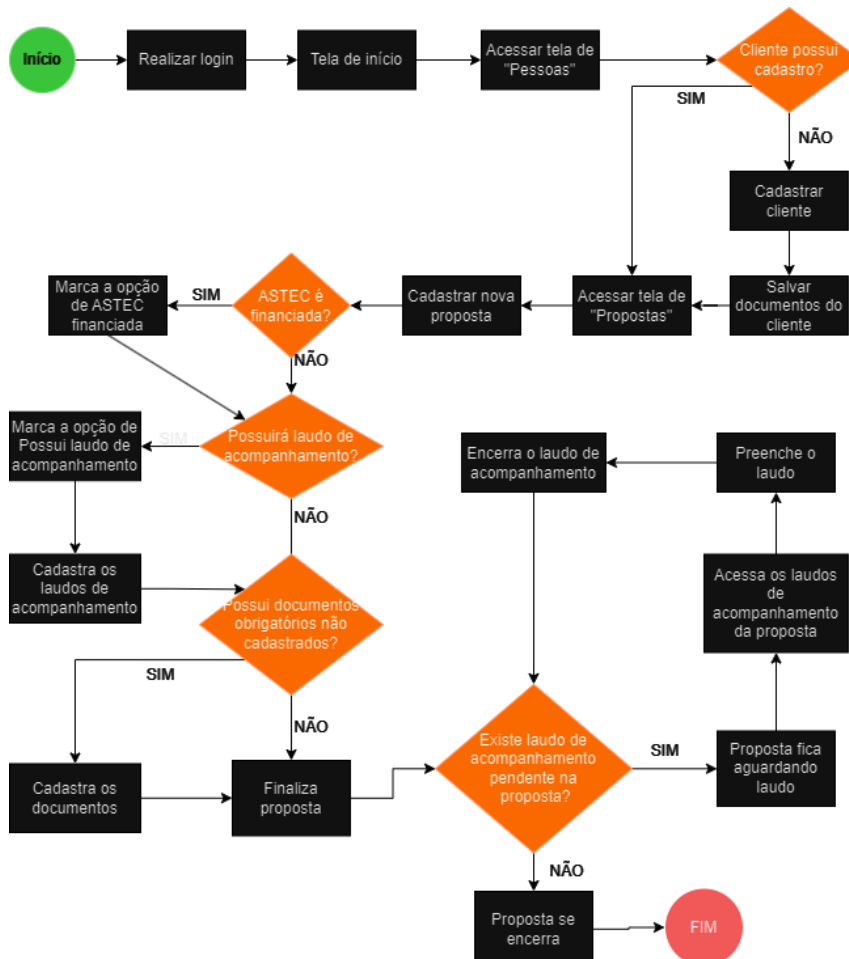
Figura 9 – Diagrama de Caso de Uso



Fonte: acervo do autor (2023).

Conforme pode ser verificado na Figura 10, existem dois fluxos principais exemplificados, um para o cadastro de proposta e outro para o preenchimento dos laudos de acompanhamento, que ocorrerão posteriormente ao cadastro da proposta. Nos dois fluxos inicialmente o usuário faz o login, no cenário de uma nova proposta o mesmo verifica se o cliente possui cadastro, caso não possuir realiza o cadastro do mesmo e após já poderá seguir para cadastrar a proposta, onde são preenchidas as informações e passado por algumas validações, como se possuirá laudos de acompanhamento, se existem documentações pendentes na proposta, etc. Após o cadastro a proposta poderá ficar em duas situações, encerrada ou aguardando laudos de acompanhamento caso existir algum laudo pendente.

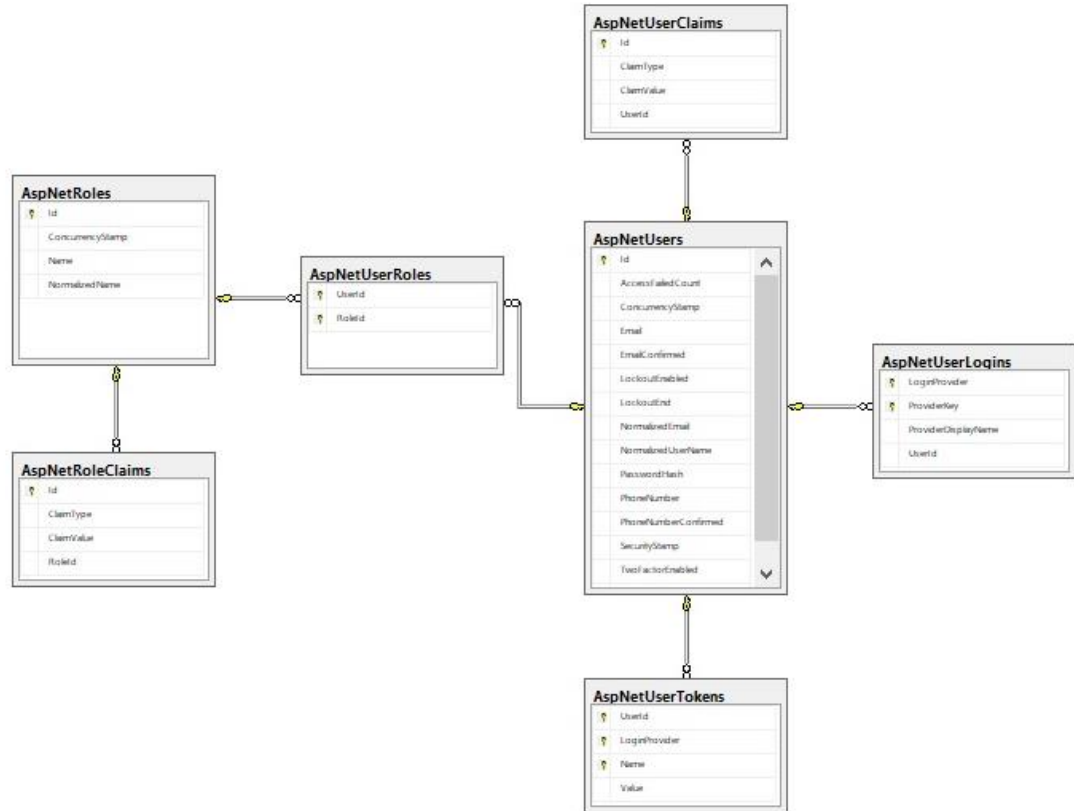
Figura 10 – Diagrama de Atividade – Cadastro de Proposta



Fonte: acervo do autor (2023).

No segundo fluxo de preenchimento de laudo de acompanhamento, o usuário irá acessar a proposta e seguirá para os laudos pendentes da proposta, onde preencherá as informações e

Figura 12 – Diagrama de Entidade Relacionamento – Identity Core



Fonte: Microsoft (2023a, n.p.).

Após a modelagem do banco de dados, foi utilizado o Entity Framework para gerar as classes, e assim pode ser realizada toda a manutenção do banco e do código sem gerar *scripts* manuais, apenas com migrações que o próprio *framework* gera.

Conforme pode ser visto na Figura 13 do diagrama de classe, cada coluna da tabela se tornou uma propriedade, bem como cada chave estrangeira gerou mais de uma propriedade, pois assim o Entity Framework consegue realizar as consultas sem necessidade de comandos de banco de dados puros no código, apenas trabalhando com as classes.

