SISTEMA DE GERENCIAMENTO PARA O LABORATÓRIO DE ANÁLISE DE SOLO, ÁGUA E PLANTA

Paulo Victor Benevides Marinho[1], *Lenardo Chaves e Silva*[2]

Resumo: A correta gestão dos dados em um ambiente de trabalho é imprescindível para a execução das tarefas de maneira produtiva. O Laboratório de Análise de Solo, Água e Planta é um ambiente de estudo e pesquisa que gerencia seus dados através de planilhas eletrônicas. Cada amostra deixada para inspeção é catalogada com uma variada gama de medições que descrevem com precisão o material analisado. Consequentemente, à medida que novos valores são inseridos o processo de investigação do material se torna cada vez mais custoso e propenso a erros. Este artigo apresenta uma solução para administrar as informações coletadas pelo laboratório, utilizando, para isso, um sistema *web* capaz de armazenar as amostras de solo, água e planta analisadas pelo local, além de gerenciar os dados de produtores responsáveis por estes exemplares. O *software* foi desenvolvido utilizando como base o ambiente de execução *Node.js* e a biblioteca especializada em criações de interface, o *React*. Como resultado, o produto final cumpre bem os requisitos e ainda oferece outros recursos como geração de boletins e autenticação de usuários, buscando desempenhar um papel importante para o laboratório e entregando mais segurança no gerenciamento das informações.

Palavras-chave: Gerenciamento de Amostras; Sistema Web; Análises Laboratoriais; Engenharia de Software.

1. INTRODUÇÃO

O Laboratório de Análise de Solo, Água e Planta (LASAP) está localizado no campus sede (Mossoró-RN) da Universidade Federal Rural do Semi-Árido (UFERSA). Seu espaço conta com diversos laboratórios e salas de aula, a fim de promover a integração entre as esferas da pesquisa, ensino e extensão do Departamento de Ciências Agronômicas e Florestais (DCAF). Nos laboratórios são feitas pesquisas e análises sobre as amostras de solo, água e planta entregues no local. Os procedimentos realizados envolvem estudos como: fertilidade do solo, nutrição de plantas, análise física do solo e análise da qualidade da água para irrigação. Muitas dessas informações são armazenadas em planilhas eletrônicas desenvolvidas através do editor de planilhas *Microsoft Excel*.

A utilização desse *software* promove a virtualização das informações obtidas nos laboratórios, oferecendo uma maior segurança no armazenamento dos dados coletados. Paralelamente, o *Excel* permite lidar com fórmulas matemáticas de maneira simplificada, automatizando todo o processo de cálculo através de instruções pré-programadas. Entretanto, à medida que a quantidade de dados se torna muito densa, extrair informações da planilha vai se tornando um processo cada vez mais complexo. Além disso, a ferramenta é bastante suscetível a erros por conta da proximidade com que estão dispostos os valores, possibilitando, assim, a inserção incorreta de novos dados e a análise comprometida de informações. A falta do trabalho colaborativo em tempo real é outra desvantagem observada no uso do *Excel* e responsabiliza um grupo de usuários a ter atenção redobrada no compartilhamento de arquivos para evitar desencontro de atualizações. Esses são alguns dos principais motivos que demonstram a necessidade de mudança na forma de armazenar os dados do LASAP.

Nesse ponto surge a ideia da implementação de um sistema *web* alinhado às necessidades do laboratório. Os sistemas *web* são ferramentas hospedadas em um servidor na *Internet*. Segundo Lie e Bos (2005), um servidor pode ser definido como um computador ou *software* responsável por fornecer respostas mediante solicitações de clientes. Na maioria das vezes, essas respostas do servidor são exibidas no formato de páginas *web*, sendo estas geralmente acessadas através dos navegadores. Os sistemas *web* podem ser utilizados para simplificar serviços da vida real, oferecendo as mais variadas funcionalidades que uma aplicação possa desempenhar.

Sobre a utilização desse tipo de sistema, Conallen (2003) aponta como principais vantagens de um sistema *web*: a possibilidade de ser executado em qualquer navegador de *Internet*; atualização dos dados simultânea para todos os usuários; redução de custos, sendo possível realizar serviços remotamente; escalabilidade, ou seja, a capacidade que o sistema tem em lidar com grandes volumes de dados; não exigir memória da máquina, uma vez que realiza todas as funções no servidor; entre outras vantagens que fazem dos sistemas *web* serem uma das principais soluções da Engenharia de *Software*.

[1] Discente do Bacharelado em Ciência da Computação da Universidade Federal Rural do Semi-Árido (UFERSA) – Campus Mossoró.
[2] Professor Doutor do Departamento de Computação da Universidade Federal Rural do Semi-Árido (UFERSA) – Campus Mossoró.

O presente trabalho teve como objetivo o desenvolvimento de um sistema *web* capaz de administrar os dados coletados nas instalações do LASAP. A aplicação reúne as amostras estudadas de solo, água e planta, além dos dados dos clientes portadores desses espécimes. Cada uma das entidades com seus atributos fundamentais tanto para a análise do material quanto para o prosseguimento do fluxo de trabalho do local. O sistema possui uma estrutura simples e organizada, permitindo acessar os registros com facilidade e proporcionando uma boa experiência de usabilidade. Dessa forma, o usuário final do sistema pode desfrutar de uma ferramenta capaz de disponibilizar todas as informações cadastradas em tempo real, como também acessá-las de qualquer dispositivo que possui conexão com a *Internet*.

Este artigo está organizado em sete seções, incluindo a atual, que faz uma breve apresentação do tema, apontando também o problema e o objetivo da pesquisa. Na Seção 2, são citados trabalhos que abordam temas semelhantes ao artigo vigente. Na Seção 3, é apresentada a metodologia seguida pelo projeto. Em seguida, na Seção 4 é abordado o desenvolvimento do sistema proposto, descrevendo os requisitos funcionais e não funcionais, protótipos elaborados para nortear a construção do sistema e as tecnologias que foram utilizadas na etapa de implementação. Na Seção 5, são apresentados os resultados alcançados, expondo as principais funcionalidades da aplicação desenvolvida. Por fim, na Seção 6 são discutidas as considerações finais e possíveis sugestões para trabalhos futuros.

2. TRABALHOS RELACIONADOS

A seguir são apresentados os trabalhos que foram encontrados na literatura e possuem propósitos semelhantes ao Sistema de Gerenciamento do LASAP. A seção aborda as principais funcionalidades que cada ferramenta possui, como também as linguagens que foram utilizadas nas implementações e os benefícios promovidos por essas aplicações. A discussão se encerra com um breve comentário acerca dos trabalhos citados, comparando as suas características e destacando a importância do vigente artigo.

Takaki e Sakuray (2023) apresentaram um sistema de controle para atividades de coleta e análise de dados. A ferramenta foi desenvolvida utilizando a linguagem de programação *Python* e contribui para uma melhor organização de um ambiente de pesquisa. Os procedimentos são realizados pelo Laboratório de Análise de Alimentos da Universidade Estadual de Londrina (UEL), que encaminha os resultados obtidos nas investigações para serem armazenados na aplicação. Seu funcionamento é baseado em etapas e tem como principais artifícios emissão de boletos, leitura de amostras e solicitação de laudos.

