UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS VII - PATOS
CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS
CURSO DE GRADUAÇÃO EM BACHARELADO EM COMPUTAÇÃO

JOÃO LUCAS DE SOUSA MARTINS

APRIMORAMENTO DE UM SISTEMA DE PESQUISA ELEITORAL E SATISFAÇÃO: UM ESTUDO DE NATUREZA APLICADA

PATOS - PB
2024

JOÃO LUCAS DE SOUSA MARTINS

APRIMORAMENTO DE UM SISTEMA DE PESQUISA ELEITORAL E SATISFAÇÃO: UM ESTUDO DE NATUREZA APLICADA

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Computação do Centro de Ciências Exatas e Sociais Aplicadas da Universidade Estadual da Paraíba, como requisito parcial à obtenção do título de bacharel em Computação.
Área de concentração: Engenharia de Software.

Orientador: Giovanna Trigueiro de Almeida Araujo

PATOS - PB
2024

M386a Martins, João Lucas de Sousa.
Aprimoramento de um sistema de pesquisa eleitoral e satisfação [manuscrito] : um estudo de natureza aplicada / João Lucas de Sousa Martins. - 2024.
41 f. : il.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Ciência da computação) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas, 2024.

"Orientação : Prof. Esp. Giovanna Trigueiro de Almeida Araújo, Coordenação do Curso de Computação - CCEA".

1. Sistema legado. 2. Engenharia de software. 3. Evolução de software. I. Título

21. ed. CDD 004.9

Elaborada por Kaliane Eveny Martins de Oliveira - CRB - 15/986 BSC7

JOAO LUCAS DE SOUSA MARTINS

APRIMORAMENTO DE UM SISTEMA DE PESQUISA ELEITORAL E SATISFAÇÃO: UM ESTUDO DE NATUREZA APLICADA

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Ciência da Computação da Universidade Estadual da Paraíba, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação

Aprovada em: 19/11/2024.

Documento assinado eletronicamente por:

- **Vinícius Augustus Alves Gomes** (***.754.334-**), em **29/11/2024 11:02:45** com chave **9ba1899eae5a11ef8b7006adb0a3afce**.
- **Jucelio Soares dos Santos** (***.475.114-**), em **29/11/2024 15:13:49** com chave **aebd5d3cae7d11eface306adb0a3afce**.
- **Giovanna Trigueiro de Almeida Araújo** (***.352.004-**), em **29/11/2024 19:07:54** com chave **61ec5d70ae9e11ef804a06adb0a3afce**.

Documento emitido pelo SUAP. Para comprovar sua autenticidade, faça a leitura do QrCode ao lado ou acesse https://suap.uepb.edu.br/comum/autenticar_documento/ e informe os dados a seguir.
Tipo de Documento: Termo de Aprovação de Projeto Final
Data da Emissão: 30/11/2024
Código de Autenticação: 265f99

Dedico este trabalho a minha família, amigos, professores e todos que me apoiaram nesta jornada. Este trabalho só foi possível graças a vocês.

AGRADECIMENTOS

Aos meus pais, pela dedicação e amor incondicional. Vocês são minha base e minha força para seguir em frente.

Agradeço à coragem e luta da minha mãe, Maria de Lourdes de Sousa, que ao longo de sua vida se dedicou à minha criação e sempre fez de tudo para o bem-estar de nossa família. Incluo aqui também minhas tias, que contribuíram para o processo de minha criação.

Agradeço também aos meus irmãos, Pedro César de Sousa Martins e Dara Lívia de Sousa Campos, que sempre estiveram ao meu lado, apoiando-me nas decisões para o meu crescimento profissional. Sou igualmente grato aos meus primos, alguns dos quais são como irmãos e que sempre me ajudaram.

Aos amigos que fiz na cidade de Patos-PB, deixo meu carinho por todos os momentos de felicidade e também pelas dificuldades que compartilhamos. Vocês tornaram meus dias muito mais alegres e nos momentos de necessidade foram apoio constante. Levarei todos no meu coração com uma eterna gratidão.

Aos que estiveram ao meu lado, direta ou indiretamente, expresso minha gratidão, pois foram de grande importância em cada situação que enfrentei nesses anos de vida acadêmica.

Aos meus professores e orientadores, pelo conhecimento transmitido e pela orientação ao longo deste trabalho. Sua sabedoria foi fundamental para o meu crescimento acadêmico e profissional.

Apesar das batalhas enfrentadas durante o curso, sempre procurei ver a universidade com bons olhos, valorizando os professores que com suas dedicações diária, ofereceram o melhor de si aos alunos.

RESUMO

Este trabalho apresenta a análise e aprimoração do sistema SisPesquisa, uma plataforma web de Pesquisa Eleitoral e Satisfação, com o objetivo de adaptá-la a um novo cenário de mercado. Originalmente um serviço de uso exclusivo, o SisPesquisa passa a ser comercializado, exigindo ajustes para melhor atender às demandas de um público mais amplo. Para isso, o trabalho foi realizado em etapas, incluindo uma revisão bibliográfica sobre engenharia de software, metodologias ágeis (como Kanban) e as tecnologias empregadas no sistema, seguida de uma análise da arquitetura do sistema legado para identificar pontos de melhoria. O desenvolvimento utilizou tecnologias como Django, HTML, CSS, Bootstrap, JavaScript/jQuery e SQL, implementando novas funcionalidades e corrigindo limitações estruturais para alcançar maior eficiência, segurança e controle de acesso. As melhorias realizadas incluem a criação de regras de negócios mais robustas, um controle de acesso refinado e funcionalidades específicas para diferentes tipos de usuários. O *redesign* do *front-end* e a reestruturação das permissões de acesso garantiram uma interface mais intuitiva e segura, proporcionando maior autonomia para o usuário e um ambiente mais organizado. O usuário, por sua vez, agora possui acesso aos dados e relatórios das pesquisas apenas mediante o que foi criado por sua responsabilidade.

Palavras-chave: Sistema Legado; Engenharia de Software; Evolução de Software.

ABSTRACT

This study presents an analysis and enhancement of the SisPesquisa system, a web platform for Electoral and Satisfaction Survey, with the aim of adapting it to a new market scenario. Originally, SisPesquisa was an exclusive service, but now it is starting to be commercialized, demanding improvements to better serve the demands of a broader audience. For that end, this study was realized in steps, including a bibliographical revision of software engineering, agile methodologies (such as Kanban) and of the technologies utilized in the system. After this step, we analyzed the architecture of the legacy system to identify possible points of improvement. The development utilized technologies such as Django, HTML, CSS, Bootstrap, JavaScript/jQuery and SQL, and implemented new functionalities and corrected structural limitations to achieve better efficiency, safety and access control. The enhancements made include the creation of more robust business rules, refined access control and specific functionalities for different types of users. The front-end redesign and the restructuring of the access permissions guaranteed an interface that is more intuitive and secure, offering greater user autonomy and a more organized program. The user, in turn, now only has access to data and survey reports that were created under their own responsibility.

Keywords: Legacy System; Software Engineering; Software Evolution.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

CSS	*Cascading Style Sheets*
CSRF	*Cross-Site Request Forgery*
DOM	*Document Object Model*
HTML	*Hypertext Markup Language*
ID	*Identificação*
MVC	*Model-View-Controller*
MTV	*Model-Template-View*
ORM	*Object-Relational Mapper*
SQL	*Structured Query Language*
URL	*Uniform Resource Locator*
WIP	*Work in Progress*
XSS	*Cross-Site Scripting*

SUMÁRIO

1 INTRODUÇÃO

Com o passar dos anos, o avanço da tecnologia da informação tem transformado a forma como as organizações gerenciam seus processos e dados. Em um cenário onde a eficiência e a adaptabilidade são fundamentais, sistemas legados — softwares antigos que continuam em uso — muitas vezes enfrentam desafios para atender às novas demandas do mercado e à evolução tecnológica. Tais sistemas, embora cruciais para o funcionamento das empresas, tendem a apresentar dificuldades de manutenção e limitações de escalabilidade. Entretanto, como apontam Pressman e Maxim (2021), esses sistemas frequentemente precisam evoluir por razões como a necessidade de adaptação a novos ambientes ou tecnologias computacionais, ou ainda para atender a novos requisitos de negócio.

Este trabalho tem como objetivo documentar o processo de aprimoramento do sistema SisPesquisa, um sistema legado utilizado para o gerenciamento de pesquisas. Através da análise da arquitetura existente e da aplicação de métodos ágeis, especialmente o Kanban, foi possível implementar melhorias sem comprometer a integridade do sistema. A evolução contínua desse software é essencial para garantir que ele continue atendendo às necessidades dos usuários e se mantenha eficiente em um ambiente tecnológico em constante mudança.

Durante o desenvolvimento deste trabalho, foram aplicadas técnicas de engenharia de software que visam não apenas corrigir falhas, mas também introduzir melhorias funcionais e de desempenho no SisPesquisa. O uso de boas práticas, como a aplicação do Kanban e a documentação detalhada, facilitou o acompanhamento das tarefas e a gestão das modificações realizadas no sistema. Ademais, foi considerado o aspecto da evolução de software, que envolve a adaptação contínua do sistema às novas demandas, garantindo sua longevidade.

Por fim, este trabalho está dividido em seções que abordam desde a fundamentação teórica sobre engenharia de software, evolução de software e metodologias ágeis, até a descrição das tecnologias empregadas no SisPesquisa, como o *framework* Django e o banco de dados MySQL. A documentação das melhorias realizadas no sistema também é detalhada, mostrando o processo de aprimoramento e adaptação para atender às novas necessidades dos usuários e garantir maior eficiência e robustez no uso do sistema.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Aprimoração de um sistema web legado de Pesquisa Eleitoral e Satisfação(SisPesquisa), para que sua finalidade de realizar pesquisas seja incluída em um novo cenário de mercado, onde deixa de ser um serviço de uso exclusivo e passa a comercializar seus serviços.

