UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Desenvolvimento de No-Code no FlowUp: Construção de campos personalizáveis

---

Relatório Técnico relativo ao Trabalho de Conclusão Curso do Bacharelado em Sistemas de Informação na modalidade Empresa

---

Aluno
João Victor Vieira Silva dos Santos

Orientador
Cleviton Vinicius Fonsêca Monteiro
BSI/UFRPE

Outubro de 2024

João Victor Vieira Silva dos Santos

Desenvolvimento de No-Code no FlowUp: Construção de campos personalizáveis

Relatório Técnico apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Cleviton Vinicius Fonsêca Monteiro

Recife

Outubro de 2024

Agradecimentos

Gostaria de expressar minha profunda gratidão a Deus, por me abençoar com força, saúde e perseverança, permitindo-me concluir esta jornada. É com imenso respeito e carinho que agradeço à minha noiva, Vitória, cujo apoio incondicional, incentivo constante e amor foram fundamentais para minha motivação e sucesso. Sua presença ao meu lado é um lembrete constante de que, juntos, podemos superar qualquer desafio.

A minha família e amigos, ofereço meu sincero agradecimento. Vocês foram minha rede de segurança, oferecendo apoio emocional, risadas e momentos de escape quando mais precisava. Cada um de vocês desempenhou um papel essencial, proporcionando o equilíbrio necessário para manter minha sanidade e foco ao longo desta jornada acadêmica.

Um agradecimento especial ao time FlowUp, por ser uma fonte de inspiração, conhecimento e orientação. A generosidade e o entusiasmo com que compartilharam suas experiências foram importante para o meu desenvolvimento pessoal e profissional.

Não posso deixar de expressar minha gratidão à Rural, por fornecer o ambiente acadêmico e os recursos necessários para o meu crescimento intelectual. Aos professores e colegas de curso, agradeço pela atmosfera de colaboração e pela troca de de ideias que enriqueceram minha experiência acadêmica.

Resumo

O presente trabalho tem como foco a implementação de funcionalidades No-Code no sistema ERP FlowUp com o objetivo de oferecer maior flexibilidade e personalização para os usuários. Utilizando a metodologia Design Thinking, foram desenvolvidos campos personalizados que permitem uma melhor adaptação do sistema às necessidades específicas de cada empresa. As entrevistas realizadas com stakeholders confirmaram a utilidade da funcionalidade, sugerindo melhorias para sua expansão. O estudo conclui que a introdução de No-Code no FlowUp é uma solução eficiente para personalização, oferecendo novas possibilidades de evolução e integração de dados.

Palavras-chave: No-Code, FlowUp, ERP, personalização, Design Thinking.

Abstract

This study focuses on the implementation of No-Code features in the FlowUp ERP system to provide greater flexibility and customization for users. Using the Design Thinking methodology, customizable fields were developed, allowing for better adaptation of the system to the specific needs of each company. Interviews with stakeholders confirmed the usefulness of the feature, suggesting improvements for its expansion. The study concludes that the introduction of No-Code in FlowUp is an efficient solution for customization, offering new possibilities for system evolution and data integration.

Key words: No-Code, FlowUp, ERP, customization, Design Thinking.

Sumário

1 Introdução

1.1 Apresentação

A era digital apresenta desafios crescentes em termos de eficiência operacional, flexibilidade e capacidade de inovação para as empresas. Nesse cenário, os sistemas de ERP (Planejamento de Recursos Empresariais) desempenham um papel importante ao integrar o gerenciamento de atividades diárias como contabilidade, gestão de projetos e gestão de capital humano, promovendo eficiência e uniformidade.

Contudo, os sistemas ERP trazem alguns desafios, embora sejam essenciais para a gestão empresarial integrada, a realidade é que cada organização possui requisitos únicos que os sistemas padrão muitas vezes não conseguem atender completamente, Souza (2000)[1]. Por isso, muitos sistemas ERP precisam estar em constante desenvolvimento para atender às necessidades específicas das empresas e acompanhar as demandas do mercado. Faz-se, então, necessário desenvolver a capacidade de personalizar e adaptar os sistemas ERP para atender a esses requisitos específicos.