Novachinski e Silva (2006) publicaram a respeito da ferramenta SisLab, uma solução criada para gerenciar as informações do Laboratório de Análise de Solos, Plantas e Corretivos da Embrapa Agropecuária Oeste. A aplicação tem como principais funções o cadastro de clientes e amostras, geração de boletins e digitação de resultados. A utilização do *software* permite a classificação do solo e a análise de fertilidade, além de fornecer uma maneira segura para armazenamento das informações em um banco de dados relacional. O SisLab foi desenvolvido através do *Microsoft Access*, um programa capaz de criar interfaces gráficas para serem utilizadas em conjunto com bases de dados.

Silva e Nascimento (2023) descreveram o processo de criação de um sistema para interpretação de análise de solo, o *Soil Analysis*. O mesmo foi idealizado para aprimorar a eficiência e precisão dos resultados obtidos em lavouras de capim, buscando reduzir os custos dos produtores rurais e entregar soluções para manejo, correção e fertilidade do solo. Para implementação do sistema foram utilizados o Javascript, HTML, CSS e MySQL, resultando em uma aplicação *web* que pode ser acessada por diferentes dispositivos. As funcionalidades disponíveis na ferramenta são a inserção de propriedades, gestão de pastagens para alimentação dos animais e o cadastro de análises, onde as medições de cada elemento das análises químicas e físicas do solo são inseridas.

O Labguru[3] é um *software* de gerenciamento de laboratórios criado em 2008 por Jonathan Gross. A ferramenta conta com diversos recursos que permitem aumentar a produtividade e reduzir a ocorrência de falhas durante o fluxo de trabalho. Uma das suas principais características é a customização, uma vez que possibilita a criação de coleções personalizadas, definindo atributos, fórmulas e regras para os itens inseridos neste conjunto. O sistema ainda oferece a automação de processos, trabalho colaborativo, simulação de experimentos, gerenciamento de estoque, dentre outras funcionalidades.

Buscando por soluções correlatas foi possível evidenciar a falta de trabalhos que abordam diretamente sobre o gerenciamento simultâneo de amostras de solo, água e planta. Embora sistemas como o *Soil Analysis* e o SisLab apresentem domínios parecidos com a temática deste artigo, não é possível atender plenamente as exigências básicas do LASAP. Os principais impedimentos são a utilização do programa *Access* por parte do SisLab, que não permite o trabalho colaborativo em tempo real e o acesso dos usuários por diferentes dispositivos, e as amostras serem específicas de um tipo no *Soil Analysis*. O trabalho de Takaki e Sakuray (2023), por sua vez, tratou de análises de outro tipo de material, impossibilitando a adoção da ferramenta para o problema. Por fim, o Labguru trouxe uma variada gama de utilidades, porém, com uma certa complexidade no processo de customização total do aplicativo e ainda sem suporte para a língua portuguesa.

[3] Disponível em: https://www.labguru.com/

3. METODOLOGIA

Na Engenharia de *Software* existem dois principais tipos de metodologias de desenvolvimento: a metodologia tradicional e os métodos ágeis (LEAU, Y.B. et al., 2012). A primeira delas baseia-se na utilização de uma sequência linear de etapas para se chegar ao resultado final do projeto. Algumas das fases comumente observadas neste tipo de abordagem são a especificação de requisitos, implementação e realização de testes. Essa metodologia adota uma perspectiva mais rígida com relação aos seus processos, de tal forma que uma tarefa não possa ser iniciada até que a anterior seja concluída. Possui uma documentação bem estruturada que deve ser seguida de maneira rigorosa, não oferecendo flexibilidade para mudanças. As principais metodologias tradicionais são: o modelo em cascata (*Waterfall*), *V-Model*, RUP (*Rational Unified Process*) e Modelo Espiral.

Paralelamente, têm-se a metodologia ágil, um conceito que se originou com a difusão do Manifesto Ágil (BECK, K. et al., 2001). Essa iniciativa tinha como objetivo propagar ideais acerca do desenvolvimento de *software*, buscando atingir melhores resultados em um curto período de tempo. Tal concepção seria efetivada através da flexibilidade de mudanças e da comunicação direta com o cliente, a fim de diminuir a rigidez de processos observada no modelo tradicional. De acordo com Amaral et al. (2011), o gerenciamento ágil é uma abordagem que busca simplificar a organização de projetos, utilizando, para isso, um conjunto de princípios cujo objetivo é melhorar o desempenho dos resultados alcançados pela equipe em cada etapa, limitando o esforço na gestão e entregando mais inovação e valor ao cliente. Alguns dos principais métodos atualmente utilizados no mercado são o *Scrum*, o *Kanban*, o XP (*extreme programming*) e o *Lean Startup*.

A metodologia adotada para o Sistema de Gerenciamento do LASAP mistura elementos tanto do modelo tradicional quanto do desenvolvimento ágil, adotando os conceitos do *Waterfall* e do *Kanban*. A ideia da utilização de ambos os métodos seria de organizar o projeto em etapas, porém, com uma maior produtividade na execução de tarefas. Inicialmente, foram realizadas reuniões com as partes interessadas na criação do sistema, permitindo obter informações fundamentais para o processo de desenvolvimento da aplicação. Em seguida, realizou-se a etapa da especificação de requisitos, apontando os principais recursos e funcionalidades que deveriam estar presentes no sistema. Por conta da utilização simultânea da metodologia tradicional com os métodos ágeis, os requisitos foram ajustados à medida que o *software* era desenvolvido, possibilitando, assim, uma maior flexibilização dos processos. Na terceira etapa, a realização da prototipagem do sistema, que serviria de suporte para a construção das telas da aplicação. Na quarta etapa, iniciou-se a fase de implementação, onde as tarefas eram realizadas em ciclos com duração média de duas semanas, promovendo agilidade ao cronograma. Para esta etapa, foi utilizada uma das principais características do *Kanban*, o quadro de acompanhamento de atividades (ver Figura 1). Para explorar esse tipo de abordagem fez-se necessária a introdução do *Trello*[4], uma ferramenta especializada na gestão de projetos. A adoção do método também permitiu a execução da quinta e última etapa, a fase de testes, procedimento essencial para a visualização da corretude do sistema.

Figura 1. Quadro Kanban do Projeto. (Autoria Própria).

O *Trello* é um *software* capaz de organizar diferentes tipos de ambientes de trabalho. Sua ideia básica é entregar uma visualização geral do andamento das tarefas de um projeto. Com o *Trello*, é possível gerenciar membros de equipes, delegando funções e controlando possíveis pendências. O esqueleto da aplicação é formado por listas e cartões, que descrevem em detalhes as tarefas apontando seus prazos e objetivos. Além disso, a ferramenta permite a anexação de arquivos, integrações com outros aplicativos, automação de processos, entre outros recursos.

Aplicando ao projeto o modelo de quadro *Kanban* disponibilizado pelo *Trello*, foi possível gerenciar o fluxo de desenvolvimento do sistema. A estrutura do painel é formada por sete colunas que determinam o estágio em que cada tarefa se encontra. Todas as colunas possuem um rótulo e são identificadas da seguinte maneira: "*Backlog"*, onde fica a lista de tarefas a serem implementadas no projeto; "*Design"*, momento em que se inicia a prototipação da funcionalidade requerida; "A fazer", tarefas que foram selecionadas para serem concluídas no período estipulado entre uma reunião e outra; "Em andamento", funcionalidades que estão sendo desenvolvidas atualmente; "Revisão de código", etapa em que a lógica de programação é reavaliada a fim de obter melhores resultados; "Fase de teste", período no qual as funções são executadas sob a possibilidade de encontrar algum erro de implementação; e "Concluído", tarefas que foram aprovadas após percorrerem todas as etapas do processo.