1.1.2 Objetivos Específicos

- Compreender a arquitetura e as tecnologias do sistema legado, a fim de garantir que as novas implementações sigam os padrões estabelecidos e respeitem a integridade do sistema existente;
- Estudar processos de engenharia de software e metodologias ágeis, com o propósito de aplicar esses conhecimentos no desenvolvimento das melhorias necessárias no sistema;
- Implementar as modificações e melhorias necessárias no sistema, tornando-o visualmente mais robusto e funcional, de forma a atender plenamente às demandas do novo cenário de mercado.

1.2 JUSTIFICATIVA

A partir do momento em que um software é desenvolvido, independentemente de sua finalidade ou do setor em que é utilizado, ele desempenha um papel fundamental na resolução de problemas ou na assistência em atividades onde a tecnologia é essencial. Na área empresarial, existem sistemas que são utilizados diariamente, e muitas empresas possuem seus próprios softwares que, com o passar do tempo, demandam melhorias, correções e a implementação de novas funcionalidades. Como afirma Sommerville (2019), os sistemas de software se adaptam e evoluem ao longo de sua vida útil, desde a implantação inicial até a desativação.

Nesse sentido, este trabalho tem como motivação aprimorar um software legado que anteriormente era utilizado de forma exclusiva no mercado, sem a possibilidade de terceirização, mas que agora apresenta um benefício maior ao ser comercializado, permitindo que qualquer usuário (no caso, contratante do sistema) o utilize para realizar pesquisas eleitorais ou de satisfação.

Portanto, este estudo movimenta-se pelo interesse em demonstrar como é possível transformar um sistema legado, independentemente de sua finalidade, por meio do conhecimento das tecnologias empregadas em sua construção, aliado a processos de engenharia de software e metodologias ágeis, que organizam os passos necessários para o aprimoramento desse sistema.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Conceitos Fundamentais

A seguir, serão discutidos os conceitos de sistemas legados, engenharia de software, métodos ágeis e evolução de software, que formam o alicerce para as práticas adotadas no desenvolvimento e na evolução de sistemas como o SisPesquisa.

Bem como, a definição de sistema legado, ou software legado, pode variar dependendo do autor, apresentando diferentes características. Segundo Sommerville (2019), um sistema legado é essencial para as empresas, embora tenha sido desenvolvido com tecnologias ou métodos ultrapassados. Esses sistemas, apesar de antigos, ainda executam funções críticas para o empreendimento.

Pressman e Maxim (2021) complementam essa definição ao destacar que um sistema legado pode não ser expansível para modificações, apresentar códigos complexos e possuir pouca ou nenhuma documentação. Além disso, frequentemente conta com um histórico de implementações mal organizadas. Mesmo assim, esses sistemas permanecem vitais para o negócio, sendo indispensáveis.

Embora as definições apresentem pontos diferentes, ambas convergem em um aspecto específico: os sistemas legados são de suma importância para as organizações, pois continuam a gerenciar processos empresariais essenciais. Além disso, o que os torna ainda mais valiosos é o fato de serem repositórios de conhecimento corporativo, coletados ao longo de anos, que possuem as lógicas de processos da organização (Stroulia *et al.*, 1999).

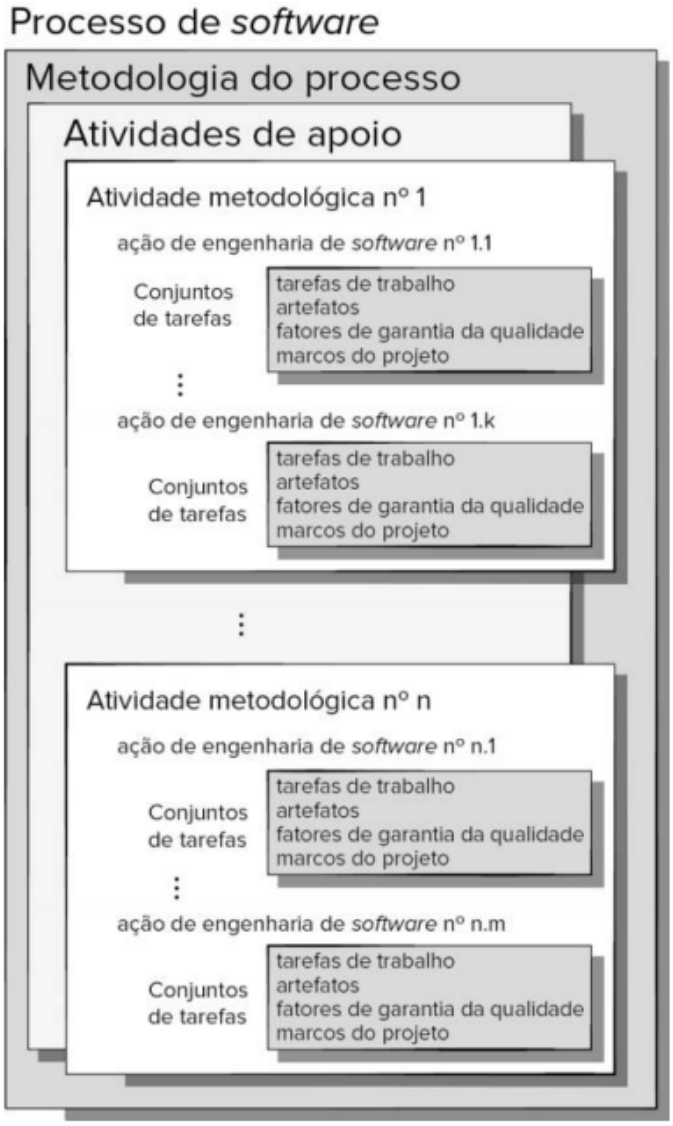
2.1.1 Engenharia de Software

A engenharia de software é uma área da computação, responsável pelo desenvolvimento e manutenção de sistemas de software. Ela abrange um conjunto de processos, métodos e ferramentas que visam garantir que o sistema atenda às expectativas dos clientes e cumpra sua finalidade. Embora o desenvolvimento de sistemas corporativos e a criação de jogos com gráficos avançados possam ter objetivos distintos, ambos se beneficiam das práticas de engenharia de software, aplicadas com diferentes técnicas (Sommerville, 2019).

Partindo para os processos de software, estes delineiam os passos necessários para cada atividade durante a produção de um produto de software. Tais atividades podem envolver o desenvolvimento de software a partir do zero, no entanto, nem todas as aplicações seguem esse caminho, já que algumas são desenvolvidas a partir de sistemas pré-existentes (Sommerville, 2011). De acordo com Pressman e Maxim (2021), “Cada uma dessas atividades, ações e tarefas se aloca dentro de uma metodologia ou modelo que determina sua relação com o processo e uma com a outra”.

A figura a seguir apresenta esquematicamente o processo de software e como as atividades estão organizadas.

Figura 1 – Processo de software



Fonte: (Pressman; Maxim, 2021).

De acordo com a imagem acima, cada atividade no processo de software consiste em um conjunto de ações de engenharia de software, e essas ações, por sua vez, são compostas por tarefas específicas que delineiam o que deve ser alcançado. Em complemento, essas tarefas são projetadas para garantir que os resultados do software sejam feitos com a qualidade desejada, dentro dos prazos estabelecidos, permitindo o monitoramento contínuo do progresso no desenvolvimento.

Como os processos de software consistem em uma sequência de atividades técnicas destinadas a colaborar e gerenciar o desenvolvimento, seu objetivo é especificar, projetar, implementar e testar a aplicação. Por outro lado, é comum o uso de ferramentas que auxiliam na organização desses processos, permitindo que programadores utilizem softwares para gerenciamento de requisitos, prototipação de design, depuração e testes automatizados (Sommerville, 2019).

As atividades do processo de software começam com a especificação, onde os requisitos do sistema e suas funcionalidades são definidos. Nesta etapa, o objetivo é capturar as necessida-

des dos usuários e documentar as expectativas em relação ao software. Em seguida, o projeto e a implementação tratam da tradução desses requisitos em uma arquitetura de sistema e na codificação propriamente dita. Quando uma abordagem ágil é utilizada, o projeto e a implementação são intercalados, sem que documentos formais de design sejam necessariamente produzidos, promovendo uma flexibilidade maior no desenvolvimento (Sommerville, 2019).

A validação do software ocorre para garantir que o sistema desenvolvido atende aos requisitos especificados. Essa fase envolve uma série de testes, revisões e verificações. Finalmente, a evolução do software considera as modificações e melhorias que o sistema pode necessitar ao longo do tempo, para continuar atendendo às necessidades dos usuários e do mercado. Essas atividades são cíclicas e interativas, permitindo que o software evolua de acordo com novas demandas (Sommerville, 2019).

Os processos de software podem passar por melhorias e mudanças devido à alta demanda da indústria por software mais eficiente, econômico e que seja desenvolvido em prazos mais curtos. Uma abordagem ágil apoia esse processo, pois foca em um trabalho interativo, com baixo custo, entrega rápida de requisitos e agilidade nas alterações de funcionalidades conforme as necessidades do cliente (Sommerville, 2019).