Para criar os componentes visuais foi utilizado o Material UI, ou MUI, uma biblioteca que disponibiliza diversos componentes previamente preparados e prontos para serem utilizados dentro dos projetos, já possuindo uma excelente responsividade em relação às diferentes telas que podem vir a acessar a aplicação, como a diferença entre telas de dispositivos *Desktop*, *Mobile* e *Tablets*. Esses componentes são traduzidos em elementos HTML pelo React JS, sendo assim exibidos na interface do usuário.

Nos formulários dos registros, foi utilizado o “Yup”, que é uma biblioteca JavaScript amplamente utilizada para validação de dados em aplicativos web e móveis. A biblioteca Yup permite que os sejam definidas regras de validação de forma declarativa e altamente configurável, facilitando a validação de campos, como números, datas, *strings* e objetos complexos.

Além da apresentação das interfaces visuais o *front-end* precisa realizar sua comunicação com a API REST que desenvolvemos por meio de requisições, e para realizar esta comunicação utilizou-se a biblioteca Axios, que permite realizar qualquer requisição REST como cliente a um servidor.

A utilização do Axios foi dividida em duas partes, a primeira foi a configuração da comunicação, que se resume a configurar os dados de comunicação da API REST em um arquivo do projeto *front-end* que possui a importação do Axios, a criação de um elemento com a URL da API e sua respectiva exportação. Como complemento desta configuração foi acrescentado um “interceptor” que executa uma ação adicional a cada requisição enviada do cliente ao servidor e que é utilizado para adicionar as informações de autenticação do usuário, se existirem.

Para o lado do servidor (*server-side*) a linguagem utilizada no protótipo foi o C#, uma linguagem com uma capacidade de desenvolvimento em diversas plataformas e com uma gama de bibliotecas grande. Com o intuito de colaborar no desenvolvimento *back-end* foi utilizada para a linguagem C# o *framework* ASP.NET, para criar uma API REST devido à sua capacidade de oferecer segurança, desempenho e facilidade de desenvolvimento. Com recursos integrados de autenticação, suporte a diferentes formatos de dados e documentação automática, simplificando a criação de uma API confiável e eficiente. Além disso, sua escalabilidade e o suporte da comunidade tornam-no uma opção sólida para projetos que requerem uma abordagem RESTful. Além disso, foi utilizado o ORM (*Object Relational Mapping*) Entity Framework, facilitando o gerenciamento de banco de dados integrado.

A autenticação e autorização da aplicação foi realizada utilizando o Identity Core do ASP.NET, permitindo que apenas adicionando uma *data annotation* nas rotas fosse necessário estar passando um JWT nas requisições. Além disso o Identity Core permitiu que fosse validado as permissões do usuário, utilizando a mesma estratégia de *data annotation*, pois entre as entidades disponibilizadas pelo Identity, está a UserClaims, onde é definido o tipo da permissão (Usuário, Pessoa, Proposta, etc.) e as permissões (Excluir, Visualizar, etc.), assim o desenvolvimento foi agilizado sem necessidade de criar um grande contexto de autenticação de autorização.

Para realizar testes e validações do *back-end*, foi utilizado o Swagger, que integrado com o código ASP.NET gera automaticamente uma documentação da API, expondo todas as rotas, modelos, etc. Permitindo realizar requisições de testes sem a necessidade de um *front-end*.

Foi utilizado o banco de dados relacional SQL Server, que integra nativamente com o ecossistema .NET, garantindo a confiabilidade. Para o gerenciamento do banco de dados foi utilizado o SGBD SQL Server Management Studio, por ser o padrão para o banco de dados utilizado.

O JSON foi utilizado para o tráfego de dados no quesito cliente e servidor, e o HTTP foi o responsável pelas requisições entre cliente e servidor.


Para a realização do desenvolvimento do protótipo foram utilizados dois editores de código fonte, o Visual Studio Code para o *front-end* e Visual Studio na versão Community para o *back-end*.

4.2.2 Utilização e Funcionamento

O protótipo trata-se de uma aplicação web responsiva, com um único tipo de acesso para os usuários do sistema. Portanto um usuário cadastrado no sistema poderá realizar ações em todas as rotinas, porém isso dependerá das permissões do seu usuário, já que cada processo possui um cadastro de permissão, definindo se o usuário poderá Visualizar, Editar, Excluir ou Processar.

Ao acessar o sistema será necessário realizar a autenticação do usuário pela tela de login, conforme a Figura 14, informando seu e-mail e senha. Esses dados serão validados no *back-end* e caso tenha sucesso será retornado um JWT para o *front-end*, onde será utilizado em todas as requisições do sistema.

Figura 14 - Login (RF 01)



The image shows a login form titled "Login". It contains two input fields: "Email" with the value "pierre.clopes@unidavi.edu.br" and "Senha" with a masked password represented by ten dots. Below the fields is a blue button labeled "ENTRAR".

Fonte: acervo do autor (2023).

Caso as informações inseridas sejam válidas o usuário será direcionado para a página inicial do sistema, onde terá as informações das propostas e laudos de acompanhamento pendentes conforme a Figura 15.

Figura 15 - Página Inicial e Menu (RF 03)

Fonte: acervo do autor (2023).

Nessa tela inicial é possível visualizar na esquerda o Menu do protótipo, que será o mesmo durante toda ação executada pelo usuário. O Menu disponibiliza acesso aos seguintes recursos: Página inicial, Pessoas, Propostas, Culturas, Imóveis, Usuários, Filiais, Tipos de documentação, Tipos de proposta, Sair /Logout (RF 02) e Alternar tema.

Na tela de Pessoas, serão listadas todas as pessoas cadastradas no sistema, exibindo as colunas de Código, Nome, CNPJ/CPF, Telefone e Ações, conforme a Figura 16. Nas ações serão exibidos os botões de Editar e Excluir. Na parte superior está a barra de ferramentas, exibindo uma barra de pesquisa que irá filtrar as pessoas pelo Nome, e o botão Nova, onde poderá ser cadastrada uma nova pessoa.

Figura 16 - Listagem de pessoas (RF 05)

Código	Nome	CNPJ/CPF	Telefone	Ações
1	Pierre Capistrano Lopes	94.309.389/0001-03		

Fonte: acervo do autor (2023).

Para cadastrar uma nova pessoa será exibido um formulário como pode ser visto na Figura 17, com os campos Nome (obrigatório), Apelido (obrigatório), Tipo (obrigatório), CNPJ/CPF (obrigatório), CFTA, Telefone, RG, Email e Observação, além dos campos do endereço que são Cidade (obrigatório), CEP (obrigatório), Bairro (obrigatório), Número (obrigatório), Complemento e Observação. O CNPJ/CPF será validado de acordo com o Tipo da pessoa, que poderá ser selecionada Física ou Jurídica. O campo CFTA (Conselho Federal dos Técnicos Agrícolas) será obrigatório caso a pessoa esteja marcada como Técnico, e dessa forma poderá ser relacionado como responsável técnico nas propostas.

Figura 17 - Cadastro de Pessoa (RF 05)

Fonte: acervo do autor (2023).