[4] Disponível em: https://trello.com/

4. DESENVOLVIMENTO

Nesta seção são apresentados os artefatos fundamentais para a construção do sistema. Iniciando pela especificação dos requisitos, que representam as principais funcionalidades as quais o produto deveria possuir. Em seguida, uma explanação sobre a fase de prototipagem, onde foram modeladas as telas da aplicação dando uma visão geral de *design* e comportamento. E por fim, a apresentação das tecnologias utilizadas para implementação, relatando as linguagens, *frameworks* e demais recursos escolhidos.

4.1. Requisitos

O levantamento dos requisitos é uma tarefa de fundamental importância para o desenvolvimento de um *software*. São eles os responsáveis por guiarem os desenvolvedores para uma correta idealização do produto. Sommerville (2011) definiu um requisito de sistema como a descrição de uma função, restrição e/ou serviço que a aplicação deve atender. Para ele, os requisitos apresentam as reais necessidades do cliente, especificando todo o comportamento externo do sistema. Também é preciso descrever o requisito em diferentes níveis de detalhamento, com o objetivo de extrair o máximo de informações possíveis sobre uma determinada ação do sistema, porém, nada que envolva sua parte arquitetural. Além disso, os requisitos são divididos em funcionais e não funcionais, que serão abordados nos próximos tópicos desta subseção. A seguir são elencados os requisitos do sistema conforme a sua funcionalidade, apresentando uma breve definição sobre as particularidades de cada um deles.

4.1.1. Requisitos Funcionais

Os requisitos funcionais (RF) de um sistema são aqueles que descrevem exatamente o que a aplicação deve fazer, possibilitando especificar as entradas e saídas para uma determinada ação. A especificação deve ser completa e consistente, apontando todos os serviços requeridos pelo usuário e fazendo esse processo de maneira a não gerar incongruências no texto (Sommerville, 2011). Na Tabela 1, é possível identificar a natureza funcional dos requisitos, uma vez que exploram diretamente o que deve ser feito no sistema. Em seguida, é feito um detalhamento acerca de cada exigência e seus respectivos atributos utilizados como entrada e saída de dados.

Tabela 1. Requisitos Funcionais do Sistema. (Autoria Própria).

Identificador	Descrição
RF1	O sistema deve permitir o cadastro e a autenticação de usuários.
RF2	O sistema deve permitir a redefinição de senha do usuário.
RF3	O sistema deve permitir o gerenciamento de clientes.
RF4	O sistema deve permitir o gerenciamento de amostras de solo.
RF5	O sistema deve permitir o gerenciamento de amostras de água.
RF6	O sistema deve permitir o gerenciamento de amostras de plantas.
RF7	O sistema deve possibilitar a análise das amostras.
RF8	O sistema deve gerar boletins no formato PDF.
RF9	O sistema deve permitir o gerenciamento de dados do usuário.

RF1: Funcionalidade necessária para o usuário obter acesso ao sistema. Um usuário deve possuir: e-mail, senha e nome de usuário.

RF2: Requisito essencial para que o usuário tenha uma maior segurança da sua conta. O usuário deve informar seu e-mail e receber uma mensagem com um código de segurança. Após informar o código ao sistema, o usuário poderá prosseguir com o cadastro de uma nova senha.

RF3: Cada usuário poderá cadastrar, visualizar, buscar, editar e excluir clientes do laboratório. Um cliente deve possuir: um número de protocolo, nome, data de entrada no sistema, um documento de identificação

(CPF/CNPJ), telefone, e-mail, endereço, cidade, bairro, UF e CEP.

RF4, RF5 e RF6: Cada usuário poderá cadastrar, visualizar, buscar, editar e excluir uma amostra, seja ela de solo, água ou planta. Para cadastrar uma amostra é necessário: uma identificação ou descrição de suas características, documento do cliente responsável, finalidade, data de entrada e data de saída da amostra no laboratório. Também é possível inserir uma forma de pagamento, o valor e a situação atual em que este se encontra.

RF4: Por ser uma amostra do tipo solo, o usuário ainda poderá informar: taxa de areia grossa, taxa de areia fina, taxa de silte, taxa de argila, umidade, densidade aparente, densidade real, porosidade, a porcentagem de água disponível, profundidade, classe textural, taxa de pH, condutividade, quantidade de matéria orgânica e acidez do solo, além das concentrações de potássio, sódio, magnésio, cálcio, fósforo, alumínio, hidrogênio e nitrogênio.

RF5: Por ser uma amostra do tipo água, o usuário ainda poderá informar: taxa de pH, condutividade, leitura do carbonato e leitura do bicarbonato, além das concentrações de potássio, sódio, cálcio, magnésio, cloro, nitrogênio e fósforo.

RF6: Por ser uma amostra do tipo planta, o usuário ainda poderá informar: a cultura da amostra, as concentrações de nitrogênio, fósforo, enxofre, boro, molibdênio, cloro, chumbo, cádmio, níquel, cromo, cobre, manganês, ferro e zinco, além das concentrações de potássio, sódio, magnésio e cálcio.

RF7: Com os dados das amostras reunidos no sistema, o mesmo deve calcular: a soma de bases trocáveis (SB), a capacidade de troca de cátions (CTC efetiva), capacidade de troca de cátions a pH 7,0 (t), a porcentagem de saturação por bases (V), a porcentagem de saturação por alumínio (m) e a porcentagem de sódio trocável (PST). Além do cálculo da relação de adsorção de sódio (RAS), a dureza da amostra e a quantidade de cátions e ânions presentes nela. Os valores devem ser reunidos e apresentados em um quadro informativo.

RF8: Documentos devem ser gerados para que os dados das amostras possam ser compartilhados com clientes do laboratório.

RF9: Funcionalidade necessária para que o usuário possa alterar seu nome, e-mail ou senha assim que desejar.

4.1.2. Requisitos Não Funcionais

Os requisitos não funcionais (RNF), por sua vez, são aqueles que afetam o sistema de maneira geral e não apenas seus componentes, como uma tabela ou um formulário. Como o próprio nome sugere, são comportamentos que não estão relacionados diretamente às funções do sistema. Nos requisitos não funcionais as propriedades observadas dizem respeito a desempenho, escalabilidade, segurança, usabilidade, entre outros aspectos. Muitas vezes descrevem restrições que são impostas para atender aos dispositivos de *hardware* e *software* de seus clientes finais. Para Sommerville (2011), os requisitos não funcionais são em sua maioria mais críticos que os requisitos funcionais, visto que os usuários podem contornar a ausência de funcionalidades. Entretanto, violar um requisito não funcional pode significar a inutilização do sistema. Por exemplo, desenvolver uma ferramenta para uma versão recente de um sistema operacional, quando todos os interessados na ferramenta possuem apenas versões mais antigas do *software*. Dessa forma, a ferramenta desenvolvida não poderá ser utilizada por nenhum dos seus potenciais usuários devido as limitações impostas naquela situação. Na Tabela 2, são especificados apenas requisitos baseados na experiência do usuário, pelo fato do sistema desenvolvido ser uma aplicação *web* e poder ser acessado por qualquer dispositivo com acesso à *Internet*.

Tabela 2. Requisitos Não Funcionais do Sistema. (Autoria Própria).

Identificador	Descrição
RNF1	O sistema deve ser intuitivo e de fácil utilização.
RNF2	O tempo de resposta de cada requisição do sistema não deve ultrapassar 10 segundos.