2.1.2 Métodos Ágeis

No ano de 2001, programadores, autores e consultores renomados assinaram o "Manifesto para o Desenvolvimento Ágil de Software", no qual defendiam interações que estavam fora dos processos tradicionais de software, priorizando a aplicação executando funcionalidades sem extensa documentação, o envolvimento contínuo dos clientes em novas implementações e a capacidade de responder a mudanças no sistema sem seguir atividades rigidamente prescritas (Pressman; Maxim, 2021).

Segundo Filho (2019), "Métodos ágeis constituem um grupo de metodologias diferentes entre si, mas caracterizadas por princípios comuns, mais baseados no trabalho cooperativo do que no formalismo e na documentação escrita". Além disso, todos os métodos ágeis compartilham a característica de que o software é desenvolvido e entregue incrementalmente (Sommerville, 2019).

Os processos ágeis para Valente (2020):

> A característica principal de processos ágeis é a adoção de ciclos curtos e iterativos de desenvolvimento, por meio dos quais um sistema é implementado de forma gradativa; começando por aquilo que é mais urgente para o cliente. De início, implementa-se uma primeira versão do sistema, com as funcionalidades que segundo o cliente são para ontem, isto é, possuem prioridade máxima. Em seguida, essa versão é validada pelo cliente. Se ela for aprovada, um novo ciclo — ou iteração — inicia-se, com mais algumas funcionalidades, também priorizadas pelos clientes. Normalmente, esses ciclos são curtos, com duração de um mês, talvez até um pouco menos. Assim, o sistema vai sendo construído de forma incremental, sendo cada incremento devidamente aprovado pelos clientes. O desenvolvimento termina quando o cliente decide que todos os requisitos estão implementados.

Atualmente, as empresas operam em ambientes de rápida mudança, onde é necessário responder rapidamente aos novos cenários de mercado e às inovações dos concorrentes. Como o software está intimamente ligado aos negócios, ele precisa ser desenvolvido com agilidade (Sommerville, 2019). Em muitas situações, os requisitos do projeto podem não estar completamente definidos antes do início do desenvolvimento. Portanto, é necessário adotar um desenvolvimento ágil para que o software seja capaz de se adaptar a um ambiente dinâmico (Pressman; Maxim, 2021).

2.1.2.1 Princípios da Agilidade

A *Agile Alliance* define 12 princípios que orientam o desenvolvimento ágil de software. Esses princípios visam assegurar a agilidade e adaptabilidade no desenvolvimento, enfatizando a entrega rápida de valor ao cliente e a capacidade de responder às mudanças. Uma das principais ideias é que o software deve ser entregue frequentemente, em pequenos incrementos, permitindo o feedback contínuo de todas as partes interessadas (Pressman; Maxim, 2021).

Além disso, os métodos ágeis priorizam a comunicação direta entre os membros da equipe, que são motivados e auto-organizados. O ambiente de trabalho deve ser propício ao desenvolvimento de software de alta qualidade, com foco em excelência técnica e simplicidade no design, eliminando trabalhos desnecessários. Esses princípios garantem que o desenvolvimento siga um ritmo sustentável, permitindo a continuidade do trabalho por longos períodos. Por fim, a introspecção constante da equipe sobre seu desempenho busca melhorar continuamente a forma como os objetivos são alcançados (Pressman; Maxim, 2021).

2.1.2.2 Método ágil Kanban

O Kanban é um método que auxilia equipes de desenvolvimento a manter um ritmo de trabalho sustentável, minimizando desperdícios, entregando valor frequentemente e promovendo uma cultura de melhoria contínua (Anderson, 2013). No contexto do desenvolvimento de software, este método também utiliza um "Quadro Kanban", que "é dividido em colunas", representando etapas específicas do processo de desenvolvimento, como *backlog*, especificação e implementação. Cada tarefa avança pelo quadro à medida que os membros da equipe "puxam"o trabalho para a próxima etapa, caracterizando esse método como um sistema *pull* (Valente, 2020).

Nesse contexto, é uma metodologia enxuta que descreve métodos para melhorar qualquer processo ou fluxo de trabalho. Focado na gestão de alterações e na entrega de serviços, o Kanban auxilia equipes a gerenciar mudanças solicitadas e integrá-las a sistemas baseados em software, ao mesmo tempo que enfatiza o entendimento das necessidades e expectativas dos clientes. A gestão é feita pelos membros da equipe, que têm liberdade para se organizar e concluir o trabalho, enquanto as políticas evoluem conforme necessário para otimizar os resultados (Pressman; Maxim, 2021).

Esse método foi inicialmente desenvolvido pela Toyota como parte de práticas de engenharia industrial, sendo posteriormente adaptado ao desenvolvimento de software por David

Anderson (Pressman; Maxim, 2021). O método se baseia em seis práticas fundamentais:

1. **Visualizar o fluxo de trabalho:** Um quadro Kanban, dividido em colunas que representam os estágios de desenvolvimento do software, é utilizado para acompanhar o progresso das funcionalidades. As tarefas são movimentadas entre colunas como "por fazer", "fazendo"e "feito", permitindo que a equipe visualize o fluxo de trabalho;

2. **Limitar o WIP (*Work in Progress*):** O Kanban limita a quantidade de trabalho em progresso, incentivando os desenvolvedores a concluírem suas tarefas antes de começarem novas. Isso reduz o tempo de ciclo e melhora a qualidade das entregas, permitindo uma maior frequência de entregas para os envolvidos;

3. **Gerenciar o fluxo de trabalho:** A gestão do fluxo visa reduzir desperdícios, compreendendo e ajustando o fluxo de valor por meio da análise de interrupções, implementação de mudanças e medição de seus efeitos no processo;

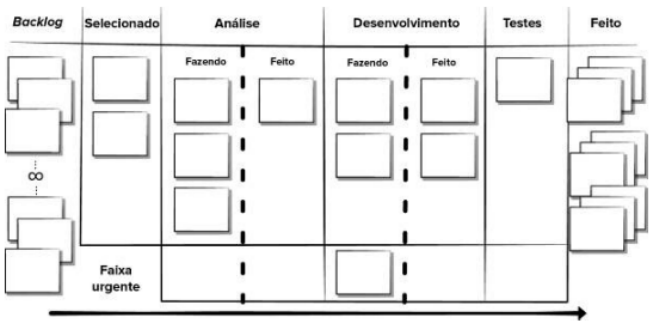
4. **Explicitar políticas de processo:** Definir claramente os motivos pelos quais determinados itens foram selecionados para serem trabalhados e os critérios para considerar uma tarefa como "feita";

5. **Foco na melhoria contínua:** Por meio de ciclos de *feedback*, são introduzidas mudanças baseadas em dados de processo, e seus impactos são medidos para garantir uma melhoria constante;

6. **Mudança colaborativa:** Alterações no processo devem ser feitas de forma colaborativa, envolvendo todos os membros da equipe e demais interessados sempre que necessário.

Essa adaptação do Kanban para o desenvolvimento de software tem se mostrado eficiente em ajudar equipes a manterem um ritmo sustentável e a entregarem valor com mais frequência (Pressman; Maxim, 2021).

Figura 2 – Exemplo de quadro Kanban



Fonte: (Pressman; Maxim, 2021).

2.1.3 Evolução de Software

Sommerville (2019) destaca que os sistemas de software precisam evoluir ao longo de sua vida útil para permanecerem úteis, uma vez que as mudanças nas empresas e nas expectativas dos usuários geram novos requisitos. Essa evolução pode ocorrer para corrigir erros, adaptar-se a novas plataformas e melhorar o desempenho ou outras características não funcionais. É importante mencionar que a evolução do software é crucial para as organizações, uma vez que elas investem grandes quantias de dinheiro em seus sistemas e dependem deles para suas operações diárias.

A evolução de software não segue um processo linear ou com fases claramente delimitadas. Modelos iterativos de desenvolvimento sugerem que um conjunto completo de requisitos de um sistema raramente pode ser entendido no início, ou que os desenvolvedores ainda não sabem exatamente como construir o sistema completo. Por isso, os sistemas são construídos em ciclos iterativos, ou *builds*, onde cada versão é uma melhoria da anterior, refinada a partir do *feedback* dos clientes (Tripathy; Naik, 2014). Nesse contexto, as atividades de manutenção e evolução não existem como fases distintas, mas estão intrinsecamente conectadas durante todo o ciclo de desenvolvimento.

No desenvolvimento de software, o modelo de mini-ciclo de mudanças, divide o processo de evolução em cinco fases principais: solicitação de mudança, análise e planejamento da mudança, implementação, verificação e validação, e alteração da documentação. Esse modelo destaca atividades importantes no ciclo de mudanças, como a compreensão do programa, análise de impacto, refatoração e propagação de mudanças (Tripathy; Naik, 2014). A partir desse modelo, é notório como iniciar e finalizar cada iteração de mudança no sistema, realizando, por exemplo, uma análise de código, modificação no código e revalidação do mesmo.

Para gerenciar a evolução de sistemas legados como SisPesquisa, segundo Tripathy e Naik (2014), existem algumas opções comumente disponíveis. A opção considerada é *carry on maintenance*, que apesar dos desafios associados à manutenção de sistemas legados, a organização opta por continuar a realizá-la por um determinado período. Essa decisão geralmente é tomada quando o sistema ainda desempenha um papel crucial para a operação do negócio, e sua substituição imediata não é viável, seja devido a restrições orçamentárias, falta de recursos técnicos ou riscos associados à migração para uma solução moderna.

2.1.3.1 Manutenção de Software

A manutenção de software é um processo que se inicia quase imediatamente após a liberação do software para os usuários. Conforme mencionado por Pressman e Maxim (2021), "o software é liberado para os usuários e, em alguns dias, os relatos de erros começam a chegar à empresa de engenharia de software". Além de correções, usuários identificam a necessidade de adaptações para atender aos seus ambientes específicos, e novos grupos corporativos podem reconhecer as vantagens do software, demandando melhorias para integrá-lo às suas operações.