Ao cadastrar a pessoa será possível acessar o botão de Documentações, que irá direcionar o usuário para a listagem de documentações que estão vinculadas a essa pessoa, conforme Figura 18. Nessa tela serão exibidos os campos Código, Nome e Ações. Além da

barra de ferramentas da listagem, exibindo também o botão Voltar, que irá direcionar o usuário de volta para o cadastro da pessoa.

Figura 18 - Listagem de Documentações (RF 10)

Código	Nome	Ações
6	Imposto	
7	Cadastro Ambiental Rural	

Fonte: acervo do autor (2023).

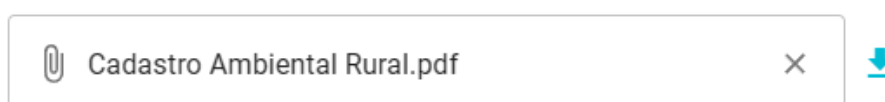
Conforme Figura 19 cadastrar uma nova documentação, será necessário informar no Nome (obrigatório), Tipo de documentação (obrigatório) e o Arquivo (obrigatório). Serão aceitos arquivos com extensão PNG, JPG, PDF e KML.

Figura 19 - Cadastro de Documentação (RF 10)

Fonte: acervo do autor (2023).

Após cadastrar a documentação será possível efetuar o *download* do arquivo anexado, pois será exibido um botão de *download* ao lado do *input* de arquivo, conforme Figura 20.

Figura 20 - Download de Arquivo de Documentação (RF 10)





Fonte: acervo do autor (2023).

Na tela de Culturas, serão listadas as culturas cadastradas no sistema conforme Figura 21, exibindo as colunas de Código, Nome, Preço por Kg e Ações.

Figura 21 - Listagem de Culturas (RF 11)

Listagem de culturas

Pesquisar... NOVA +

Código	Nome	Preço por Kg	Ações
1	Cebola	10	 

Fonte: acervo do autor (2023).

Ao cadastrar uma nova cultura, será exibido o formulário com os campos Nome (obrigatório), Preço por Kg (obrigatório) e Observação, conforme Figura 22, sendo obrigatório apenas os dois primeiros campos.

Figura 22 - Cadastro de Culturas (RF 11)

Nova cultura

SALVAR SALVAR E FECHAR VOLTAR

Geral

Nome Preço por Kg

R\$ Preço por Kg

Observação



Fonte: acervo do autor (2023).

Na tela de Imóveis, serão listados os imóveis cadastrados no sistema, exibindo as colunas de Código, Nome, Matrícula, Área total (ha) e Ações, conforme a Figura 23.

Figura 23 - Listagem de Imóveis (RF 13)

Listagem de imóveis

Pesquisar... NOVO +

Código	Nome	Matrícula	Área total (ha)	Ações
1	Imóvel Teste	20202	20	 

Fonte: acervo do autor (2023).

Ao cadastrar um novo imóvel será aberto o formulário com os campos Nome (obrigatório), Área total (obrigatório), Área agrícola, Área de pastagem, Arquivo KML, Proprietário (obrigatório), Cidade (obrigatório), Matrícula (obrigatório), Latitude, Longitude, Roteiro de acesso (obrigatório) e Observação, conforme a Figura 24. O Proprietário é referente a um registro de Pessoa da base, logo sendo necessário seu cadastro completo.

Figura 24 – Cadastro de Imóvel (RF 13)

Novo imóvel

Geral

Fonte: acervo do autor (2023).

O botão de Documentações ficará disponível assim que o registro for cadastrado, e terá a mesma funcionalidade do botão presente no detalhe de pessoas, conforme foi definido no RF 10.

Na tela de Usuários, serão listados os usuários cadastrados no sistema, exibindo as colunas de Código, Email e Ações, exemplificado na Figura 25.

Figura 25 – Listagem de Usuários (RF 04)

Listagem de usuarios

Código	Email	Ações
f541c92f-f400-4030-8f8e-dfab4cea4c03	pierre.clopes@gmail.com	<input type="button" value="👁"/> <input type="button" value="🗑"/>

Fonte: acervo do autor (2023).

Ao cadastrar um novo usuário deverá ser informado o seu Email (obrigatório), Senha (obrigatório) e a Confirmação da senha (obrigatório), conforme Figura 26.

Figura 26 – Cadastro de Usuário (RF 04)

Novo usuario

Geral

Fonte: acervo do autor (2023).

Após cadastrar um usuário, ficará habilitado o botão Permissões, onde poderá ser acessado a tela de Permissões do usuário, conforme a Figura 27. Nesta tela serão exibidas as colunas Tipo, Valor e Ações.

Figura 27 – Listagem de Permissões de Usuário (RF 14)

Listagem de permissão de usuário

VOLTAR ←

Tipo	Valor	Ações
Pessoa	Visualizar,Excluir,Editar,Processar	✎
Cultura	Editar,Excluir,Processar,Visualizar	✎
Usuario	Excluir,Processar,Visualizar,Editar	✎
Imovel	Excluir,Processar,Visualizar,Editar	✎
Filial	Excluir,Processar,Visualizar,Editar	✎
Documentacao	Excluir,Processar,Visualizar,Editar	✎
Proposta	Editar,Processar,Visualizar,Excluir	✎

Fonte: acervo do autor (2023).

As permissões são geradas automaticamente ao cadastra um novo usuário, portanto não podem ser inclusas, apenas alterado seu valor, conforme Figura 28. Assim ao alterar uma permissão será apresentado os campos Usuário (leitura apenas), Tipo (leitura apenas) e as opções Editar, Excluir, Processar e Visualizar, sendo necessário apenas marcar as opções referentes a permissão que aquele usuário pode ter.

Figura 28 – Cadastro de Permissão de Usuário (RF 14)

Permissão de usuário

SALVAR SALVAR E FECHAR VOLTAR ←

Geral

Usuário: pierre.clopes@gmail.com - Tipo: Pessoa

Editar Excluir Processar Visualizar







Fonte: acervo do autor (2023).

Ao acessar a tela de Filiais, será exibido todas as filiais do sistema, conforme a Figura 29. Exibindo assim as colunas Código, Nome, Sigla e Ações.

Figura 29 – Listagem de Filiais (RF 08)

Listagem de filiais

Pesquisar... NOVA +

Código	Nome	Sigla	Ações
1	Filial	FL	 
2	Filial 2	FL2	 
3	Filial 3	FL3	 

Fonte: acervo do autor (2023).

Conforme a Figura 30, ao cadastrar uma nova Filial, será necessário informar os campos Nome (obrigatório), Pessoa (obrigatório), Sigla (obrigatório) e Observação. A pessoa necessariamente precisa ser uma pessoa jurídica, caso seja selecionado uma pessoa do tipo Física será exibida uma validação, impedindo o cadastro.

Figura 30 – Cadastro de Filial (RF 08)

Nova filial

SALVAR SALVAR E FECHAR VOLTAR

Geral

Nome Pessoa Sigla

Observação





Fonte: acervo do autor (2023).

Na tela de Tipo de documentação estão listadas todos os Tipos de documentação cadastrados no sistema, exibindo as colunas Código, Nome, Sigla e Ações, como pode ser verificar na Figura 31.