RNF1: A simplicidade do sistema permite que usuários de diferentes formações o utilizem facilmente.

RNF2: É necessária uma boa performance do sistema para melhorar a experiência do usuário.

4.2. Protótipos

Após realizada toda a parte de planejamento do *software*, onde o escopo do trabalho foi definido juntamente com os objetivos e métodos que seriam aplicados ao projeto; e a etapa do levantamento de requisitos, em que

foram elucidadas as necessidades do cliente que deveriam ser atendidas pela aplicação; é iniciada a fase de prototipagem. Nesse momento, as partes visual e comportamental do sistema foram definidas mediante a elaboração de modelos, que devem ilustrar o funcionamento das telas e da interface gráfica da aplicação. A partir disso, o processo de implementação se torna mais cristalino aos olhos do programador, uma vez que o modelo desenvolvido serve como material de referência na criação do *software*. Além disso, o protótipo concede ao cliente uma visão prévia do sistema, possibilitando que se tenha um *feedback* do usuário final antes mesmo de ser iniciada a confecção do produto.

No mercado do *design* gráfico, uma ferramenta que ganhou bastante destaque nos últimos anos foi o *Figma*[5]. Uma aplicação web que permite aos seus usuários a manipulação de recursos UX/UI de forma colaborativa. O *UX Design* (*user experience design*) é uma abordagem que trata da experiência do usuário ao interagir com algum produto ou serviço. Enquanto o *UI Design* (*user interface design*) se refere a todos os elementos visuais de uma aplicação. Criado em 2016 por Dylan Field e Evan Wallace, o *Figma*, segundo uma pesquisa realizada pelo *website UXTools*[6] no ano de 2023, é o principal software para prototipação e design de sistemas, ficando à frente de ferramentas como *Sketch* e *Adobe XD*. Além das características já apresentadas, o *Figma* ainda possui diversas bibliotecas e componentes reutilizáveis, permitindo uma maior agilidade na criação de telas. Devido a todas as vantagens e recursos oferecidos pelo *Figma*, esta foi a ferramenta escolhida para desenvolver o protótipo do Sistema de Gerenciamento do LASAP. Na Figura 2, é possível visualizar o modelo de algumas das principais telas do sistema proposto gerado por intermédio do *Figma*.

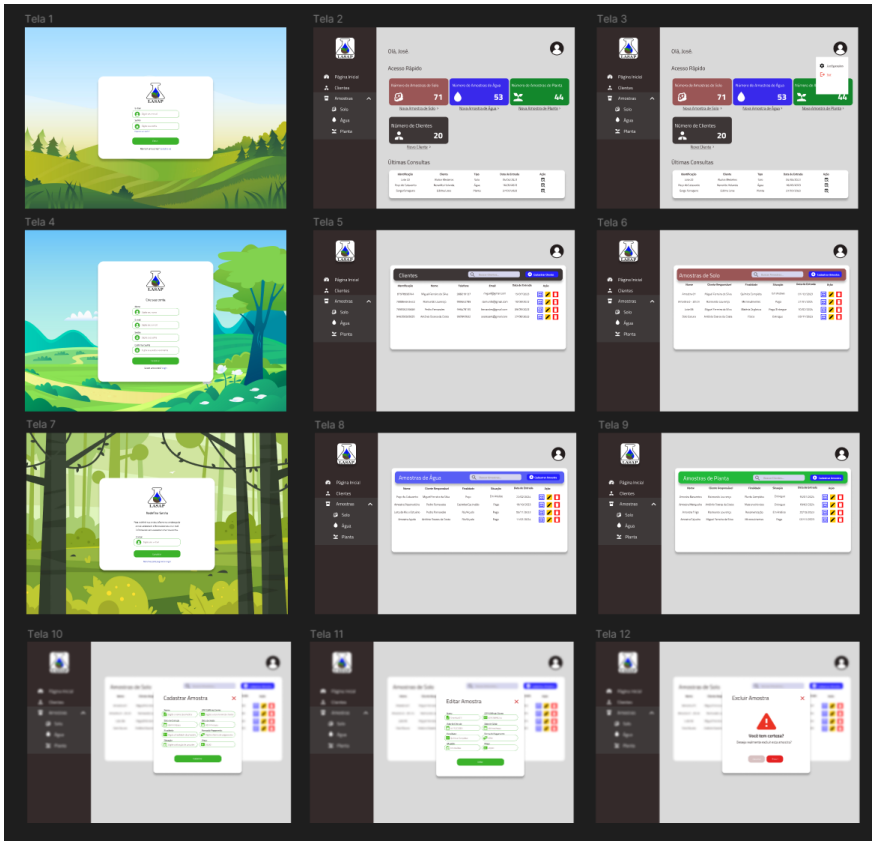

Figura 2. Protótipo *Figma* para o Sistema de Gerenciamento do LASAP (Autoria Própria).

As três primeiras telas representam *login* e página inicial, com a última sendo utilizada para exibir um menu de configurações. O conteúdo das telas 4 a 6 simbolizam o cadastro de usuários e as páginas de clientes e amostras de solo. Nas telas 7 a 9, estão dispostas a recuperação de senha e as páginas de amostras de água e amostras de planta. Por último, os formulários de cadastro e edição de amostras, bem como a representação do processo de exclusão de uma delas, que aciona um alerta de confirmação para o usuário.

4.3. Tecnologias

Com toda a parte documental em mãos, o próximo passo foi implementar o sistema por meio das linguagens de programação. O desenvolvimento *web*, área que abrange os sistemas desenvolvidos para *Internet*, possui duas grandes divisões bem estabelecidas: o *front-end* e o *back-end*, ambos representam conceitos essenciais para

[5] Disponível em: https://www.figma.com/
[6] Disponível em: https://uxtools.co/survey/2023/

produzir uma aplicação. O *front-end* é a parte visual do sistema, aquela que o usuário irá interagir diretamente ao utilizar o *software*. Quanto ao *back-end*, se refere a parte lógica do sistema, responsável pelas regras de negócio e pelo gerenciamento dos dados armazenados na aplicação. A seguir, serão apresentadas as principais tecnologias e ferramentas de desenvolvimento que permitiram a criação do sistema.

4.3.1. HTML, CSS e Javascript

São as tecnologias padrão da *web*, encarregadas de preparar toda a parte visual de uma aplicação. O HTML (*HyperText Markup Language*) é uma linguagem de marcação responsável por definir a estrutura principal de uma página. Sua utilização permite a inserção de elementos como, texto, imagem, áudio e vídeo, fazendo isso por meio das chamadas *tags*, marcadores que posicionam o conteúdo determinando início e fim da sua formatação. O CSS (*Cascading Style Sheets*), por sua vez, é uma linguagem de estilização ideal para personalizar o *layout* de uma página. Através do CSS, é possível definir características como as cores, o tamanho, a tipografia e os espaçamentos, customizando, assim, elementos HTML da aplicação. O *Javascript* é uma linguagem de programação estruturada que permite aos desenvolvedores aplicarem dinamismo ao sistema. Algumas das funcionalidades possíveis de serem reproduzidas com o *Javascript* são: a atualização em tempo real do conteúdo, validação de formulários, envio de dados registrados para o servidor e o redirecionamento do usuário entre diferentes páginas. Sua execução é feita diretamente no navegador, algo que eleva o desempenho da aplicação.