Dessa forma, a manutenção se torna essencial não apenas para corrigir defeitos, mas também para garantir a relevância e a adaptabilidade do software às novas demandas do mercado e das organizações.

De acordo com Sommerville (2019), uma maneira eficaz de facilitar a manutenção de sistemas legados é realizar a reengenharia desses sistemas, com o objetivo de melhorar sua estrutura e compreensibilidade. Esse processo pode envolver a nova documentação, refatoração da arquitetura, tradução para linguagens de programação mais modernas, além da atualização dos dados. Apesar de a funcionalidade do sistema não ser alterada, e grandes mudanças na arquitetura serem evitadas, a reengenharia apresenta duas grandes vantagens em relação à substituição do sistema: menor risco e menor custo. Refazer um software crítico envolve riscos elevados, como erros na especificação ou problemas no desenvolvimento, além de possíveis atrasos que podem resultar em perdas financeiras. Ainda assim, o custo de reengenharia pode ser significativamente inferior ao de uma reimplementação completa. Um exemplo citado por Ulrich (1990) ilustra essa diferença, onde um sistema que teria um custo estimado de US$ 50 milhões para ser reimplementado foi reengenheirado com sucesso por US$ 12 milhões. Mesmo com avanços tecnológicos, o custo da reimplementação ainda tende a ser superior ao da reengenharia.

2.1.4 UX e Usubilidade

Para que um sistema de software seja bem-sucedido, ele deve proporcionar uma experiência do usuário (UX - *User Experience*) positiva, o que depende diretamente de sua usabilidade. De acordo com Pressman e Maxim (2021), a usabilidade é uma medida qualitativa que avalia a facilidade e a eficiência com que os usuários conseguem utilizar as funções e recursos de um sistema. E deve garantir que a tecnologia seja adaptada às necessidades das pessoas.

Portanto, na aprimoração do SisPesquisa buscou-se seguir alguns dos primeiros princípios de design de interação propostos por Tognozzi (2001), que conduzem a uma melhor usabilidade:

- **Estética:** A moda nunca deve superar a usabilidade, assim, uma interface pode passar por mudanças significativas no design visual e funcional sem comprometer a produtividade;
- **Antecipação:** Uma aplicação deve ser projetada para prever o próximo passo do usuário, de modo que o software deve tentar antecipar os desejos e necessidade do usuário;
- **Autonomia:** A interface deve facilitar a movimentação do usuário pela aplicação, mas deve fazê-lo de forma que faça valer as convenções de navegação estabelecidas para a aplicação;
- **Consistência:** O uso de controles de navegação, menus, ícones e estética (p. ex., cor, forma, layout) deve ser consistente, e as aparência dos objetos precisa ser rigorosamente controlada para que as pessoas não gastem metade do tempo tentando descobrir como rolar ou imprimir;

- **Eficiência do Usuário:** Observe a produtividade do usuário, não a do computador. Embora possa parecer que aumentar a produtividade da máquina deve resultar em aumento da produtividade humana, o oposto geralmente é verdadeiro;
- **Capacidade de Aprendizagem:** A interface de uma aplicação deve ser projetada para minimizar o tempo de aprendizagem e, uma vez aprendida, minimizar a reaprendizagem necessária quando a aplicação for reutilizada.

2.2 Tecnologias

Nesta seção, serão apresentadas as principais tecnologias que sustentam o SisPesquisa, e ferramentas que continuam sendo essenciais para seu aprimoramento. O uso dessas tecnologias tem sido fundamental para garantir a escalabilidade, eficiência e a manutenção contínua do sistema ao longo dos anos.

2.2.1 Framework Django

Django é um *framework* web de alto nível, baseado na linguagem Python, criado com o objetivo de facilitar o desenvolvimento de aplicações web complexas de forma rápida e eficiente. O projeto começou em 2003, como parte de uma iniciativa de Adrian Holovaty e Simon Willison no *Journal-World*, um jornal em Lawrence, Kansas (Rubio, 2017).

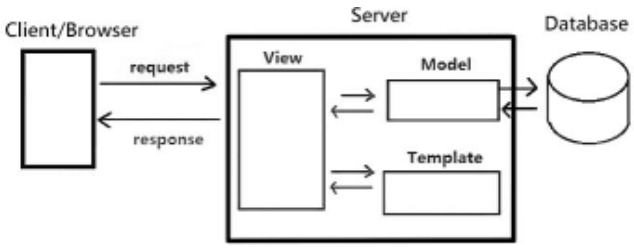
A força do Django como *framework* de desenvolvimento web está na sua filosofia de desenvolvimento rápido e modular. Seguindo o princípio do *Don't Repeat Yourself* (DRY), o Django evita a repetição de código e lógica através de sua estrutura modular, o que o torna uma escolha robusta e eficiente para criar aplicações web complexas. Django também se manteve fiel ao padrão *Model-View-Controller* (MVC), permitindo uma organização clara entre a lógica do sistema, a visualização e os dados (Rubio, 2017).

Embora o Django siga o padrão de arquitetura *Model-View-Controller* (MVC), ele o adapta para uma variação conhecida como *Model-Template-View* (MTV). O *model* é responsável pela gestão dos dados e pela lógica de negócios principal, enquanto o *template* lida com a exibição desses dados aos usuários. Já a *view*, no Django, atua como o "controlador", processando a lógica da aplicação e as requisições de entrada. Essa adaptação do MVC, que prioriza a simplicidade e a rapidez, facilita a manutenção e a escalabilidade do código, separando de maneira clara as responsabilidades do sistema, sem comprometer a flexibilidade e a reutilização de componentes (Alchin, 2013).

A arquitetura MTV (*Model-Template-View*), que o Django utiliza para estruturar aplicações web, é ilustrada na Figura 3. Essa arquitetura separa a lógica de negócios (*Model*), a lógica de apresentação (*Template*) e o controle de interação do usuário (*View*).

Além da variação *Model-Template-View* (MTV), o Django pode ser descrito de forma mais precisa com o padrão MVTU, que inclui a configuração de URLs como um elemento-chave da arquitetura (Vincent, 2022). O padrão segue a seguinte estrutura:

Figura 3 – Arquitetura MTV



Fonte: (Yu *et al.*, 2023).

- ***Model*:** Gerencia os dados e a lógica de negócios principal;
- ***View*:** Define quais dados são enviados ao usuário, mas não a forma como são apresentados;
- ***Template*:** Responsável pela apresentação dos dados em HTML, com CSS, JavaScript e outros recursos estáticos;
- ***URL Configuration*:** Mapeia os componentes da aplicação para as *views* usando expressões regulares.

Essa abordagem proporciona flexibilidade e clareza na separação das responsabilidades, sem comprometer a eficiência ou escalabilidade do sistema.

O Django é conhecido por oferecer uma série de ferramentas e padrões de design que tornam o desenvolvimento web mais seguro. Entre essas funcionalidades, destacam-se o uso do ORM (*Object-Relational Mapping*) que protege contra ataques de SQL *Injection*, a utilização de tokens CSRF (*Cross-Site Request Forgery*) para prevenir ataques de falsificação de requisições, e a proteção contra ataques XSS (*Cross-Site Scripting*) por meio de seu sistema de *templates*. Essas técnicas tornam os serviços desenvolvidos com Django mais seguros, além de elevar o nível de profissionalismo dos desenvolvedores que utilizam corretamente seus métodos e configurações (Duisebekova *et al.*, 2021).

No Django, o *Object-Relational Mapper* (ORM) é uma das suas características mais poderosas, permitindo interações com o banco de dados utilizando comandos em Python para armazenar e recuperar dados. O ORM facilita a execução de consultas SQL como declarações em Python, retornando os resultados também nesse formato, o que simplifica o desenvolvimento e a manutenção de aplicações. Dessa forma, ao receber uma requisição, o Django redireciona o pedido entre os arquivos responsáveis (*urls.py, views.py*) e, caso necessário, interage com o banco de dados por meio do ORM para processar a resposta (Vamsi *et al.*, 2021).

Além das características já mencionadas, o Django se destaca por sua abordagem '*batteries-included*', que oferece uma ampla gama de funcionalidades nativas para o desenvolvimento web. Entre os principais recursos, destacam-se:

- Autenticação de usuários;
- Testes;

- Modelos de banco de dados, formulários, rotas de URL e *templates*;
- Interface administrativa;
- Atualizações de segurança e desempenho;
- Suporte a múltiplos *backends* de banco de dados.

Essa abordagem permite que os desenvolvedores concentrem seus esforços no que torna a aplicação única, sem precisar reinventar funcionalidades básicas (Vincent, 2022).

2.2.2 HTML, CSS, JavaScript/jQuery e Bootstrap

No aprimoramento do SisPesquisa, as tecnologias de *front-end* empregadas nos *templates* do sistema foram mantidas, o que permite criar uma interface responsiva, interativa e eficiente. O uso de HTML, CSS, JavaScript/jQuery e Bootstrap em conjunto com o *framework* Django contribui para garantir a funcionalidade e a usabilidade do sistema.

O HTML (*Hypertext Markup Language*) é a base para a estruturação das páginas web, permitindo a organização de elementos como títulos, parágrafos e listas. Ele foi inicialmente criado para facilitar a troca de informações científicas, mas hoje é amplamente utilizado para formatação de páginas na web (Shyam; Mukesh, 2020).