Manter os sistemas ERPs atualizados pode ter um custo elevado e ser uma tarefa demorada, conforme aponta Quixy (2023)[2]. Uma das formas de promover ou garantir a atualização dos sistemas ERPs é o movimento Low-code/No-code (LC/NC), que propõe uma abordagem para construção ou melhoria de aplicativos e sistemas sem a necessidade de codificação. Rokis e Kirikova (2022)[3] descreve o movimento LC/NC como uma implementação que minimiza a codificação manual, usando interfaces gráficas e abstração visual, permitindo a participação de não-programadores. Além disso, Rokis e Kirikova (2022)[3] acrescentam que o LC/NC tem vantagens como a rápida tradução de requisitos de negócios para aplicativos e a facilidade de ajustes rápidos, sem a necessidade de uma codificação.

Segundo Sydle (2023)[4], é possível obter uma série de benefícios com o uso de LC/NC em sistemas ERP, sendo eles a redução de custos de desenvolvimento, a capacidade de configuração e customização, e uma maior produtividade e eficiência dos processos internos das empresas que utilizam esses sistemas. Diante disto e dos benefícios observados na literatura sobre LC/NC, e visando obtê-los em um sistema ERP chamado FlowUp, este trabalho propõe investigar como a implementação de funcionalidades No-Code no FlowUp pode contribuir para a agilidade no desenvolvimento do sistema e para a personalização e eficiência operacional das empresas que o utilizam.

Espera-se, portanto, que o FlowUp, ao adotar uma abordagem No-Code, possa contribuir para a redução dos custos e do tempo de desenvolvimento de funcionalidades adicionais, ao mesmo tempo em que oferece maior flexibilidade e capacidade de adaptação às mudanças no ambiente de negócios. Com a facilidade de personalização desta abordagem, as empresas podem ajustar seus processos internos mais rapidamente, sem depender exclusivamente de desenvolvedores de software para essas modificações. Esse processo simplificado de atualização pode aumentar a agilidade operacional e permitir que as organizações se ajustem a novos modelos de negócios e regulamentações.

Em resumo, o desenvolvimento de No-Code no FlowUp é uma oportunidade promissora para superar os desafios tradicionais no desenvolvimento de soluções ERP, incentivando a inovação e

aumentando a eficiência operacional.

1.2 FlowUp

O FlowUp constitui uma ferramenta de gestão empresarial desenvolvida pela empresa Fast, destinada a proporcionar o controle integrado e online de tarefas, produtividade e finanças. Conforme destaca Pereira (2021)[5], as funcionalidades centrais do FlowUp concentram-se na gestão financeira, gestão de projetos e controle de horas. Tais funcionalidades estão distribuídas em dois módulos que compõem o sistema: Task e Cash.

O módulo **Task**, representado na imagem 1, oferece suporte ao gerenciamento de projetos e atividades cotidianas dos usuários. Entre suas principais funcionalidades destacam-se:

- Gestão de Projetos: Permite o controle abrangente de projetos em andamento, com ferramentas para o acompanhamento do progresso, definição de metas e prazos.
- Quadros de Tarefas: Apresenta uma interface visual para a organização de tarefas, facilitando a visualização de atividades pendentes, em andamento e concluídas.
- Gestão de Tarefas: Facilita a criação, atribuição e acompanhamento de tarefas específicas, promovendo a divisão de trabalho e a colaboração entre os membros da equipe.
- Gestão de Pessoas: Gerencia os integrantes envolvidos em cada projeto, com funcionalidades de alocação de recursos e monitoramento de desempenho.
- Fluxos de Trabalho: Define e acompanha fluxos de trabalho automatizados, assegurando o cumprimento dos processos conforme planejado.