Figura 31 – Listagem de Tipo de Documentação (RF 16)

Listagem de tipos de documentação

Pesquisar... NOVO +

Código	Nome	Sigla	Ações
1	Cadastro Ambiental Rural	CAR	 
2	Imposto	ITR	 

Fonte: acervo do autor (2023).

Ao cadastrar um novo tipo de documentação, será necessário informar o Nome (obrigatório), Sigla (obrigatório) e Observação, conforme a Figura 32.

Figura 32 – Cadastro de Tipo de Documentação (RF 16)









A captura de tela mostra o formulário 'Novo tipo de documentação'. No topo, há três botões: 'SALVAR', 'SALVAR E FECHAR' e 'VOLTAR'. Abaixo, há uma seção 'Geral' com três campos de entrada: 'Nome', 'Sigla' e 'Observação'.

Fonte: acervo do autor (2023).

Na tela de Tipo de proposta, são exibidas as colunas Código, Nome e Ações, referente aos Tipos de proposta cadastrados na base, como pode ser visto na Figura 33.

Figura 33 – Listagem de Tipo de Proposta (RF 07)

A captura de tela mostra a listagem de tipos de proposta. No topo, há um campo de busca 'Pesquisar...' e um botão 'NOVO +'. Abaixo, há uma tabela com as seguintes colunas: 'Código', 'Nome' e 'Ações'. A tabela contém quatro linhas de dados.

Código	Nome	Ações
2	Tipo de proposta 1	 
3	Tipo de proposta 2	 
4	Tipo de proposta 3	 
5	Tipo de proposta 4	 

Fonte: acervo do autor (2023).

Conforme a Figura 34, ao cadastrar um novo Tipo de proposta, será necessário informar o Nome (obrigatório), Tipos de documentações obrigatórias e Observação. Apenas o Nome é obrigatório, porém o Tipo de documentação obrigatória é um campo *multiselect*, permitindo assim que sejam selecionados vários tipos, que serão utilizados para validar posteriormente na Proposta, como foi definido no RF 17.

Figura 34 – Cadastro de Tipo de Proposta (RF 07 e RF 17)

Fonte: acervo do autor (2023).

Na tela de Propostas, serão listadas todas as propostas cadastradas no sistema, e como pode ser visto na Figura 35, são exibidas as colunas de Código, Filial, Proponente, Tipo, Data, Cultura e Ações.

Figura 35 – Listagem de Proposta (RF 06)

Listagem de Proposta

Pesquisar... NOVA +

Código	Filial	Proponente	Tipo	Data	Cultura	Ações
1	FL	Pierre Capistrano Lopes	Tipo de proposta	31/10/2023, 16:26:00	CEBOLA	  

Fonte: acervo do autor (2023).

Ao cadastrar uma nova proposta, conforme a Figura 36, será aberto o formulário com os campos Data, Filial (obrigatório), Tipo de proposta (obrigatório), Cultura (obrigatório), Linha de crédito, Proponente (obrigatório), Avalista, Responsável técnico, Possui laudo de acompanhamento, Data de plantio, Data de colheita, Produtividade média, Produtividade esperada, Área financiada, Valor unitário, Valor total do orçamento, Valor do recurso próprio, Origem do recurso próprio, Valor total do financiamento, Valor de ASTEC, Valor total financiado, ASTEC Financiada, Prazo (meses), Número de parcelas, Carência (meses), Taxa de juros, Vencimento e Observação. Os campos que não são obrigatórios durante o cadastro da proposta serão validados posteriormente ao liberar a proposta, isso pode ser realizado pelo botão Liberar no final do formulário, que ficará disponível apenas em propostas que estejam com o status cadastrada.

Figura 36 – Cadastro de Proposta (RF 06)

Nova proposta

Geral

Data: dd/mm/aaaa --:--
 Filial:
 Tipo de proposta:
 Cultura:

Linha de crédito:
 Proponente:
 Avalista:
 Responsável técnico:

Possui laudo de acompanhamento

Produção

Data de plantio: dd/mm/aaaa --:--
 Data de colheita: dd/mm/aaaa --:--
 Produtividade média: kg/ha Produtividade média

 Produtividade esperada: kg/ha Produtividade esperada

Orçamento

Área financiada: ha Área financiada

 Valor unitário: R\$ Valor unitário

 Valor total do orçamento: R\$ Valor total do orçamento

 Valor do recurso próprio: R\$ Valor dos recursos próprios

Origem do recurso próprio:
 Valor total do financiamento: R\$ Valor total do financiamento

 Valor de ASTEC: R\$ Valor de ASTEC

 Valor total financiado: R\$ Valor total financiado

ASTEC Financiada

Financiamento

Prazo (meses):
 Número de parcelas:
 Carência (meses):
 Taxa de juros:
 Vencimento: dd/mm/aaaa --:--

Observação:

Fonte: acervo do autor (2023).

Os campos Valor total do orçamento, Valor total do financiamento e Valor total financiado, não podem ser alterados manualmente, pois serão calculados com base na área financiada, valor unitário, Valor do recurso próprio e Valor de ASTEC, caso a proposta esteja marcada para financiar a ASTEC.

Na parte inferior do formulário estão os botões de navegação para Documentações, Imóveis da proposta e Laudos de acompanhamento. Eles levam para os registros vinculados a essa proposta, no caso das documentações é o mesmo formulário utilizado para documentações das pessoas, porém fazendo a ligação com a proposta, seguindo o que foi definido no RF 10.

Ao clicar no botão Imóveis da proposta, o usuário será encaminhado para a tela onde será exibido todos os imóveis referentes a aquela proposta. Como pode ser visto na Figura 37, são exibidas as colunas de Código, Imóvel, Área e Ações.

Figura 37 – Listagem de Imóveis da Proposta (RF 15)

Listagem de imóveis da proposta

Pesquisar...

Código	Imóvel	Área	Ações
1	Imóvel Teste	10	<input type="button" value="✎"/> <input type="button" value="🗑"/>

Fonte: acervo do autor (2023).



O cadastro de imóvel da proposta serve para definir quais imóveis serão utilizados naquela proposta e qual a área do imóvel que será utilizada. Portanto ao cadastrar um novo imóvel da proposta, será exibido o formulário com os campos Área (obrigatório) e Imóvel (obrigatório), conforme Figura 38. Caso seja informado uma área maior que a área total do imóvel ou que a área financiada na proposta, o *back-end* retornará uma validação.

Figura 38 – Cadastro de Imóvel da Proposta (RF 15)

Fonte: acervo do autor (2023).

Ao clicar no botão Laudos de Acompanhamento no formulário de proposta (vide Figura 36), o usuário será direcionado para a tela que exibe todos os Laudos de acompanhamento cadastrado para aquela proposta. Conforme a Figura 39, são exibidas as colunas Código, Sequencial, Data da vistoria, Data do laudo e Ações.

Figura 39 – Listagem de Laudos de Acompanhamento (RF 09)

Código	Sequencial	Data da vistoria	Data do laudo	Ações
1	0	31/10/2023, 16:31:00	31/10/2023, 16:31:00	 

Fonte: acervo do autor (2023).