Para estilização, o CSS (*Cascading Style Sheets*) é empregado para controlar a apresentação dos elementos HTML, tornando possível gerenciar o layout de várias páginas de maneira centralizada, economizando esforço e tempo (Shyam; Mukesh, 2020).

A linguagem JavaScript (JS), por sua vez, adiciona interatividade às páginas web, permitindo validação de formulários no lado do cliente e, consequentemente, um processamento de dados mais rápido. Sua ampla adoção e versatilidade fazem dele uma peça-chave nas aplicações web modernas (Shyam; Mukesh, 2020). Para facilitar o uso do JavaScript, o jQuery é utilizado como uma biblioteca que simplifica o código e manipulação do DOM.

Por fim, o Bootstrap é usado para garantir um design responsivo e voltado para dispositivos móveis. Esse *framework* fornece *templates* prontos para componentes de interface, como formulários, botões e navegação, facilitando o desenvolvimento de uma interface de usuário padronizada e eficiente (Shyam; Mukesh, 2020).

2.2.3 Banco de Dados MySQL

O MySQL é o banco de dados utilizado pelo sistema SisPesquisa. Segundo Vincent (2022), o Django ORM (*Object-Relational Mapper*) oferece suporte integrado a diversos bancos de dados, como o MySQL, permitindo que os desenvolvedores escrevam código Python no arquivo *models.py*, que automaticamente será traduzido para SQL e compreendido pelo banco de dados.

Vale ressaltar que, conforme Yu e Yang (2019), MySQL é um sistema de gerenciamento de banco de dados relacional amplamente utilizado, especialmente em aplicações web. Ele utiliza

a linguagem SQL, uma das mais padronizadas para o acesso a bancos de dados. Devido ao seu pequeno tamanho, alta velocidade, baixo custo e código aberto, MySQL é frequentemente escolhido como banco de dados em sistemas de pequeno e médio porte.

3 APRESENTAÇÃO DO SISPESQUISA

O SisPesquisa é um sistema web desenvolvido para o gerenciamento de pesquisas, especialmente nas categorias Eleitoral e de Satisfação. Sua criação teve como principal objetivo otimizar e simplificar os processos de gestão dessas pesquisas, oferecendo ferramentas de análise avançada que auxiliam na interpretação e uso estratégico dos dados obtidos.

No sistema, é possível cadastrar diversas informações fundamentais para a execução das pesquisas. Os usuários têm a capacidade de adicionar novas pesquisas, preenchendo dados essenciais para sua finalidade. Entre as informações obrigatórias estão: o município onde a pesquisa será realizada, o tipo da pesquisa, sua descrição, datas de início e encerramento, além de elementos que compõem a base de dados, como perguntas, setores, localidades e pesquisadores. Esses registros formam a estrutura necessária para garantir a coerência e completude das informações.

Em complemento, após a conclusão de cada pesquisa, o sistema disponibiliza relatórios detalhados que consolidam todas as respostas coletadas. Esses relatórios incluem visualizações automatizadas, gráficos dinâmicos e análises estatísticas avançadas, permitindo aos usuários acesso a *insights* aprofundados e suporte na tomada de decisões estratégicas com maior rapidez e precisão.

As ações de gerenciamento eram realizadas por usuários com acesso ao sistema, sendo a única distinção entre eles o nível de permissão de Super Usuário. O Super Usuário possuía acesso a todas as pesquisas do sistema, enquanto o usuário padrão tinha acesso restrito apenas às pesquisas que ele próprio havia criado.

No entanto, existia uma vulnerabilidade que permitia que um usuário padrão acessasse dados de pesquisas alheias caso soubesse o ID da pesquisa. Isso podia ser feito simplesmente alterando o caminho da URL que tivesse o ID desejado como parâmetro. Essa falha foi identificada e devidamente corrigida durante o processo de aprimoração do sistema.

Inicialmente, o público-alvo do SisPesquisa era composto por contratantes, como prefeitos e vereadores, interessados em obter dados detalhados sobre suas campanhas eleitorais ou índices de satisfação. Para esses usuários, o sistema fornecia acesso exclusivo aos relatórios, garantindo uma visão clara e organizada dos resultados de suas pesquisas.

É importante ressaltar que o autor deste trabalho não participou do desenvolvimento inicial do SisPesquisa durante sua criação. O sistema foi recebido em um estágio funcional e em operação, servindo como ponto de partida para as análises e aprimoramentos realizados.

3.1 Motivo da Aprimoração do SisPesquisa

A empresa *Insight Technology* GPC identificou a oportunidade de comercializar o sistema para clientes que desejassem realizar pesquisas de forma independente e disponibilizá-las a seus contratantes. Essa nova abordagem no cenário de mercado exigiu adaptações no sistema, que anteriormente era de uso exclusivo. A forma de gerenciar as pesquisas continua a mesma, porém,

tornou-se necessário implementar diversas alterações e correções, incluindo controle de acesso às informações, para atender às expectativas dos novos clientes.

A seguir, são destacados os principais pontos de aprimoramento:

- Garantir que os dados criados no sistema sejam tratados de forma específica para cada usuário responsável por eles;
- Apresentar as informações corretas de acordo com o usuário logado no sistema;
- Implementar novas regras de negócio para adequar o sistema às demandas dos clientes;
- Estabelecer regras de acesso para os novos usuários;
- Redesenhar a interface e criar telas para acomodar as novas funcionalidades do sistema;
- Desenvolver funcionalidades adicionais para otimizar os processos no sistema.

Desse modo, o público-alvo do sistema passou a incluir um novo perfil: clientes que utilizam praticamente todas as funcionalidades do sistema.

4 DESENVOLVIMENTO DA APRIMORAÇÃO DO SISPESQUISA

O processo de aprimoração do sistema SisPesquisa foi documentado de maneira organizada e estruturada utilizando o software web KanbanTool. Esse software permitiu o gerenciamento eficaz das tarefas relacionadas ao desenvolvimento, distribuídas em um quadro Kanban, no qual cada tarefa (*card*) representava uma alteração ou uma nova funcionalidade implementada no sistema.

Para cada tarefa, foram documentados os passos necessários para sua execução, incluindo descrições detalhadas das alterações, imagens ilustrativas de como as telas deveriam ser exibidas após as modificações, scripts SQL associados às operações no banco de dados e diagramas SQL quando necessário. Esse processo facilitou a visualização das mudanças e garantiu que cada etapa fosse seguida conforme planejado.

Nesta seção, serão descritas as principais funcionalidades aprimoradas, cada uma referenciada pela tarefa correspondente no KanbanTool.

4.1 Plano de Contrato

Nesta tarefa, foi desenvolvida a lógica inicial para permitir que o Cliente do sistema possa realizar suas pesquisas de forma personalizada. Para isso, foram implementados novos elementos no banco de dados, relacionados ao tipo de contrato e à identificação do Cliente que utiliza o sistema. Um modelo foi criado com um relacionamento 1:1 (um para um) com o modelo de autenticação padrão do Django. Dessa forma, quando o Cliente estiver autenticado no sistema, ele poderá acessar informações relacionadas ao contrato e verificar limitações e permissões específicas para a realização de novas pesquisas.

No desenvolvimento, foram incluídas regras de negócio que definem o comportamento do Cliente no sistema. Uma lógica foi implementada para verificar o status da empresa vinculada ao Cliente, restringindo o acesso caso a empresa esteja inativa ou possua restrições contratuais, como saldo negativo, impedindo a criação de novas pesquisas ou o acesso a determinados dados.

Para usuários com permissões administrativas, foi garantido acesso a informações gerais dos clientes sob sua gestão, permitindo um controle eficiente sobre as pesquisas realizadas e a situação contratual. Além disso, o painel principal foi ajustado para exibir informações personalizadas, garantindo uma interface adequada tanto para os Clientes quanto para os administradores.

4.2 Módulo Pesquisa

Nesta tarefa, foi realizado um aprimoramento na interface dos formulários de criação e edição de pesquisas, resultando em um design mais intuitivo e acessível para os usuários. Foram implementadas validações nos campos do formulário, como a exigência de um número mínimo de caracteres na descrição e a verificação de consistência nas datas, garantindo que a data de encerramento seja posterior à data de início. Também, foi adicionada uma funcionalidade que

permite a personalização visual das pesquisas, com a possibilidade de associar uma imagem de capa para identificar cada pesquisa de forma única.

4.3 Módulo Perguntas

Nesta tarefa, foi realizada uma melhoria na interface do módulo de perguntas da pesquisa, com o objetivo de proporcionar um design mais claro e funcional para os usuários. Os modais para criação, remoção e edição de perguntas foram reformulados como parte desse aprimoramento. Na funcionalidade de edição, foi incluído um campo que permite definir a ordem das perguntas, exigindo ajustes nas regras de processamento e no formulário correspondente, de forma a refletir corretamente essa nova funcionalidade.

Adicionalmente, foram implementados mecanismos para garantir que apenas perguntas criadas pelo responsável pela pesquisa sejam exibidas em campos de seleção. Para suportar essa funcionalidade, foi necessário modificar a estrutura interna do sistema, incluindo um campo que permite associar cada pergunta ao seu respectivo criador. Isso assegura que a filtragem dos dados seja realizada de maneira consistente e adequada.

4.4 Módulo Localidade

Nesta tarefa, foi realizada uma melhoria na interface do módulo de localidades da pesquisa, com o objetivo de oferecer um design mais claro e acessível para os usuários. Os modais para criação, edição e remoção de setores e localidades foram reformulados como parte desse aprimoramento. Na funcionalidade de edição de localidades, foram adicionados campos que permitem detalhar informações específicas, como número de casas e quantidade a entrevistar, o que exigiu ajustes nos processos internos para refletir essas novas informações.