4.3.2. React

O *React*[7] é uma biblioteca *Javascript* que permite a criação de interfaces através de componentes. Cada componente representa uma parte isolada do sistema e pode ser reutilizado em diferentes locais da aplicação. Existem diferentes níveis de representação para um componente, partindo desde pequenos botões e campos de formulário até gráficos e tabelas com variadas possibilidades de seleção e filtragem de dados. O *React* adota em sua sintaxe a extensão JSX (*Javascript XML*), uma forma de combinar HTML e *Javascript* em um único documento. Apesar de oferecer uma nova maneira de implementação, o código JSX não pode ser interpretado pelo navegador e precisa ser transformado para *Javascript* por intermédio de um compilador como o *Babel* (REACT, c2024).

4.3.3. Material UI

O *Material UI*[8] é uma biblioteca *React* que possui diversos componentes pré-construídos. Por meio desse utilitário, é possível agilizar o processo de criação de uma tela do sistema, adaptando os objetos de acordo com as necessidades do desenvolvedor. O *Material UI* foi concebido como uma implementação do *Material Design*, uma linguagem idealizada pela multinacional norte-americana, *Google*, objetivando a padronização das interfaces gráficas criadas pela empresa. A personalização dos componentes pode ser realizada através de propriedades dispostas em cada um desses elementos ou ainda com auxílio da biblioteca *styled-components*, que proporciona a combinação de CSS e *Javascript* no mesmo arquivo (MATERIAL UI, c2024).

4.3.4. Node.js

O ambiente de execução *Node.js*[9] foi a principal ferramenta utilizada para criar o *back-end* do sistema. Por conta da sua capacidade de executar código Javascript no lado do servidor, o *Node.js* permite a criação de aplicações autossuficientes que não necessitam de *software* adicional para entrar em funcionamento, como é o caso dos aplicativos desenvolvidos pela linguagem de programação Java, dependentes da JVM (*Java Virtual Machine*). O *Node.js* também possui alta performance, graças à uma de suas principais características, a arquitetura orientada a eventos. Esse modelo trata os processos de maneira assíncrona, o que possibilita lidar com múltiplas requisições de maneira eficiente (NODE.JS, s.d.).

4.3.5. Express

Um *framework* do *Node.js* que fornece recursos para construção de aplicações web. O *Express*[10] oferece uma grande quantidade de métodos HTTP e *middlewares*, possibilitando a rápida criação de *APIs*. Uma Interface de Programação de Aplicações ou API (*Application Programming Interface*) é um conjunto de métodos e funções que foram programados para que outras aplicações possam utilizar os seus serviços. Para manipular uma API basta acessar o *link* ou endereço em que a mesma foi disponibilizada. Os métodos HTTP e os *middlewares*

[7] Disponível em: https://react.dev/
[8] Disponível em: https://mui.com/material-ui/
[9] Disponível em: https://nodejs.org/
[10] Disponível em: https://expressjs.com/

disponibilizados pelo *Express* são os responsáveis pela comunicação entre o *front-end* e o *back-end*. Através de requisições HTTP, o *front-end* pode solicitar a visualização, criação, edição e exclusão de dados, além de vários outros procedimentos. E como forma de atender a esses pedidos, o *back-end* envia respostas com as informações solicitadas. Esse processo é executado graças ao sistema de rotas do *Express*, que define os caminhos para a chamada das funções (EXPRESS, c2017).

4.3.6. Sequelize

O *Sequelize*[11] é um ORM (*Object-Relational Mapping*), uma técnica de desenvolvimento de *software* capaz de manipular bancos de dados relacionais por meio da programação orientada a objetos. Neste tipo de paradigma, entidades do mundo real são representadas como objetos, e cada um deles possui características e comportamentos específicos, retratados na programação como atributos e métodos, respectivamente. A utilização desse conceito permite ao *Sequelize* mapear classes para cada tabela criada no banco de dados, fazendo com que os dados sejam tratados como objetos. Além disso, ele atua como uma interface de comunicação entre o código-fonte da aplicação e o banco de dados, permitindo a abstração de comandos SQL necessários para consultas na base de dados. O *Sequelize* é uma ferramenta desenvolvida para *Node.js* e tem compatibilidade com os principais gerenciadores de banco de dados, entre eles o *MySQL*, *PostgreSQL*, *Oracle* e *SQLite* (SEQUELIZE, c2024).

4.3.7. PostgreSQL

É um sistema de gerenciamento de banco de dados (SGBD) gratuito e de código aberto que utiliza SQL, a principal linguagem utilizada neste tipo de *software*, crucial para consulta, manipulação e armazenamento de registros em bancos de dados. A estrutura dos modelos criados com o *PostgreSQL*[12] segue o padrão objeto-relacional, semelhante aos bancos relacionais, porém com suporte a orientação a objetos. O SGBD em questão ainda é altamente escalável, sendo capaz de gerenciar uma elevada quantidade de dados e suportar um alto número de usuários de maneira simultânea. Pode ser executado em sistemas operacionais como *Windows*, *Linux* e *MacOS* (POSTGRESQL, c1996-2024).

5. RESULTADOS

O Sistema de Gerenciamento do Laboratório de Análise de Solo, Água e Planta foi desenvolvido buscando atender todos os requisitos listados na Seção 4.1. deste trabalho. O *design* da aplicação teve como horizonte o protótipo apresentado na Seção 4.2. A seguir, serão explanadas as principais funcionalidades e telas do sistema.

No momento em que o usuário acessa o sistema e ainda não realizou sua autenticação, existem apenas três páginas disponíveis para visualização: tela de *login*, cadastro de usuários e tela de recuperar senha (ver Figura 3).

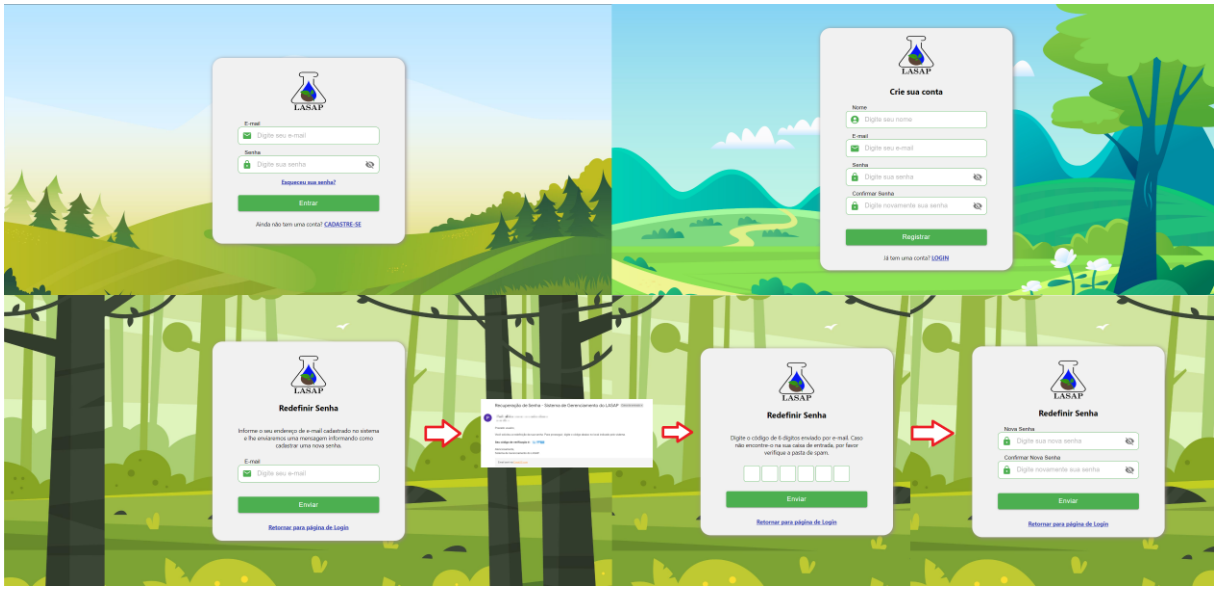

Figura 3. Telas de *Login*, Cadastro de Usuários e Recuperação de Senha. (Autoria Própria).