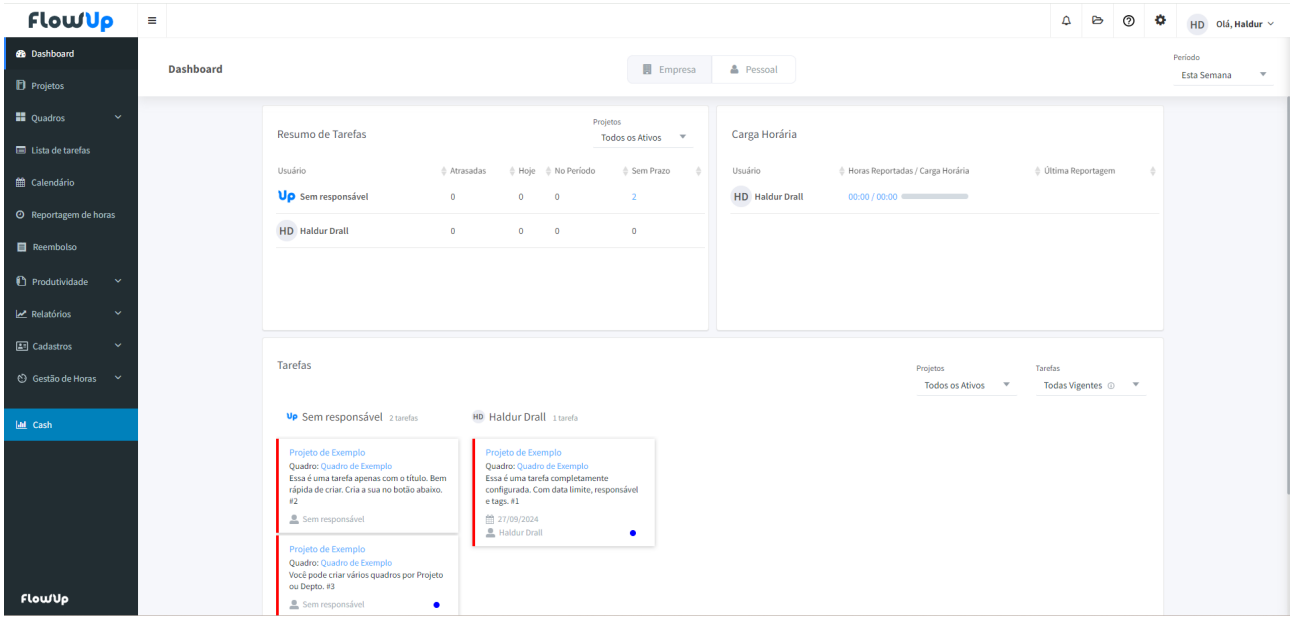

Figura 1: Tela de dashboard do Task.
Fonte: Própria (2024).

O módulo **Cash**, representado na imagem 2, por outro lado, é dedicado à gestão financeira da empresa, disponibilizando funcionalidades como:

- Gestão Financeira: Monitoramento centralizado de todas as movimentações financeiras.
- Tela de Extrato: Proporciona uma visualização de entradas e saídas financeiras, auxiliando no controle do fluxo de caixa.
- Gestão de Vendas de Serviços e Produtos: Gerencia a comercialização de serviços e produtos, com registro de transações.
- Cadastro de Serviços e Produtos: Guardando informações de serviços e produtos, facilitando os processos de vendas.
- Relatórios Contábeis: Gera relatórios financeiros e contábeis.