Também foi implementada a funcionalidade de remoção em lote, permitindo que o usuário selecione e remova localidades de forma individual ou coletiva, facilitando a gestão dos dados.

Para assegurar que cada localidade e setor estejam vinculados corretamente, o sistema passou a associar essas informações ao responsável pela pesquisa no momento da criação. Esse vínculo possibilita a exibição de dados filtrados, como setores e localidades, com base no criador da pesquisa, garantindo que apenas os dados relevantes sejam apresentados ao usuário.

4.5 Módulo Pesquisador

Nesta tarefa, foi realizada uma melhoria na interface do módulo de pesquisadores associados às localidades da pesquisa, com o objetivo de oferecer um design mais claro e intuitivo para os usuários. O modal para remoção de pesquisadores de localidades foi reformulado, e a funcionalidade de remoção foi ampliada, permitindo que o usuário remova um ou mais pesquisadores, ou todos os pesquisadores associados a uma localidade específica da pesquisa.

Em complemento, foram implementados ajustes para vincular os pesquisadores ao responsável pela criação da pesquisa, garantindo que os dados apresentados sejam filtrados de acordo com essa associação. Essa melhoria reforça a consistência e organização no gerenciamento de informações relacionadas às localidades e pesquisadores.

Após a conclusão dessas tarefas, alguns modelos relacionados ao sistema foram aprimorados com a adição de um campo que permite associar diretamente as pesquisas criadas ao responsável por elas. Essa associação assegura que os dados vinculados às pesquisas sejam exibidos apenas para o usuário que as criou, proporcionando maior controle e segurança sobre as informações.

Anteriormente, o sistema permitia acesso irrestrito a todas as informações criadas, independentemente do cliente. Com essas alterações, o fluxo de informações passou a ser exclusivo para cada cliente, garantindo que apenas os dados criados ou associados por ele sejam acessíveis, promovendo uma gestão mais segura e eficiente.

4.6 Municípios

Nesta tarefa, foi corrigido o processo de criação da entidade "município"e implementado um *template* para listagem das entidades cadastradas no sistema. Anteriormente, era comum haver registros duplicados sem necessidade. Vale destacar que o *model* Entidade() se refere ao município no qual a pesquisa é realizada. Foi adicionado ao *model* Entidade() o campo CNPJ, utilizado para validar se uma entidade já está cadastrada no sistema.

Para facilitar a listagem, foi criado o *template* municipio_list_view.html, juntamente com a *class-based view* EntidadeListView(), que exibe todos os municípios cadastrados no sistema. O *modal* de cadastro de entidade, modal_entidade.html, passou por um *redesign*, tornando o processo de cadastro de novos municípios mais intuitivo e eficiente.

A validação do CNPJ no momento da criação de um novo município foi implementada através da *function view* verificar_cnpj_entidade(), acionada por uma função *Ajax*. Essa função recebe a resposta da validação e, caso o CNPJ ainda não esteja registrado, realiza uma requisição a uma API externa, obtendo os dados do município e da Unidade Federativa (UF). Esses dados são automaticamente preenchidos nos campos do formulário por meio de código JQuery e JavaScript, simplificando o processo de cadastro e assegurando a integridade das informações inseridas.

4.7 Setores

Nesta tarefa, foi realizado o *redesign* no *front-end* dos *templates* e modais de setores e localidades, além da implementação de novas funcionalidades. O *template* renderizado na *class* SetorListView() agora exibe apenas os setores pelos quais o usuário é responsável, a partir do município selecionado no campo *select* Município. E o *model* Setor() possui um novo campo responsável, o que permite ao usuário cadastrar um setor mesmo sem ter criado uma pesquisa,

por meio da nova *class-based view* SetorNovoCreateView(). Nesta nova *class*, diferentemente da criação de setor dentro de uma pesquisa, foi adicionado no *form* SetorNovoForm() o campo entidade, para associar o setor ao município.

O *modal* de edição de setor foi redesenhado para seguir o padrão visual do sistema. Além disso, foi implementada a funcionalidade de remoção de setor através da *class* SetorDeleteView(), que renderiza o novo template modal_setor_delete.html. Esta class também inclui uma validação, permitindo a remoção de um setor apenas se ele não possuir localidades associadas.

Em relação às localidades, o *redesign* foi aplicado ao *template* localidade_list_view.html, bem como nos modais de criação, edição e remoção de localidades. Na *class* LocalidadeDelete-View(), foi adicionada uma validação que impede a remoção de uma localidade caso ela esteja associada a uma pesquisa com status de "concluída"ou que possua entrevistas associadas.

4.8 Contratante

Nesta tarefa, foi iniciada a implementação de múltiplos contratantes associados a uma única pesquisa, permitindo que os clientes possam vincular mais de um contratante às suas pesquisas, o que antes não era possível. Para viabilizar essa mudança, foram feitas modificações no banco de dados, como a exclusão de campos e tabelas desnecessárias, e a criação de novas tabelas que suportam as novas funcionalidades.

Em seguida, foi necessário ajustar o código, excluindo o model relacionado a um contratante anterior e removendo todas as dependências associadas a ele. Após essas alterações, novas funcionalidades foram implementadas, com a criação de novos *models* e a codificação de diversas *views* e funções. Essas mudanças incluíram a criação, atualização, listagem e exclusão de contratantes, além de funções para verificar a existência de contratantes, vincular gestores aos contratantes, enviar senhas e validar se o login já está cadastrado.

4.9 Contratante x Pesquisa

Após a implementação da funcionalidade de contratantes, foi avançado para a associação dos contratantes a uma pesquisa, permitindo que o cliente do sistema gerencie seus contratantes diretamente dentro de cada pesquisa. O design do *template* de configuração da pesquisa foi ajustado para refletir essas mudanças.

Na parte de codificação, foram implementadas diversas funcionalidades nas *views* para validar as novas regras de negócio. Isso incluiu a criação de mecanismos para associar e desvincular contratantes de uma pesquisa, além de funções para obter dados do contratante e gerenciar o status de bloqueio ou desbloqueio dos contratantes dentro de uma pesquisa.

4.10 Assinatura

Antes de iniciar a tarefa, foi necessário refatorar uma função de validação de acesso do usuário à pesquisa, considerando a possibilidade de incluir contratantes. A função foi ajustada para garantir que a pesquisa seja acessada apenas pelo usuário devidamente associado a ela. Além disso, a lógica de acesso foi ampliada para incluir contratantes, que precisam ter assinado um termo de ciência e responsabilidade e não estar bloqueados pelo cliente responsável pela pesquisa.

O primeiro passo da tarefa foi um *redesign* no *front-end* do *template* de listagem de pesquisas. Quando um contratante visualizar a sua lista de pesquisas, ele verá o botão para assinar o termo de responsabilidade antes de acessar os botões de ação da pesquisa. Utilizando recursos do Django, como *function tags*, foi possível implementar a lógica para exibir os botões de forma dinâmica e condicional.

Para isso, foi criada mais uma *function tag* que verifica se o contratante já assinou o termo de responsabilidade. Também foi implementada uma função para exibir um *modal* com o termo, permitindo que o contratante assine. Após a assinatura, um código é enviado por e-mail para confirmação, e, após a validação, o contratante ganha acesso aos botões de ação. Por fim, foi adicionada uma funcionalidade para permitir que o contratante visualize e imprima o termo já assinado.

4.11 Pesquisadores

A primeira modificação realizada foi o *redesign* do *template* de listagem de pesquisadores, que agora inclui três novos botões para funcionalidades adicionais e altera a exibição dos pesquisadores. Dois modais foram criados para as funcionalidades de criação e edição de pesquisador. No *modal* de criação, foi implementada uma requisição *Ajax* para verificar, por meio de uma *function view*, se o login já está cadastrado no sistema.

Na criação de pesquisadores, foi adicionada a funcionalidade de gerar uma senha automaticamente. A senha é gerada por uma função específica e atribuída ao campo de senha do modelo Pesquisador(), enquanto o campo de login é preenchido com um valor concatenado para identificar a associação com o cliente do sistema. Além disso, no formulário de atualização de pesquisador, o campo de login foi ajustado para ser imutável, visto que é um campo único.

Foram também implementadas três novas funcionalidades: uma *function view* para gerar automaticamente até 15 pesquisadores, facilitando o processo de criação; outra para gerar novas senhas para todos os pesquisadores, aumentando a segurança; e, por fim, uma funcionalidade para listar os dados de acesso de todos os pesquisadores, permitindo copiá-los facilmente, o que agiliza o envio das informações para os pesquisadores.

4.12 Listagem das Pesquisas

Nesta tarefa, foi realizado o *redesign* do *template* de listagem de pesquisas. Essa tela é renderizada tanto para o Super Usuário quanto para o Cliente, com a adição de um botão visível apenas para o Cliente. Esse botão, denominado "Ativar/Desativar Suporte"aciona a *function view* pesquisa_suporte_permissao(), que permite ao suporte (representado pelo Super Usuário) acessar as ações de configuração da pesquisa. Dessa forma, o Super Usuário pode auxiliar o Cliente na correção de dados e informações da pesquisa.