Ao cadastrar um novo laudo, será exibido o formulário com os campos Seq, Data da vistoria, Data do laudo, Produtividade do plano, Produtividade obtida, Situação do empreendimento e Observação, além das opções “Área cultivada corresponde a financiada”, “Lavoura plantada corresponder a financiada”, “Croqui identifica a área”, “Possui área com recursos próprios”, “Época de plantio atendeu ao zoneamento agrícola”, “Crédito aplicado corretamente” e “Mutuário vem atendendo as recomendações”. Não será necessário preencher nenhum campo, pois os laudos podem ser cadastrados previamente a realização da vistoria, e somente quando a vistoria for realizada que os dados serão preenchidos. Portanto o laudo também possui um controle de status, podendo estar cadastrado ou encerrado. Conforme a Figura 40, para encerrar será exibido o botão Encerrar na parte inferior do formulário, e ao

Encerrar o laudo será obrigatório preencher todos os campos, ou seja, nenhum campo além das opções poderá estar vazio ao encerrar o registro.

Figura 40 – Cadastro de Laudo de Acompanhamento (RF 09)

Novo laudo de acompanhamento

SALVAR SALVAR E FECHAR VOLTAR

Geral

Seq: Data de vistoria: dd/mm/aaaa --:-- Data do laudo: dd/mm/aaaa --:-- Produtividade do plano: kg/ha Produtividade obtida: kg/ha

Área cultivada corresponde a financiada? Lavoura plantada corresponde a financiada? Croqui identifica a área?

Possui área com recursos próprios? Época de plantio atendeu ao zoneamento agrícola?

Conclusão

Crédito aplicado corretamente? Mutuário vem atendendo as recomendações?

Situação do empreendimento:

Observação:

ENCERRAR → DIAGNÓSTICOS

Fonte: acervo do autor (2023).



O campo sequencial (Seq) será preenchido automaticamente ao cadastrar o registro, pois será o identificador do laudo na proposta.

Na parte inferior do formulário poderá ser acessada a tela de Diagnósticos do laudo, onde o usuário será direcionado para a tela onde exibirá todos os diagnósticos cadastrados para esse Laudo de acompanhamento, conforme a Figura 41.

Figura 41 – Listagem de Diagnósticos do Laudo de Acompanhamento (RF 09)

Listagem de diagnósticos do laudo

VOLTAR ← NOVO +

Código	Diagnóstico	Nível	Ações
2	Erva daninha	Alto	 

Fonte: acervo do autor (2023).

Ao cadastrar um novo diagnóstico, conforme a Figura 42, são exibidos os campos Área afetada (obrigatório), Nível (obrigatório), Diagnóstico (obrigatório) e Observação, além das opções “Alterou a produtividade” e “Fazer controle”.

Figura 42 – Cadastro de Diagnóstico do Laudo de Acompanhamento (RF 09)

Novo diagnóstico do laudo

Geral

Área afetada:
 Nível:
 Diagnóstico:

Alterou a produtividade?
 Fazer controle?

Observação:

Fonte: acervo do autor (2023).

Na proposta também existe o botão Imprimir, que fica disponível tanto nas ações da listagem quanto no final do formulário de detalhe. Ao clicar em imprimir, o formulário será preparado para impressão, bloqueando todos os campos e adicionando uma área para assinatura do Proponente e do Responsável técnico, conforme Figura 43.

Figura 43 – Formulário de Impressão de Proposta (RF 12)

Proposta número 1

Geral

Data:
 Filial:
 Tipo de proposta:
 Cultura:

Linha de crédito:
 Proponente:
 Avaliador:
 Responsável técnico:

Possui laudo de acompanhamento

Produção

Data de plantio:
 Data de colheita:
 Produtividade média:
 Produtividade esperada:

Orçamento

Área financiada:
 Valor unitário:
 Valor total do orçamento:
 Valor do recurso próprio:

Origem do recurso próprio:
 Valor total do financiamento:
 Valor de ASTEC:
 Valor total financiado:

ASTEC Financiada

Financiamento

Prazo (meses):
 Número de parcelas:
 Carência (meses):
 Taxa de juros:
 Vencimento:

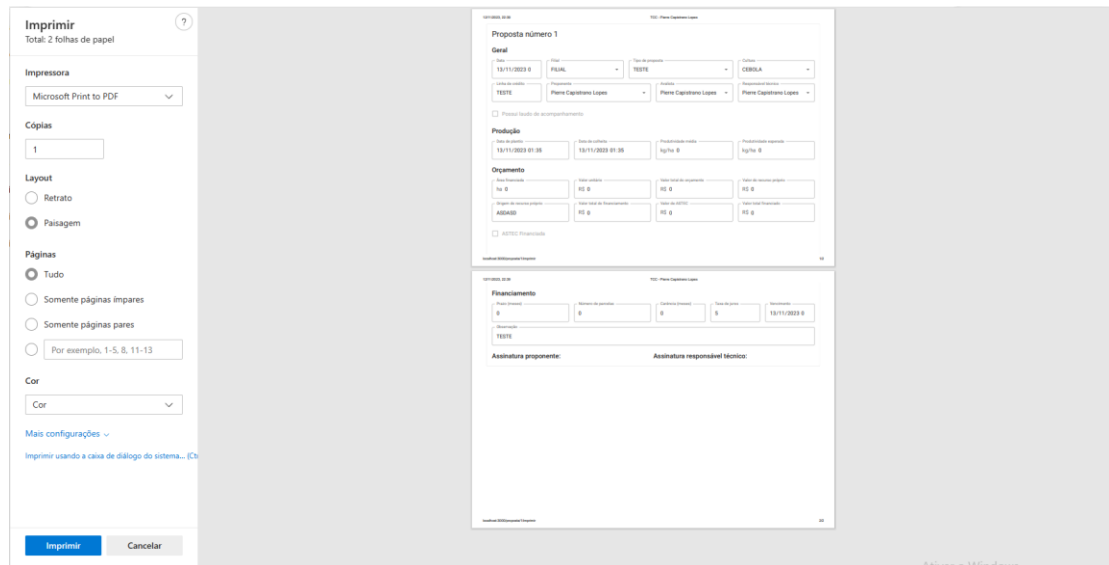
Observação:

Assinatura proponente: _____ Assinatura responsável técnico: _____

Fonte: acervo do autor (2023).

Assim ao dar o comando imprimir no navegador, o menu lateral será escondido e será impresso apenas os dados da proposta, conforme Figura 44.

Figura 44 – Impressão de Proposta (RF 12)



Fonte: acervo do autor (2023).

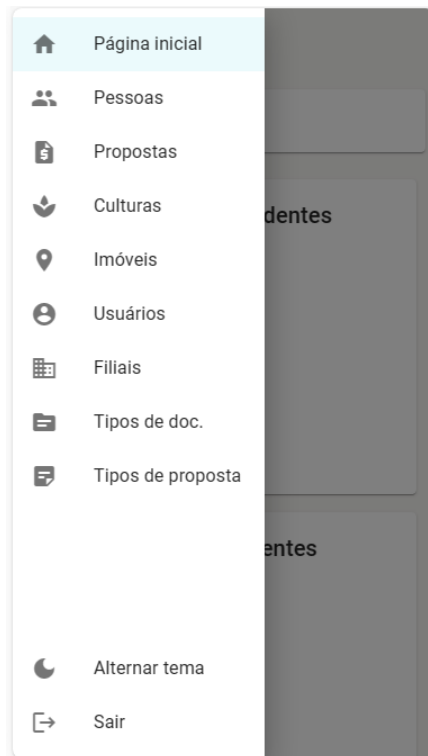
Todas as telas possuem responsividade, conforme definido no RNF 07, assim como pode ser visto na Figura 45, o menu fica escondido em telas menores, exibindo um botão que abre o menu somente nos momentos que o usuário precisar, conforme a Figura 46.