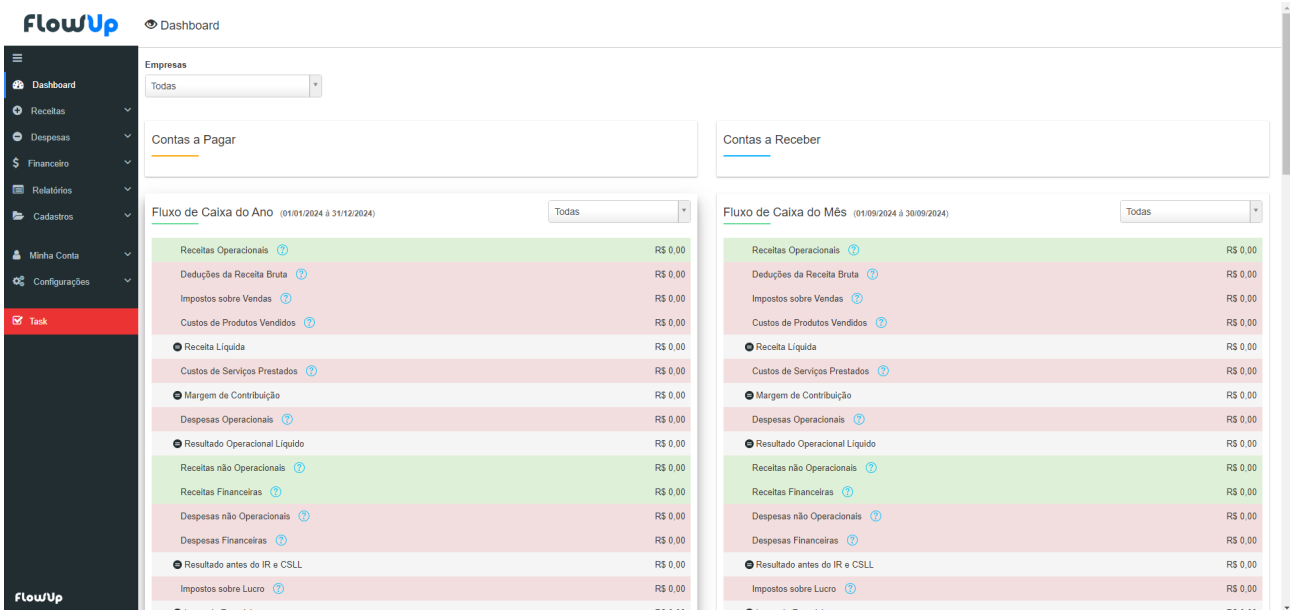

Figura 2: Tela dashboard do Cash.
Fonte: Própria (2024).

Apesar de o FlowUp oferecer uma grande variedade de funcionalidades, uma limitação significativa reside na impossibilidade de personalização dos campos disponíveis nas telas do sistema. Essa limitação impedem que os usuários ajustem os formulários conforme suas necessidades específicas. Por exemplo, a ausência de campos no cadastro de projetos como “Data de Aprovação da Planta”, “Responsável Técnico”, “Metragem Quadrada Construída”e “Orçamento da Obra”para empresas do setor de arquitetura e construção pode impedir o usuário de realizar análises mais precisas. Sem esses dados, não é possível, comparar o avanço físico da obra com o orçamento planejado, ou relacionar a performance de cada responsável técnico com o cumprimento de prazos e custos. Com a personalização, é possível obter uma análise mais completa das informações sobre o andamento da obra, relacionando custos, orçamentos e o progresso físico, garantindo um controle mais eficaz do projeto. Essa limitação na adequação do FlowUp pode dificultar a diferentes tipos de negócios e

fluxos de trabalho, comprometendo sua eficiência na resposta às demandas particulares de diversos setores empresariais.

Diante desses desafios, propõe-se a adição de uma funcionalidade *No-Code* ao FlowUp que permitirá a criação de campos personalizados, o que facilitará a adaptação do sistema às necessidades específicas de cada organização. A proposta se baseia nos problemas identificados no Design Thinking, onde as principais dificuldades enfrentadas pelos clientes foram analisadas. A introdução dessa funcionalidade visa não apenas ampliar a capacidade de adaptação, mas também oferecer uma solução mais adequada às demandas dinâmicas do mercado.

1.3 Objetivos

O objetivo geral deste trabalho é a implementação de funcionalidades No-Code no FlowUp, para que as empresas adequem o FlowUp às suas necessidades específicas e às demandas do mercado.

Assim para chegar no objetivo geral, foram divididos nesses objetivos específicos:

- Identificar quais são as principais necessidades e desafios das organizações em termos de eficiência operacional, flexibilidade e inovação, as quais podem ser atendidas através da implementação de funcionalidades No-Code no FlowUp.
- Analisar como a abordagem do No-Code pode ser integrada ao FlowUp, tendo em vista aspectos como a usabilidade, a personalização e a integração dos dados seguindo padrão de desenvolvimento FlowUp.
- Investigar os desafios e limitações associados à adoção do No-Code no FlowUp, e propor soluções para superá-los.
- Avaliar o impacto da adoção das funcionalidades No-Code no FlowUp a partir das percepções dos membros da equipe de desenvolvimento e gestão do FlowUp, identificando como essas funcionalidades contribuíram para melhorar a personalização e a eficiência das operações para os clientes.