Iniciando com um *login* de usuários (parte superior esquerda), o *software* em questão procura entregar mais segurança ao seu cliente final. Para ter acesso aos dados armazenados no sistema, o usuário deve realizar o processo

[11] Disponível em: https://sequelize.org/
[12] Disponível em: https://www.postgresql.org/

de autenticação informando *e-mail* e senha como indicado no formulário. Caso ainda não possua uma conta registrada, é possível selecionar o botão "CADASTRE-SE" e ser redirecionado para a tela de cadastro de usuários (parte superior direita). Nela, o usuário deve seguir as orientações para inserir um nome, *e-mail* e senha válidos, além de confirmar a senha para certificar que a mesma foi informada corretamente. Após selecionar o botão "Registrar", o usuário terá sua conta cadastrada no sistema e então poderá retornar à tela de *login* e fazer o processo de autenticação.

Além disso, há também uma tela de redefinir senha (parte inferior), utilizada para garantir que o usuário continue com acesso ao sistema em caso de esquecimento da sua palavra-chave. O funcionamento dessa página se inicia com um formulário de campo único que necessita do *e-mail* do usuário para enviar novas instruções. Em seguida, o usuário receberá uma mensagem com um código de verificação a ser transcrito para um novo campo de 6 dígitos (primeira metade da parte inferior direita) do formulário. Realizando esse processo de maneira correta, será exibido ao usuário dois novos campos pedindo a nova senha e a confirmação dessa nova senha (segunda metade da parte inferior direita), respectivamente. Com isso, ao submeter essas últimas informações o usuário terá uma nova senha de acesso e poderá entrar no sistema novamente.

Após realizado o processo de autenticação, o usuário é apresentado a uma tela inicial com informações referentes as amostras (ver Figura 4). Na parte superior, um quantitativo dos registros armazenados de cada entidade do sistema. Descendo a visualização, estão reunidos os débitos de clientes para com o laboratório, exibindo somente aqueles que ainda não efetuaram o pagamento. As informações são dispostas pelo tipo da amostra e trazem dados que permitem a sua identificação, como nome, cliente responsável e finalidade.

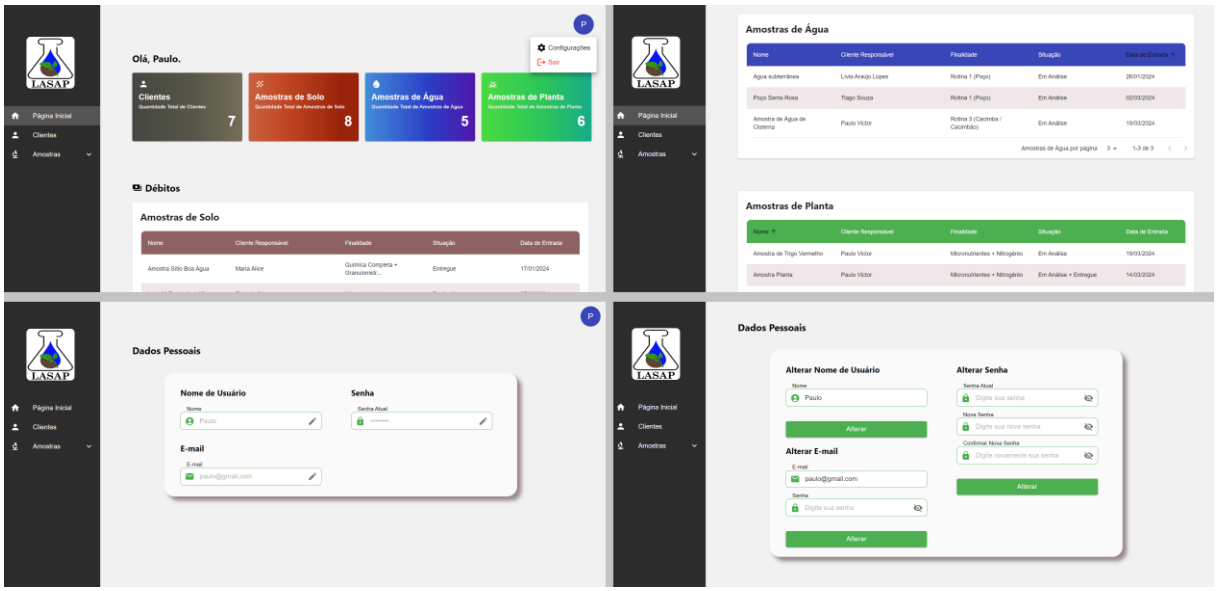

Figura 4. Telas de Página Inicial e Configurações de Usuário. (Autoria Própria).

Na tela inicial ainda há um ícone no canto superior direito que ao ser selecionado abre um menu suspenso com as opções de configuração e saída do sistema, esta última permite que o usuário encerre sua sessão assim que julgar necessário. Se tratando da página de configurações, o usuário tem o poder de alterar os seus dados cadastrais, podendo modificá-los individualmente, bastando apenas seguir as exigências de confirmação de senha para uma maior segurança deste procedimento.

Todas as telas privadas da aplicação (acessíveis somente após efetuar o *login*) contam com uma barra lateral que auxilia a navegação do usuário. A partir dela, é possível acessar as telas de clientes, amostras de solo, água e planta, além da já mencionada página inicial.

As telas de gerenciamento (ver Figura 5) apresentam estruturas semelhantes entre elas, promovendo uma padronização para o sistema. Cada uma dessas páginas é regida por uma tabela que possui as funcionalidades de cadastro, visualização, edição e exclusão dos dados, além da organização e busca de informações. Os principais atributos de cada entidade são dispostos em colunas que, por sua vez, podem ser utilizados para ordenar os dados de forma crescente ou decrescente de acordo com o parâmetro selecionado.

Outra característica importante da tabela é a paginação, uma vez que ela é capaz de dividir a quantidade total de registros em várias parcelas. A implementação desse conceito garante ao sistema mais eficiência, pois o mesmo necessitará de um tempo menor de carregamento para exibir o conteúdo da página. O usuário pode realizar o controle da paginação através de um campo que permite definir a quantidade de registros listados em cada partição. Assim como mencionado anteriormente, há também a função de busca, possibilitando o encontro de clientes pelo seu nome, CPF/CNPJ, telefone ou *e-mail*, e a busca por amostras, mediante a sua descrição, nome do responsável, finalidade e situação.