Figura 45 – Menu Lateral Fechado



Fonte: acervo do autor (2023).

Figura 46 – Menu Lateral Aberto



Fonte: acervo do autor (2023).

Outra questão necessária na responsividade são os formulários, que precisam adequar a organização dos campos de acordo com o tamanho da tela. A Figura 47 exemplifica essa responsividade com o formulário de Imóvel, que reorganiza o tamanho dos campos e sua localização para facilitar a utilização em equipamentos com um tamanho de tela menor.

Figura 47 – Cadastro de Imóvel Responsivo

Imóvel Teste

SALV... APAG... VOLT...

Geral

Nome
Imóvel Teste

Área total
ha 20

Área agrícola
ha 0

Área de pastagem
ha 0

Arquivo KML

Proprietário
Pierre Capistrano Lopes

Cidade
Leoberto...

Matrícula
123

Latitude
123

Longitude
123

Roteiro de acesso
TEste

Observação
teste

DOCUMENTAÇÕES

Fonte: acervo do autor (2023).

Tal responsividade também ocorre nas listagens, conforme Figura 48. Porém neste caso apenas é necessário ajustar o tamanho das colunas.

Figura 48 – Listagem de Imóvel Responsivo

Listagem de imóveis

Pesquisar... NOVO +

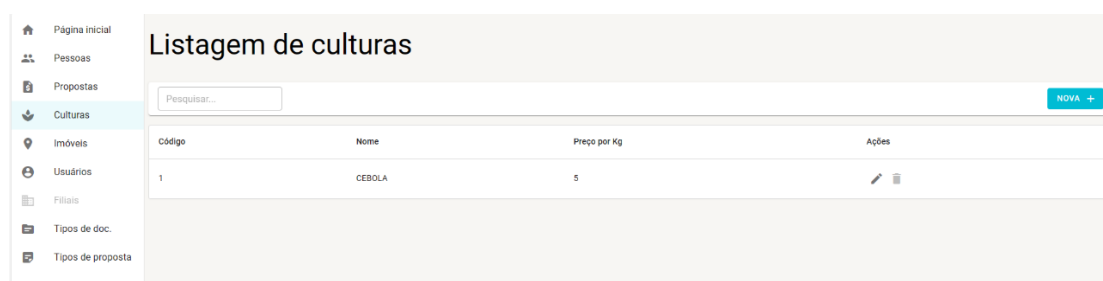
Código	Nome	Matrícula	Área total (ha)	Ações
1	Imóvel Teste	123	20	 

Fonte: acervo do autor (2023).

O RNF 08 define que os componentes visuais no *front-end* devem tratar as permissões do usuário, portanto como foi visto anteriormente, existe o cadastro de permissões do usuário, onde podem ser definidas as opções para os processos de Pessoa, Cultura, Usuário, Imóvel Filial, Documentação e Proposta.

Conforme pode ser visto na Figura 49, caso um usuário não tenha permissão para visualizar um determinado processo, o ícone no menu ficará desabilitado, como está sendo exemplificado para o processo de Filial. Além disso, também pode ser visto que caso o usuário não tenha permissão para excluir, o botão de exclusão tanto da listagem quanto do detalhe dos registros não é habilitado, como está exemplificado no processo de Cultura.

Figura 49 – Tratamento de Permissão de Usuário



Fonte: acervo do autor (2023).

Além disso, ao acessar um registro, caso o usuário não tenha permissão de editar, o formulário ficará com os campos desabilitados, permitindo apenas a visualização, como está exemplificado na Figura 50 para o processo de Imóvel.

Figura 50 – Tratamento de Permissão de Usuário no Formulário

Imóvel Teste

[← VOLTAR](#)

Geral

Nome: Imóvel Teste

Área total: ha 20

Área agrícola: ha 0

Área de pastagem: ha 0

Arquivo KML:

Proprietário: Pierre Capistrano Lopes

Cidade: Leoberto Leal

Matrícula: 123

Latitude: 123

Longitude: 123

Roteiro de acesso: TEste

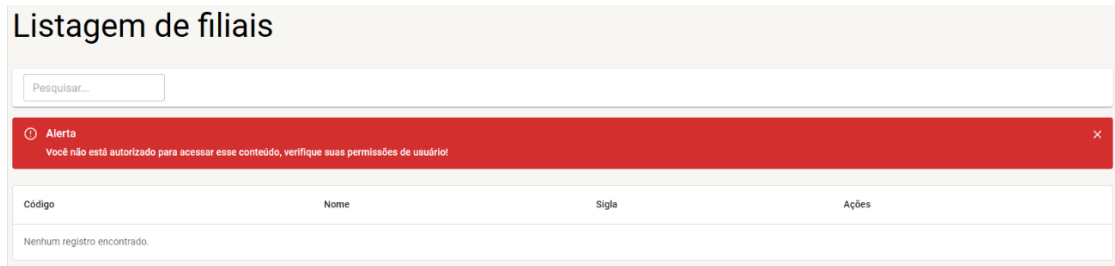
Observação: teste

[DOCUMENTAÇÕES](#)

Fonte: acervo do autor (2023).

Todas essas validações no *front-end* também são realizadas pelo *back-end*, pois são exibidas validações caso o usuário tente fazer um processo do qual ele não está autorizado. Como está exemplificado na Figura 51, onde o usuário não está autorizado para visualizar as Filiais, e caso tente acessar a rota manualmente, é exibido uma validação.

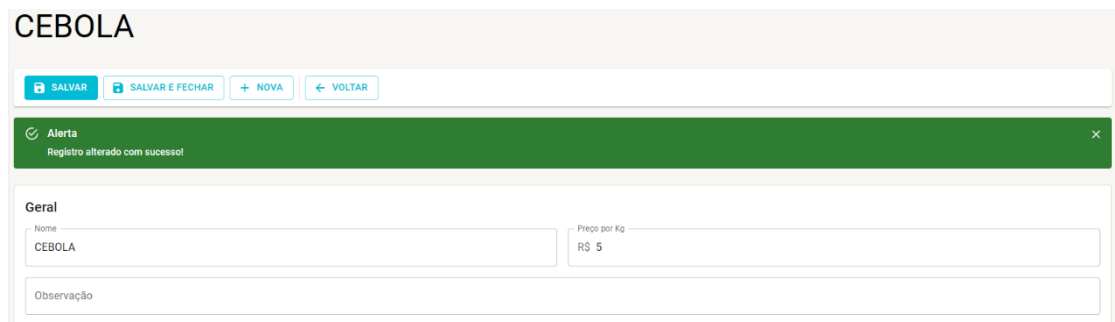
Figura 51 – Mensagem de Validação de Permissão de Usuário



Fonte: acervo do autor (2023).