2 Referencial Teórico

2.1 Sistema ERP

Os sistemas ERP (Enterprise Resource Planning) surgiram como uma resposta à crescente complexidade das operações empresariais e à necessidade de integração dos processos e funções. De acordo com Souza (2000)[1], esses sistemas começaram a ganhar relevância no final do século XX, quando as empresas reconheceram a importância de centralizar a gestão de áreas como produção, finanças e logística em uma única plataforma.

Conforme Matende e Ogao (2013)[6], os sistemas ERP modernizaram a gestão empresarial ao integrar as atividades corporativas em um único sistema, facilitando a tomada de decisões rápidas, redução de custos e maior controle gerencial. Além disso, esses sistemas promovem a coordenação

mais eficiente entre departamentos, resultando em respostas mais ágeis às demandas dos clientes e no aumento da eficiência operacional.

Nos últimos anos, os sistemas ERP continuaram a evoluir, integrando novos recursos tecnológicos que ampliam sua flexibilidade e capacidade de atender às demandas do mercado. De acordo com Martins e Belfo (2023)[7], os sistemas ERP modernos incluem inteligência artificial (IA) para automação de processos, análise de dados em tempo real, e integram o movimento low-code/no-code, que permite aos usuários personalizar e desenvolver soluções com mínima necessidade de programação. Esses novos recursos mitigam a necessidade de intervenções complexas, permitindo que empresas adaptem seus sistemas ERP às suas necessidades específicas de forma mais eficiente, melhorando a usabilidade e a capacidade de resposta das organizações.

A implementação de um ERP traz impactos em toda a organização, uma vez que os sistemas abrangem diversas funções empresariais, como recursos humanos, finanças, produção e logística. Segundo Matende e Ogao (2013)[6], os ERPs têm a capacidade de integrar as principais atividades de negócios de uma organização, garantindo a consistência e a confiabilidade dos dados compartilhados entre diferentes departamentos. Essa visão é complementada por Martins e Belfo (2023)[7], que destacam que a implementação de ERP simplifica e agiliza os processos empresariais, beneficiando a empresa por meio da automação e eficiência dos processos.

2.2 Low-Code e No-Code

A integração entre as plataformas LC/NC e suas funcionalidades, conforme apresentadas em diversos estudos, revela uma transformação significativa na maneira como não programadores e programadores abordam o desenvolvimento de sistemas. Como é notado por Rokis e Kirikova (2022)[3], a principal diferença entre Low-Code e No-Code está no nível de abstração da implementação do código. Enquanto o Low-Code requer que os usuários saibam o básico de lógica de programação, o No-Code abstrai completamente essa necessidade.

Essas plataformas, frequentemente oferecidas como PaaS, visam simplificar a construção de sistemas, que podem variar de simples a complexos, sem que os usuários precisem saber como escrever código. Um exemplo notável dessa aplicação é observado no uso de LC/NC em sistemas ERP, exemplificado pelo software Jira da empresa Atlassian, como pode ser observado na figura 3.