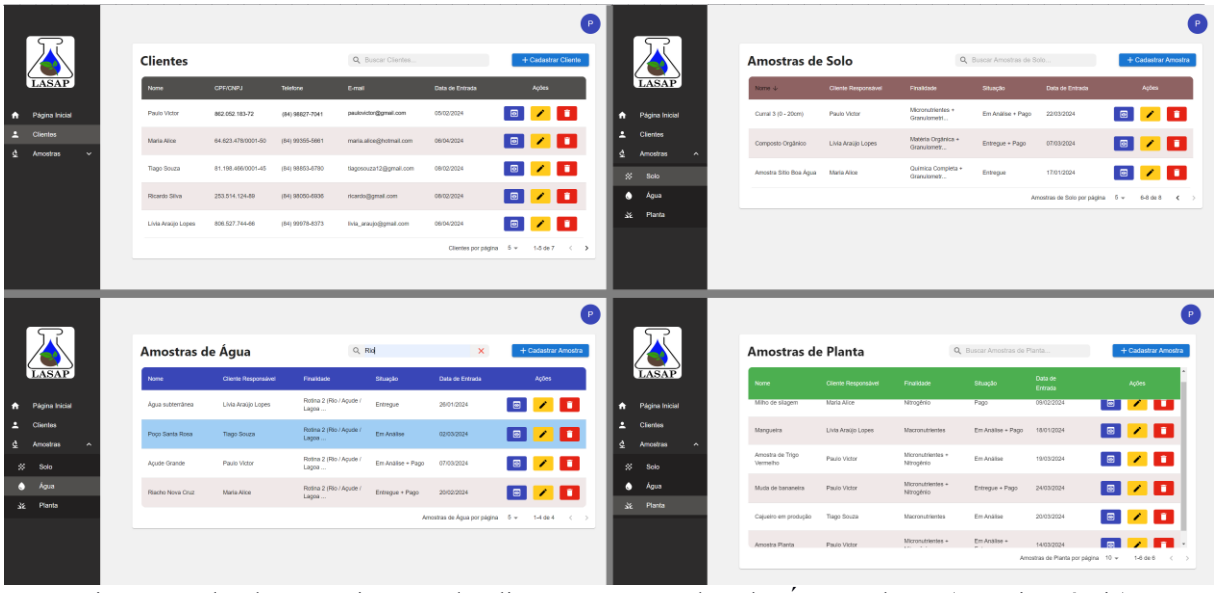

Figura 5. Telas de Gerenciamento de Clientes, Amostras de Solo, Água e Planta. (Autoria Própria).

Em se tratando do cadastro, visualização, edição e exclusão dos registros, na Figura 6 é possível observar a estrutura básica de funcionamento dessas tarefas. O processo é realizado através da manipulação de valores de uma amostra de solo. Na parte superior esquerda da imagem, temos o formulário de cadastro que possui todos os atributos requisitados na Seção 4.1.1. deste trabalho. A fim de representar as propriedades listadas, são dispostos vários campos de digitação e seleção como, menu de opções, caixas de texto, valores decimais e datas. Algumas das informações são definidas automaticamente pelo sistema mediante o preenchimento dos campos vazios ou nulos do formulário. Essas atribuições são feitas com o auxílio de fórmulas e condições presentes na lógica de implementação do sistema. Terminada a inserção dos dados, é possível visualizá-los na própria tabela ou no painel de detalhamento das informações (parte inferior esquerda da imagem), onde são expostos todos os valores dos atributos de uma determinada entidade. Esse painel pode ser acessado a partir do botão azul, localizado na coluna "Ações", correspondente ao registro.

Sobre o processo de edição, o mesmo possibilita que os dados anteriormente cadastrados sejam alterados pelos usuários a qualquer momento. Apresenta poucas mudanças no formulário quando comparado ao cadastro. Ambos possuem ícones de ajuda para o usuário, onde são descritos os conceitos e também os procedimentos utilizados para definir os valores de cada campo. Na parte superior direita da imagem é possível visualizar a descrição da propriedade "Densidade Aparente" ao posicionar o ponteiro do *mouse* sobre o ícone auxiliar. Por fim, o processo de exclusão de um registro, que possui um *pop-up* de confirmação para evitar a deleção acidental de amostras ou clientes do sistema.

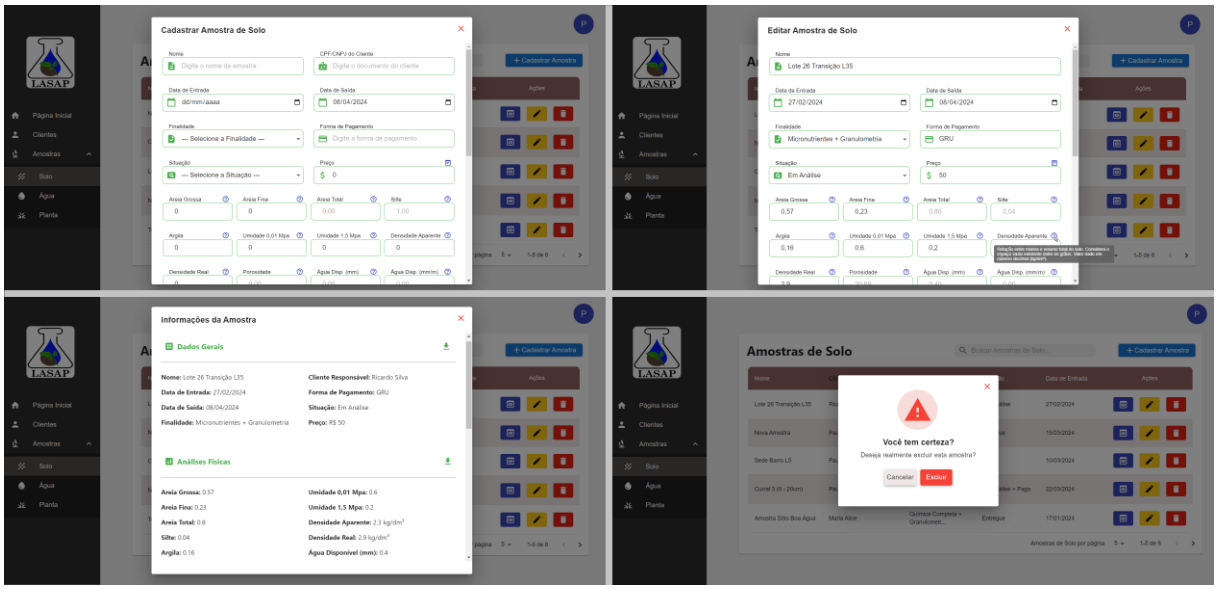

Figura 6. Cadastro, Edição, Visualização e Exclusão de Amostras de Solo. (Autoria Própria).

No quadro de “Informações da Amostra”, é importante destacar a forma de divisão do conteúdo em seções. Para amostras de solo, os valores são agrupados por: “Dados Gerais”, onde ficam as informações base de todas as amostras; “Análises Físicas”, que detalham os resultados do processo de granulometria; “Análises Químicas”, que abriga os valores utilizados no estudo da fertilidade do solo; e “Micronutrientes”, elementos responsáveis pela nutrição de plantas. Para amostras de água, as seções são divididas em “Dados Gerais” e “Análises Químicas”, utilizadas neste contexto para fins de irrigação ou de caldas orgânicas. Para as amostras de planta, os tópicos são “Dados Gerais”, “Análises Químicas”, utilizadas para avaliação do tecido vegetal, e “Metais Pesados”, grupo de elementos com propriedades metálicas presentes no solo.

Para os clientes, o quadro informativo tem uma estrutura um pouco diferente, como é possível visualizar na parte inferior da Figura 7. Sua seção de “Dados Pessoais” segue o mesmo modelo observado no quadro de amostras, porém, aqui as divisões subsequentes são formadas por tabelas que listam todas as amostras de solo, água e planta que são de responsabilidade desse cliente.