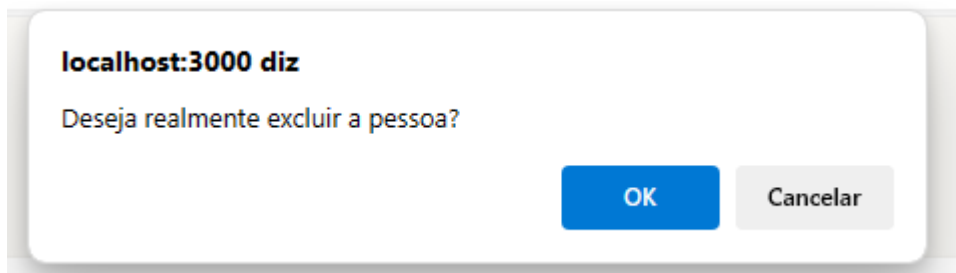
No que se refere as mensagens do sistema, elas são realizadas de duas formas, uma é pelo *Alert* do Material UI como pode ser visto na Figura 52, e a outra é pelo *Alert* padrão do JS, como pode ser visto na Figura 53.

Figura 52 – Mensagem de Sucesso Alert do Material UI



Fonte: acervo do autor (2023).

Figura 53 – Mensagem de Confirmação Alert Padrão



Fonte: acervo do autor (2023).

O *Alert* do Material UI é utilizado para exibir as mensagens em todas as telas, aparecendo abaixo da barra de ferramentas, enquanto o *Alert* padrão do JavaScript é utilizado para confirmar algumas ações com o usuário, como a exclusão de um registro.

5. CONSIDERAÇÕES FINAIS

Este trabalho de conclusão de curso se trata do desenvolvimento de um protótipo de uma aplicação web responsiva para o gerenciamento de propostas em escritórios de soluções agrícolas, auxiliando na organização das informações dos trabalhos desenvolvidos no dia a dia do escritório.

As ferramentas e linguagens de programação utilizadas auxiliaram no desenvolvimento do protótipo, assim permitindo que o mesmo fosse concluído. Na parte do *back-end* o ASP.NET auxiliou com as funcionalidades que traz para o desenvolvimento com C#, já no *front-end* o ReactJS juntamente com a biblioteca Material UI foram fundamentais na criação de componentes de forma padronizada, veloz e responsiva. Quanto ao banco de dados, o SQL Server foi adequado visando manter a segurança e integridade dos dados da aplicação, juntamente com o Entity Framework que realizou o mapeamento objeto-relacional.

Em relação aos objetivos específicos do trabalho, partindo do que visava detalhar sobre as tecnologias utilizadas para o desenvolvimento do protótipo, este foi atingido através da construção da revisão da literatura.

Em segundo lugar em relação ao objetivo de explicar o processo atual realizado por um escritório de soluções agrícolas, este foi alcançado por meio do mapeamento de cenário junto ao escritório Lopes Projetos Agrícolas.

O terceiro objetivo propunha descrever os requisitos e elaborar diagramas, este foi cumprido por meio da seção de análise, onde se detalhou os requisitos funcionais, requisitos não funcionais e regras de negócio da aplicação, bem como, onde foram apresentados os diagramas de caso de uso, entidade-relacionamento e diagrama de classes.

Por fim, o objetivo que propunha aplicar as tecnologias selecionadas para o desenvolvimento do protótipo também foi cumprido, sendo demonstrado por meio do quarto capítulo, onde foram apresentadas as técnicas e ferramentas utilizadas, sendo também demonstrado o funcionamento do protótipo com a explicação de todas as telas e rotinas da aplicação.

O protótipo desenvolvido oferece ao usuário uma ferramenta de auxílio na organização das informações e gestão das rotinas internas do escritório disponibilizando um controle eficiente de dados e documentos. O protótipo permite o cadastro de pessoas, imóveis, propostas, laudos de acompanhamento e documentações, facilitando o acesso a essas informações em um

ambiente integrado. Além de permitir o controle de acesso por meio das permissões de usuários, garantindo que somente usuários autorizados possam realizar os processos.

Assim, pode-se concluir que os objetivos específicos estabelecidos para o trabalho foram alcançados, bem como seu objetivo geral, porém, considerando as limitações do trabalho, foram listadas algumas recomendações de trabalhos futuros, como melhorias e novas funcionalidades para o protótipo.

5.1 RECOMENDAÇÕES DE TRABALHOS FUTUROS

Para o aprimoramento do protótipo foram identificados algumas melhorias e novas funcionalidades, que foram tratadas como requisitos opcionais e listados no Quadro 5. A primeira recomendação seria criar um controle de trilha de auditoria nos processos, descrevendo qual usuário realizou o processo e a data, tudo isso em uma tela padronizada para visualização dos registros.

Como segunda recomendação de desenvolvimento futuro seria adicionar um processo financeiro de contas a pagar e contas a receber, integrando com o processo de proposta, gerando a cobrança da assistência técnica automaticamente. Juntamente com a cobrança dos serviços técnicos, uma recomendação seria a integração para emissão de nota fiscal de serviço eletrônica com os *webservices* de terceiros, emitindo automaticamente o documento fiscal.

Outra recomendação seria o cadastro de croquis de localização, onde será cadastrado a imagem da área do croqui e glebas geográficas, bem como o vínculo ao imóvel referenciado.

Uma evolução para o protótipo seria o cadastro de laudos de Proagro, onde será descrita toda a visita técnica feita pelo especialista.

Outra sugestão seria uma área do cliente, onde o cliente (contratante) poderá verificar a situação atual da sua proposta sem precisar contatar o escritório diretamente.

A proposta também poderá receber melhorias, como o cadastro da capacidade de pagamento do proponente, utilizando o modelo atual descrito no mapeamento de cenário como referência. Além de também permitir a visualização das documentações vinculadas na impressão da proposta.

No contexto geral do sistema, poderá ser melhorado os alertas e validações, garantindo que as mensagens sejam exibidas em conjunto ao realizar uma ação, permitindo que o usuário efetue os ajustes necessários antes de finalizar o processo, sem exibir diversas mensagens separadamente.

No contexto da homologação da aplicação, seriam indicados testes com o gestor da empresa e alguns usuários (clientes), assim auxiliando a validação das regras de negócio implementadas e a usabilidade do sistema. Após estar homologado, o sistema poderia ser implantado.