5 PROCEDIMENTOS METODOLÓGICOS

Tendo em vista que este trabalho possui com objetivo realizar uma revisão bibliográfica sobre os conceitos de engenharia de software, metodologias ágeis e tecnologias que sustentam o Sispesquisa, afim de realizar um estudo sobre esses conceito e aplicá-los no desenvolvimento da aprimoração do sistema web SisPesquisa para endereçar o problema do contexto do trabalho, as atividades realizadas durante o trabalho de conclusão do curso foram:

- **Revisão Bibliográfica:** Está fase é dedicada à apresentar os conceitos teóricos essencias para embasar o desenvolvimento de software, como engenharia de software, evolução de software, metodologias ágeis (especialmente Kanban) e as prinicipais tecnologias nas quais o SisPesquisa é desenvolvido;
- **Análise da Arquitetura do Sistema Legado (SisPesquisa):** Etapa que visou compreender cada parte da arquitetura, identificando, no *Model*, *Templates* e *View*, os pontos que necessitam de evolução e melhorias, possibilitando a projeção eficiente das alterações necessárias;
- **Desenvolvimento e Implementação das Melhorias no SisPesquisa:** Realização do desenvolvimento da aprimoração no SisPesquisa, que envolveu a execução de cada atividade planejada no KanbanTool, utilizando tecnologias como Django, HTML, CSS, Bootstrap, JavaScript/jQuery e SQL, além das práticas do método ágil Kanban;
- **Resultados e Discussões:** Analisar o aprimoramento do sistema, debater sobre os resultados e concluir o trabalho.

5.1 Desafios na Aprimoração do SisPesquisa

Devido ao fato de o sistema ter sido construído praticamente sem documentação e, por vezes, conter códigos complexos, o desenvolvimento nem sempre foi satisfatório de realizar. Por exemplo, o template de listagem de pesquisas precisou ser modificado váriais vezes, uma vez que todos os usuários do sistema têm acesso a essa tela e diversas informações devem ser tratadas de maneira diferenciada para cada tipo de usuário. Esse fator depende dos desenvolvedores, que devem evitar maiores complexidades. Como afirma Martin (2009), os programadores são autores e responsáveis por comunicar-se de forma clara com seus leitores por meio das linhas de código, que serão julgadas por aqueles que as leem.

Para o autor deste trabalho, o início do desenvolvimento da aplicação foi bastante desafiador, pela a pouca experiência com as tecnologias e o tempo no ambiente de desenvolvimento de software. No entanto, conforme ressalta Beck (1999), "No desenvolvimento de software, 'perfeito' é um verbo, não um adjetivo. Não existe processo perfeito. Não existe design perfeito. Não existem histórias perfeitas. Você pode, no entanto, aperfeiçoar seu processo, seu design e suas histórias".

Outra questão foi que o autor deste trabalho não conseguiu apresentar artefatos que representassem o sistema antes das modificações realizadas, devido à atualização simultânea das versões do Django e do Python. Como o sistema está interligado a um projeto que inclui outros sistemas de natureza distinta, retroceder para versões anteriores demandaria um tempo significativo para ajustes e reconfigurações no ambiente. Esse processo comprometeria não apenas o cronograma do trabalho, mas também a estabilidade do ambiente compartilhado, tornando inviável a reversão às versões anteriores.

6 RESULTADOS E DISCUSSÕES

Este capítulo apresenta os resultados obtidos com a análise e planejamento do aprimoramento do SisPesquisa, abordando as limitações enfrentadas e as perspectivas para o desenvolvimento do sistema. A documentação das mudanças planejadas demonstra a importância de alinhar a evolução de sistemas legados às necessidades do mercado e dos usuários, enquanto se mantém o compromisso com a confidencialidade e as restrições legais envolvidas no projeto.

Dessa forma, algumas limitações foram enfrentadas durante o desenvolvimento do projeto, especialmente devido à assinatura de um Termo de Compromisso de Confidencialidade (Non-Disclosure Agreement - NDA), conforme está descrito no Anexo - A. Esse acordo restringiu a exposição de detalhes sobre as funcionalidades específicas e a lógica de negócio do SisPesquisa, limitando a profundidade das informações apresentadas.

6.1 Análise Comparativa do Sistema

- **SisPesquisa Antes da Aprimoração:** Antes das melhorias implementadas, o sistema SisPesquisa não contava com um controle rigoroso sobre o acesso e a segregação de dados, permitindo que os usuários tivessem acesso a todas as informações criadas, como perguntas, setores, localidades e pesquisadores. Além disso, o sistema apresentava dados redundantes, como municípios duplicados, o que gerava excesso de informações e dificultava a compreensão dos dados. Tanto o Super Usuário quanto o usuário comum (*User*) compartilhavam praticamente as mesmas funcionalidades, o que comprometia a clareza e o controle sobre as permissões. Já o Contratante possuía uma funcionalidade limitada, podendo ser associado a apenas uma pesquisa por vez e com acesso restrito aos relatórios das pesquisas;

- **SisPesquisa Após Aprimoração:** Com as alterações e novas funcionalidades implementadas, o sistema passou a contar com uma organização mais refinada dos módulos, proporcionando uma experiência mais segura e controlada para os usuários. Agora, o Cliente tem a possibilidade de ativar o suporte, permitindo que o Super Usuário auxilie diretamente nas configurações das pesquisas quando necessário. Adicionalmente, o Cliente pode gerenciar suas próprias pesquisas com maior autonomia, o que aprimora a usabilidade e a segurança. Para os Contratantes, o acesso aos dados e relatórios da pesquisa está condicionado à assinatura de um termo de confidencialidade, garantindo a proteção das informações. Ainda, uma pesquisa pode ter vários contratantes associados, ampliando a flexibilidade do sistema.

6.2 Impactos das Melhorias

Essas modificações trouxeram impactos expressivos no SisPesquisa, contribuindo para maior autonomia, suporte otimizado e segurança aprimorada. Primeiramente, com as novas

funcionalidades, os Clientes agora possuem maior autonomia e controle sobre suas pesquisas. Eles podem configurar e personalizar aspectos específicos de acordo com suas necessidades, como setores, localidades e pesquisadores, permitindo ajustes mais direcionados e independentes.

Outrossim, a introdução de um sistema de suporte otimizado pelo Super Usuário facilita intervenções diretas nas pesquisas, caso o Cliente necessite de assistência. Esse recurso proporciona um suporte ágil e efetivo, sem comprometer a segurança dos dados, garantindo que problemas sejam resolvidos rapidamente e que haja uma colaboração eficiente entre o Cliente e o Super Usuário.

Por último, as melhorias na segurança e organização dos dados implementaram novas regras de negócio que asseguram que apenas usuários autorizados tenham acesso a informações sensíveis. Essa estrutura fortalece a segurança do armazenamento e visualização de dados, proporcionando um ambiente mais confiável e organizado para todos os usuários do sistema.

7 TRABALHOS FUTUROS

Com o aprimoramento do SisPesquisa, o sistema está adaptado ao novo cenário de mercado, e oferecendo maior flexibilidade para futuras implementações. No entanto, para comprovar a eficácia das modificações realizadas, pretende-se conduzir uma pesquisa aplicada, utilizando testes de usabilidade e questionários direcionados a um grupo de possíveis usuários. Essa abordagem permitirá validar as melhorias implementadas e identificar oportunidades de refinamento do sistema.

7.1 Sugestões para Trabalhos Futuros

- **Testes de Usuários:** Aplicar questionários a um grupo de potenciais usuários do sistema para avaliar a percepção sobre as melhorias implementadas e identificar pontos de ajuste;
- **Testes de Usabilidade:** Conduzir testes estruturados para analisar a experiência do usuário durante a interação com o sistema, avaliando a eficiência, a eficácia e a satisfação;
- **Implementação de Perguntas Padrões:** Configurar o sistema para que, ao criar uma pesquisa, perguntas e alternativas padrões sejam automaticamente atribuídas, abrangendo contextos como sexo, faixa etária e escolaridade;
- **Perguntas de Múltipla Escolha:** Adicionar suporte a perguntas com múltiplas respostas, permitindo que os entrevistados selecionem mais de uma alternativa;
- **Novos Relatórios de Análise:** Desenvolver relatórios que integrem os dados das perguntas padrões e de múltipla escolha, fornecendo *insights* mais detalhados e abrangentes sobre os resultados das pesquisas.

REFERÊNCIAS

ALCHIN, M. **Pro Django**. 2. ed. [S.l.]: Apress, 2013. Citado na página 20.

ANDERSON, D. **Kanban: Successful Evolutionary Change for Your Technology Business**. Sequim, WA: Blue Hole Press, 2013. Citado na página 16.

BECK, K. **Extreme Programming Explained: Embrace Change**. Boston, MA: Addison-Wesley, 1999. Citado na página 32.

DUISEBEKOVA, K. *et al.* Django as secure web-framework in practice. *In*: **The Bulletin of Kazakh Academy of Transport and Communications**. [S.l.: s.n.], 2021. v. 116, n. 1, p. 275–281. Citado na página 21.

FILHO, W. **Engenharia de Software: produtos**. 4. ed. Rio de Janeiro: LTC, 2019. Citado na página 15.

MARTIN, R. C. **Clean Code: A Handbook of Agile Software Craftsmanship**. Upper Saddle River, NJ: Prentice Hall, 2009. Citado na página 32.

PRESSMAN, R.; MAXIM, B. **Engenharia de Software: uma abordagem profissional**. 9. ed. Porto Alegre: AMGH Editora, 2021. Citado 8 vezes nas páginas 11, 13, 14, 15, 16, 17, 18 e 19.

RUBIO, D. **Beginning Django**. [S.l.]: Apress, 2017. Citado na página 20.

SHYAM, A.; MUKESH, N. A django based educational resource sharing website: shreic. **Journal of Scientific Research, Institute of Science, Banaras Hindu University**, Varanasi, India, v. 64, n. 1, 2020. Citado na página 22.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011. Citado na página 13.