Como último requisito a ser explorado, têm-se a geração de boletins. Na Figura 7, parte superior esquerda, uma seta indica o ícone que inicia o processo de emissão do documento. Cada seção gera um documento diferente de acordo com as informações contidas em seu domínio. O arquivo é constituído pelos dados do cliente e das amostras disponíveis no sistema, além de um cabeçalho, campo de assinatura e notas de rodapé.

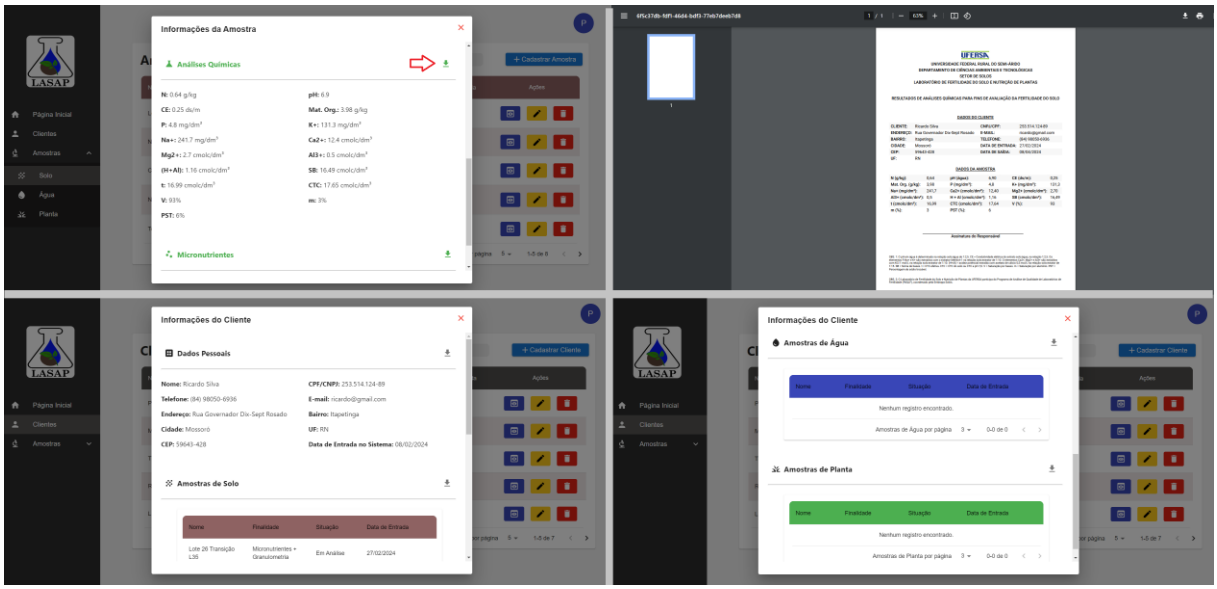

Figura 7. Visualização das Análises Químicas para Avaliação da Fertilidade do Solo, Geração de Boletins e Visualização de Dados do Cliente. (Autoria Própria).

6. CONCLUSÕES

Este projeto foi desenvolvido com o objetivo de entregar uma ferramenta capaz de realizar o gerenciamento das amostras e clientes do LASAP. Com a utilização deste novo produto, os gestores do laboratório teriam um ganho substancial em segurança e organização dos dados. Para este propósito, foi idealizada a construção de um sistema *web* que reunisse todas as informações pertinentes às análises do laboratório, promovendo ao cliente final uma solução moderna que possa ser acessada a partir de qualquer aparelho conectado à *Internet*. Além de disponibilizar as informações em tempo real e permitir o trabalho colaborativo dos usuários.

A aplicação *web* desenvolvida atende as demandas essenciais do laboratório oferecendo as funcionalidades de criação, edição e exclusão, tanto para clientes quanto para amostras de solo, água e planta. Além disso, propicia uma visualização estruturada dos dados, através da utilização de elementos de busca e paginação. Diversos valores presentes no detalhamento das amostras são calculados automaticamente pelo sistema, entregando rapidez e eficiência para as análises dos usuários. O *software* ainda possui um processo de autenticação completo e oferece a geração de boletins, viabilizando o compartilhamento dos dados com os clientes do laboratório.

Como sugestão para trabalhos futuros, é possível citar a inclusão de algumas funcionalidades que trariam ainda mais robustez ao sistema. Uma delas é o pagamento das análises laboratoriais mediante a geração de códigos de barra e *QR codes* dentro da própria aplicação. Também pode ser verificada a possibilidade da integração do *software* com a base de dados da UFERSA, fortalecendo o processo de autenticação ao utilizar a matrícula e o *e-mail* acadêmico no cadastro de usuários. Outra recomendação é incluir fotos das amostras, bem como realizar o seu georreferenciamento, permitindo que a localização do material seja definida com o auxílio de mapas. E por fim, implementar funções mais específicas como, recomendação de adubo para plantas, tratamento da água, fermentação do solo e um levantamento de potenciais problemas encontrados nas amostras.

7. REFERÊNCIAS BIBLIOGRÁFICAS

AMARAL, D.C. et al., Gerenciamento Ágil de Projetos: Aplicação em Produtos Inovadores. 1.ed. São Paulo: Saraiva, 2011.

BECK, K. et al. Manifesto for Agile Software Development. 2001. Disponível em: <http://agilemanifesto.org/>. Acesso em: 29 março 2024.

CONALLEN, J. Building Web applications with UML. 2.ed. Boston: Addison-Wesley, 2000. 300p.

EXPRESS. Express: Fast, unopinionated, minimalist web framework for Node.js. Disponível em: https://expressjs.com/. Acesso em: 05 abril 2024.

LEAU, Y.B. et al. Software development life cycle AGILE vs traditional approaches. In: International Conference on Information and Network Technology, 2012. p. 162-167.

LIE, H.W.; BOS, B. Cascading Style Sheets: Designing for the Web. 3.ed. Indianapolis: Addison-Wesley Professional, 2005.

MATERIAL UI. Material UI – Overview. Disponível em: https://mui.com/material-ui/getting-started/. Acesso em: 05 abril 2024.

NODE.JS. Introduction to Node.js. Disponível em: https://nodejs.org/en/learn/getting-started/introduction-to-nodejs. Acesso em: 05 abril 2024.

NOVACHINSKI, J.R.; SILVA, W.M. Gerenciamento de Analise de Solos com Banco de Dados Relacionais. 2006.

POSTGRESQL. PostgreSQL: About. Disponível em: https://www.postgresql.org/about/. Acesso em: 05 abril 2024.

SILVA, A.R.S.; NASCIMENTO, T.H. Soil Analysis: Sistema para interpretação de análise de solo para produção de lavouras de capim. 2023.

SOMMERVILLE, I. Engenharia de Software. 9.ed. São Paulo: Person Education, 2011.

TAKAKI, L.T.; SAKURAY, F. Sistema de Controle de Atividades Laboratoriais. Anais do Pró-Ensino: Mostra Anual de Atividades de Ensino da UEL, n. 5, p. 249-249, 2023.

REACT. React: The library for web and native user interfaces. Disponível em: https://react.dev/. Acesso em: 05 abril 2024.

SEQUELIZE. Sequelize: Feature-rich ORM for modern TypeScript & JavaScript. Disponível em: https://sequelize.org/. Acesso em: 05 abril 2024.