REFERÊNCIAS

- ALVES, William P. **Banco de Dados**. São Paulo Editora Saraiva, 2014a. E-book. ISBN 9788536518961. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788536518961/>. Acesso em: 24 mar. 2023.
- ALVES, William P. **Desenvolvimento e design de sites**. São Paulo: Editora Saraiva, 2014b. Ebook.
- AXIOS. **Introdução**. 2023. Disponível em: <https://axios-http.com/ptbr/docs/intro>. Acesso em: 30 de out. de 2023.
- BARNES, D. J.; KÖLLING, M. **Programação Orientada a Objetos com Java**. 3. ed. São Paulo: Pearson, 2012.
- CODD, Edgar Frank. (1985). **Is Your DBMS Really Relational?** Computerworld, 19(30), 29-33. Disponível em: <https://www.analytictech.com/borgatti/whatEver/readingsem/Codd.pdf>. Acesso em: 24 mar. 2023.
- DEV MEDIA. **Gestão de Regras de Negócios**. 2014. Disponível em: <https://www.devmedia.com.br/gestao-de-regras-de-negocios/30670>. Acesso em: 02 out. 2023.
- FARINELLI, Fernanda. **Conceitos básicos de programação orientada a objetos**. Instituto Federal Sudeste de Minas Gerais, 2007.
- FLANAGAN; DAVID. **Javascript**. Porto Alegre: Grupo A, 2014. Ebook.
- FRAMEWORK, Entity. **Entity framework**. Architecture [edit], v. 21, p. 22, 2021.
- FREITAS, Pedro Henrique C. et al. **PROGRAMAÇÃO back end 3**. Porto Alegre SAGAH 2021 1 recurso online ISBN 9786581492274.
- IVANOV, Maksim; BESPOYASOV, Alex. **Fullstack React with TypeScript: learn pro patterns for hooks, testing, redux, SSR, and GraphQL**. [s.l.], Newline, 2020. Ebook.
- JONES, M., Bradley, J., Sakimura, N. (2015) “**JSON Web Token (JWT) – RFC 7519**”, Internet Engineering Task Force, IETF. Disponível em: <https://tools.ietf.org/pdf/rfc7519.pdf>. Acesso em: 23 ago. 2023.
- JWT.IO. **Introduction to JSON Web Tokens**. 2023. Disponível em: <https://jwt.io/introduction>. Acesso em: 23 ago. 2023.
- MARTIN, Robert C. **Código limpo: habilidades práticas do Agile software**. Rio de Janeiro Alta Books, 2009.
- MATERIAL UI. **Material UI – Overview**. 2023. Disponível em: <https://mui.com/material-ui/getting-started/>. Acesso em: 02 set. 2023.

MDN. **Glossário**. 2022a. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Glossary>. Acesso em: 29 jun. 2023.

MDN. **Métodos de requisição HTTP**. 2023. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods>. Acesso em: 29 jun. 2023.

MDN. **Javascript**. 2022b. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 12 mai. 2023.

MICROSOFT. **Introdução ao Identity no ASP.NET Core**. 2023a. Disponível em: <https://learn.microsoft.com/pt-br/aspnet/core/security/authentication/identity>. Acesso em: 29 jun. 2023.

MICROSOFT. **O que é .NET Framework?**. 2023b. Disponível em: <https://dotnet.microsoft.com/pt-br/learn/dotnet/what-is-dotnet-framework>. Acesso em: 29 jun. 2023.

MICROSOFT. **Um tour pela linguagem C#**. 2023c. Disponível em: <https://learn.microsoft.com/pt-br/dotnet/csharp/tour-of-csharp/>. Acesso em: 15 out. 2023.

MICROSOFT. **What is SQL Server?**. 2023d. Disponível em: <https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16>. Acesso em: 15 out. 2023.

MORAIS, Izabelly Soares de; ZANIN, Aline. **Engenharia de Software**. Porto Alegre: Grupo A, 2020.

OLIVEIRA, Cláudio Luís Vieira. **JavaScript descomplicado: programação para a Web, IoT e dispositivos móveis**. São Paulo Erica 2020 1 recurso online ISBN 9788536533100.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software**. Porto Alegre: AMGH Editora Ltda, 2021. Ebook.

PRITCHETT, D. (2008). **BASE: An Acid Alternative**. ACM Queue, 6(3), 48-55. Disponível em: <https://dl.acm.org/doi/pdf/10.1145/1394127.1394128>. Acesso em: 24 mar. 2023.

REACT. **Uma biblioteca Javascript para criar interfaces de usuário**. 2023. Disponível em: <https://pt-br.react.dev/learn>. Acesso em 22 junho. 2023.

RED HAT. **API REST**. 2020. Disponível em: <https://www.redhat.com/pt-br/topics/api/whatis-a-rest-api>. Acesso em: 29 jun. 2023.

RODRIGUES, Thiago N. et al. **Integração de aplicações**. Porto Alegre: Grupo A, 2020. Ebook.

ROLDÁN, Carlos Santana. **React 17: design patterns and best practices**. 3. ed. Birmingham: Packt, 2021. Ebook.

SADAMOTO, A. M. A. et al. **NoSQL databases: an overview**. In: Proceedings of the 2012 Brazilian Symposium on Computing Systems Engineering (SBESC). IEEE, 2012.

SARAIVA, Maurício de O.; BARRETO, Jeanine dos S. **Desenvolvimento de sistemas com PHP**. Porto Alegre: Grupo A, 2018. Ebook.

SILVA, Luiz F. Calaça et al. **Banco de dados não relacional**. Porto Alegre SAGAH 2021 1 recurso online ISBN 9786556901534.

SOMMERVILE, Iam. **Engenharia de software**. São Paulo: Pearson Prentice Hall, 2011. Ebook.

SWAGGER. **About Swagger**. 2023. Disponível em: <https://swagger.io/about/>. Acesso em: 23 ago. 2023.

TAMAE, André. **Desenvolvimento de Aplicações Web com JSP, Servlets, JavaServer Faces, Hibernate, Ajax e Web Services**. São Paulo: Érica, 2004.

TYPESCRIPT. **TypeScript is JavaScript with syntax for types: what is TypeScript**. 2023. Disponível em: <https://www.typescriptlang.org>. Acesso em: 29 de out. de 2023.

ANEXOS**ANEXO I – TERMO DE AUTORIZAÇÃO PARA USO DE NOME EMPRESARIAL E DADOS****CENTRO UNIVERSITÁRIO PARA DESENVOLVIMENTO DO ALTO VALE DO ITAJAÍ**
TERMO DE AUTORIZAÇÃO PARA USO DE NOME EMPRESARIAL E DADOS

Ilmo. Sr (a): Alan Fernando Lopes.
Rio do Sul, 27 de agosto de 2023.

Eu, Pierre Capistrano Lopes matriculado no curso de Sistemas de Informação, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí com a orientação do professor M.e Jullian Hermann Creutzberg, venho solicitar a V. Sa. A autorização para uso do nome da empresa “Lopes Projetos Agrícolas” e eventual coleta de dados com a finalidade de levantamento de requisitos e desenvolvimento de Trabalho de Conclusão de Curso do no curso de Sistemas de Informação, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí com o título “PROTÓTIPO DE SISTEMA WEB PARA GERENCIAMENTO DE PROPOSTAS EM ESCRITÓRIOS DE SOLUÇÕES AGRÍCOLAS”. O nome da empresa e algumas planilhas modelo serão citados no decorrer do mesmo, como potencial público-alvo da implantação do protótipo. Assumo o compromisso de utilizar os dados obtidos somente para fins acadêmicos e científicos, bem como de compartilhar os resultados obtidos ao final da pesquisa. Agradecemos antecipadamente e esperamos contar com a sua colaboração.

Atenciosamente,
Pierre Capistrano Lopes

Lopes Projetos Agrícolas Ltda
Alan Fernando Lopes
CETA- 06903608940
Alan Fernando Lopes.
Responsável.
Lopes Projetos Agrícolas LTDA.