SOMMERVILLE, I. **Engenharia de Software**. 10. ed. São Paulo: Editora Pearson, 2019. Citado 7 vezes nas páginas 12, 13, 14, 15, 16, 18 e 19.

STROULIA, E. *et al.* Reverse engineering legacy interfaces: an interaction-driven approach. *In*: **Sixth Working Conference on Reverse Engineering (Cat. No.PR00303)**. Atlanta: IEEE, 1999. p. 292–302. Citado na página 13.

TOGNOZZI, B. **First Principles**. 2001. AskTOG. Disponível em: https://asktog.com/atc/principles-of-interaction-design/. Citado na página 19.

TRIPATHY, P.; NAIK, K. **Software Evolution and Maintenance: a practitioner's approach**. Hoboken, NJ: John Wiley & Sons, 2014. Citado na página 18.

ULRICH, W. M. The evolutionary growth of software reengineering and the decade ahead. **American Programmer**, v. 3, n. 10, p. 14–20, 1990. Citado na página 19.

VALENTE, M. **Engenharia de Software Moderna**. [S.l.]: Independente, 2020. Disponível em: https://engsoftmoderna.info/. Citado 2 vezes nas páginas 15 e 16.

VAMSI, K. M. *et al.* Visualization of real world enterprise data using python django framework. *In*: **IOP Conference Series: Materials Science and Engineering**. [S.l.: s.n.], 2021. v. 1042, p. 012019. Citado na página 21.

VINCENT, W. **Django for Beginners: Build Websites with Python & Django**. 4. ed. [S.l.]: WelcomeToCode, 2022. Citado 2 vezes nas páginas 20 e 22.

YU, Q.; YANG, W. The analysis and design of system of experimental consumables based on django and qr code. *In*: **2019 2nd International Conference on Safety Produce Informatization (IICSPI)**. Chongqing, China: [s.n.], 2019. p. 137–141. Citado na página 22.

YU, X. *et al.* Design and deployment of django-based housing information management system. **Journal of Physics: Conference Series**, v. 2425, p. 012018, 2023. Citado na página 21.

ANEXO A – TERMO DE DE COMPROMISSO DE CONFIDENCIALIDADE (*NON-DISCLOSURE AGREEMENT* - NDA)

TERMO DE COMPROMISSO DE CONFIDENCIALIDADE

Entre as partes:

João Lucas de Sousa Martins, inscrito(a) no CPF sob o nº **056.558.923-75**, regularmente matriculado(a) no curso de **Ciência da Computação**, da Universidade Estadual da Paraíba (UEPB), doravante denominado(a) **COMPROMISSADO(A)**; e
InsightGPC Technology Ltda, inscrita no CNPJ sob o nº **49.834.743/0001-56**, com sede em **Patos-PB, Rua Elias Asfora, 1195 , no bairro Maternidade, Edif. PrestContas 1° Andar**, doravante denominada **EMPRESA**,

Considerando que o(a) **COMPROMISSADO(A)** terá acesso a informações sensíveis e confidenciais da **EMPRESA** no contexto do desenvolvimento do seu **Trabalho de Conclusão de Curso (TCC)**, as partes firmam o presente **Termo de Compromisso de Confidencialidade**, também conhecido como **Non-Disclosure Agreement (NDA)**, mediante as seguintes cláusulas e condições:

CLÁUSULA 1 - DO OBJETO

O presente Termo de Confidencialidade tem por objetivo estabelecer as condições para o tratamento de informações classificadas como confidenciais e sensíveis da **EMPRESA**, a que o(a) **COMPROMISSADO(A)** terá acesso durante a realização do **TCC**, visando assegurar que tais informações não sejam divulgadas ou utilizadas indevidamente.

CLÁUSULA 2 - DA DEFINIÇÃO DE INFORMAÇÃO CONFIDENCIAL

Para efeitos deste Termo, considera-se como **Informação Confidencial** qualquer dado, documento, projeto, tecnologia, código fonte, diagramas da UML, DER, MER, prototipação, metodologia, plano de negócios, estratégias, relatórios, especificações técnicas, segredos de negócios, informações financeiras, comerciais, de mercado, fornecedores, parceiros, clientes, e qualquer outro material que tenha sido identificado como confidencial ou que, pelas circunstâncias de sua divulgação, seja entendido como tal.

2.1. Informações confidenciais incluem, sem limitação:

- Inovações tecnológicas e know-how da **EMPRESA**;
- Dados sobre processos internos, operações, clientes, e fornecedores;
- Código fonte;
- Diagramas UML, MER e DER;
- Prototipação;
- Infraestrutura de projeto;
- Resultados de pesquisas e desenvolvimentos realizados pela **EMPRESA** ou por terceiros a ela vinculados;
- Qualquer outro conteúdo classificado como confidencial pela **EMPRESA** ou protegido por direitos de propriedade intelectual.

CLÁUSULA 3 - DO COMPROMISSO DE SIGILO E CONFIDENCIALIDADE

O(a) **COMPROMISSADO(A)** compromete-se a:

3.1. **Manter em sigilo** absoluto todas as informações confidenciais a que tiver acesso durante o desenvolvimento do **TCC**, não as utilizando para nenhum outro propósito que não esteja diretamente relacionado ao objeto do seu estudo acadêmico;

3.2. **Abster-se de divulgar** informações confidenciais a terceiros, seja de forma verbal, escrita ou eletrônica, sem o prévio consentimento por escrito da **EMPRESA**;

3.3. **Adotar todas as medidas de segurança** necessárias para proteger as informações confidenciais contra uso não autorizado, cópia, ou divulgação;

3.4. **Devolver ou destruir**, conforme solicitação da **EMPRESA**, todas as informações confidenciais recebidas ou produzidas em decorrência das atividades do **TCC**, no prazo de até **48 horas** após o término do projeto ou mediante a rescisão deste Termo;

3.5. **TCC**, entregar uma copia a **EMPRESA** antes da apresentação do TCC para análise e aprovação do que pode ou não ser apresentado;

3.6. **Publicação**, não publicar em qualquer médio sem a autorização por escrita da **EMPRESA**

CLÁUSULA 4 - EXCEÇÕES À CONFIDENCIALIDADE

As obrigações de sigilo previstas neste Termo não se aplicam às informações que:

4.1. Já sejam de conhecimento público no momento de sua revelação, ou venham a se tornar públicas sem qualquer violação deste Termo;

4.2. Sejam obtidas por outras fontes de forma legítima e sem violação de qualquer obrigação de confidencialidade;

4.3. Sejam exigidas por autoridade judicial ou governamental competente, desde que o(a) **COMPROMISSADO(A)** notifique a **EMPRESA** com antecedência suficiente para que esta possa adotar medidas de proteção cabíveis.

CLÁUSULA 5 - DO USO DAS INFORMAÇÕES

O(a) **COMPROMISSADO(A)** utilizará as informações confidenciais exclusivamente para os fins do seu **TCC**, abstendo-se de empregar tais informações para fins comerciais, pessoais, ou outros que não estejam diretamente relacionados ao escopo do projeto.

CLÁUSULA 6 - DA RESPONSABILIDADE

O(a) **COMPROMISSADO(A)** será integralmente responsável por qualquer dano ou prejuízo resultante de sua conduta inadequada, negligente ou dolosa em relação à guarda e uso das informações confidenciais.

6.1. Caso ocorra qualquer violação deste Termo, a **EMPRESA** poderá adotar as medidas judiciais cabíveis, buscando a reparação de danos causados pela revelação indevida das informações confidenciais.

CLÁUSULA 7 - DA VIGÊNCIA E TÉRMINO

Este Termo de Confidencialidade entra em vigor a partir da data de sua assinatura e terá vigência até a conclusão do **TCC**. Entretanto, as obrigações de sigilo previstas neste Termo permanecerão em vigor por um período de **5 anos** após o término do projeto, conforme acordado entre as partes, independentemente do vínculo acadêmico ou profissional existente.

CLÁUSULA 8 - DAS PENALIDADES

O descumprimento de quaisquer das obrigações aqui estabelecidas sujeitará o(a) **COMPROMISSADO(A)** às penalidades cabíveis, incluindo:

8.1. **Rescisão imediata** do acesso às informações confidenciais e a outros recursos oferecidos pela **EMPRESA**;

8.2. **Indenização** por quaisquer danos diretos ou indiretos sofridos pela **EMPRESA** em decorrência da violação das cláusulas deste Termo.

CLÁUSULA 9 - DAS DISPOSIÇÕES GERAIS

9.1. Este Termo não gera nenhum vínculo empregatício, de parceria ou de sociedade entre as partes, limitando-se exclusivamente a regulamentar o uso de informações confidenciais durante o desenvolvimento do **TCC**.

9.2. Qualquer modificação deste Termo só será válida se feita por escrito e assinada por ambas as partes.

9.3. A eventual tolerância de uma das partes em relação à violação de qualquer cláusula deste Termo não implica renúncia de seus direitos.

CLÁUSULA 10 - DO FORO

Fica eleito o foro da comarca de **PATOS-PB** para dirimir quaisquer controvérsias decorrentes da interpretação ou execução deste Termo, com renúncia a qualquer outro, por mais privilegiado que seja.

Patos/PB, em 23 de Outubro de 2024.

João Lucas de Sousa Martins
CPF: 056.558.923-75

49.834.743/0001-56
Adenes Irineu Freire Junior
DIRETOR DE TI
INSIGHTGPC LTDA

ADENES IRINEU FREIRE JUNIOR
Cargo: DIRETOR DE TI
Empresa: INSIGHTGPC LTDA
CNPJ: 49.834.743/0001-56