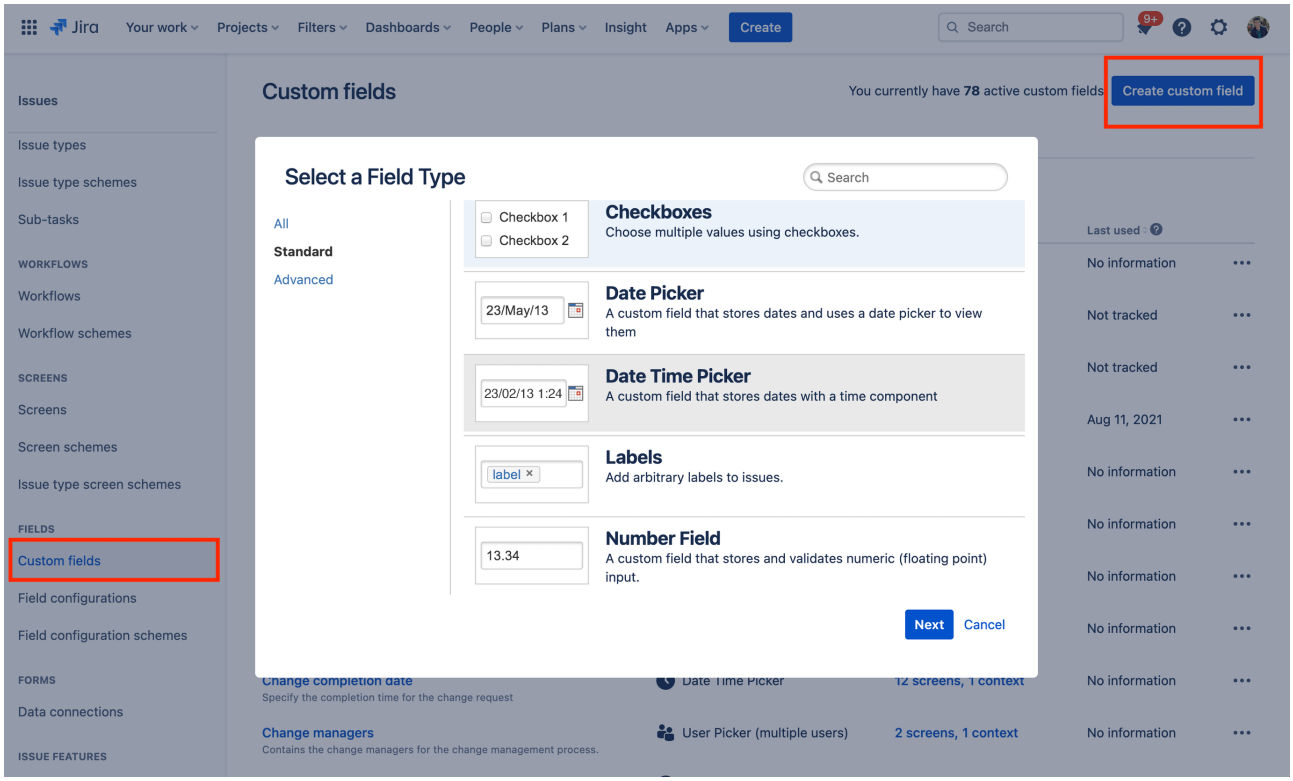

Figura 3: Figura da tela criação de campos personalizados do Jira
Fonte: Jira (2024).

Adicionalmente, Frank, Maier e Bock (2021)[8] classifica as funcionalidades LC/NC em distintas categorias que atendem a diferentes perfis de usuários e necessidades de desenvolvimento. Estas incluem:

- Plataformas básicas de gerenciamento de dados: essas plataformas oferecem principalmente recursos para visualização e edição de uma quantidade limitada de dados. As interfaces de usuário são projetadas segundo as necessidades do usuário, enfatizando uma gestão amigável das entidades. Simplificando o manuseio de dados para não-programadores.
- Sistemas de gerenciamento de fluxo de trabalho: o principal recurso desta categoria é o design visual dos fluxos de trabalho. Estes sistemas suportam a criação, execução e gerenciamento de processos de negócios, permitindo interações dinâmicas nos aplicativos. Muitas vezes incluem suporte adicional para integrações de terceiros, tornando-os adequados para aplicações que exigem fluxos de trabalho complexos.
- Ambiente de desenvolvimento integrado (IDEs) estendidos , centrados em Interface Gráfica do Usuário (GUI) e em dados: são caracterizadas como suporte para escrever e ajustar código-fonte na plataforma ou para integrar arquivos de código-fonte de fontes externas. Isso requer alguma familiaridade com linguagens de programação voltadas para usuários mais técnicos que precisam customizar suas aplicações.
- Plataformas multiúso para configuração, integração e desenvolvimento de aplicativos de negócios: são plataformas que combinam recursos das categorias anteriores com ênfase na integração e desenvolvimento de uma variedade de artefatos de aplicativos. Eles são projetados para

desenvolvimento e integração abrangentes de aplicativos de negócios, suportando artefatos de desenvolvimento internos e externos.

A funcionalidade de Campos Dinâmicos destaca-se principalmente nos Sistemas de Gerenciamento de Fluxo de Trabalho, evidenciando a importância do design e execução de processos interativos. Em contraste com plataformas que se concentram exclusivamente na gestão de dados, estas oferecem uma abordagem mais robusta e interativa para aplicações que requerem funcionalidades de grande complexidade.

2.3 Sistemas ERP com Funcionalidades Low-Code No-Code

Com a chegada da era digital, houve uma crescente demanda por agilidade e flexibilidade nos processos empresariais, o que levou ao surgimento do low-code e no-code como alternativas para a customização e atualização rápida de sistemas ERP.

Para Cigam (2021)[9], “o ERP low-code/no-code oferece uma abordagem estratégica para empresas que precisam de agilidade na implementação e manutenção de seus sistemas de gestão. Uma das principais vantagens do ERP low-code/no-code é a capacidade de personalização, permitindo que as empresas ajustem seus sistemas de acordo com suas necessidades específicas”. Da mesma forma, Sydle (2023)[4] destaca que “com a agilidade e eficiência se tornando cada vez mais obrigatórias para a transformação digital dos negócios, o desenvolvimento low-code surgiu para simplificar todo o processo de desenvolvimento, permitindo que aplicativos dinâmicos e modernos sejam criados de forma confiante, rápida e segura”.

Em resumo, a integração de funcionalidades low-code e no-code em sistemas ERP é uma tendência em ascensão, impulsionada pela necessidade de agilidade e flexibilidade no ambiente empresarial. Essas abordagens facilitam a atualização das funcionalidades, reduzindo a necessidade de desenvolvedores especializados, resultando em uma redução significativa de custos, ao mesmo tempo, possibilitando que as empresas respondam rapidamente às mudanças do mercado, reduzam custos e promovam uma maior eficiência operacional.

2.4 FlowUp: Tecnologias e arquitetura

Nesta seção, serão descritas as principais tecnologias utilizadas no desenvolvimento e a arquitetura de software adotada para o *FlowUp*. A primeira subseção aborda as ferramentas e *frameworks* que compõem a base tecnológica do sistema. A segunda subseção explora a arquitetura de software do *FlowUp*, destacando a adoção do padrão *Model-View-Controller* (MVC) e seus benefícios para a organização e manutenção do sistema.

2.4.1 Tecnologias

O projeto *FlowUp* foi elaborado a partir de uma variedade de tecnologias para atender às demandas de desenvolvimento de software moderno, assegurando a eficiência, a segurança e a escalabilidade. O projeto foi desenvolvido utilizando C#, uma linguagem de programação poderosa e

versátil, apoiada pela estrutura do *.NET* na versão 4.6.2, a qual fornece uma base sólida para a criação de aplicações robustas, com amplo suporte a bibliotecas. O uso do *Entity Framework* versão 6.1.3, um *framework ORM*, possibilita uma manipulação de dados mais ágil e segura, tornando o gerenciamento dos dados mais fácil entre o banco de dados *MySQL* e a aplicação.

Além disso, o projeto integra tecnologias *front-end* e de servidor, utilizando o *Microsoft.AspNet.Razor* versão 3.2.3 para criar visualizações HTML dinâmicas e interativas, melhorando a experiência do usuário. Esta escolha reflete a necessidade de uma interface de usuário ágil e receptiva, seguindo as boas práticas de desenvolvimento web mais recentes. O *FlowUp* utiliza o *Microsoft Visual Studio*, um *IDE* que oferece várias ferramentas de *debugging*, design de *UI* e gestão de banco de dados, tornando a execução e a avaliação da aplicação mais ágeis. O *FlowUp* utiliza a *AWS* para os seus serviços de hospedagem, repositório e banco de dados, assegurando uma infraestrutura de hospedagem escalável e confiável. A interação entre estas tecnologias permite que o *FlowUp* ofereça uma solução moderna e atualizada, atendendo às expectativas dos usuários e às exigências do mercado atual.

2.4.2 Arquitetura

No *FlowUp*, utiliza-se a arquitetura orientada a objetos baseada no MVC, que é uma abordagem de arquitetura de software. Valente (2020)[10] mostra que essa arquitetura é divida em três camadas, como apresentado na figura 4. A camada *view* (Visão) é responsável pela interface gráfica, outra é o *controller* (Controlador), cujo objetivo é lidar com eventos que interagem com o usuário, solicitando uma nova *view*, atualizando um status ou armazenando informações. E outra é o *model* (Modelo) é responsável por implementar as funções e os métodos necessários para gerir os dados do banco de dados. No *FlowUp* a camada do *model* é chamada de *services* (Serviços) e visa lidar também com as regras de negócios. Existe outro conceito relacionado a arquitetura do *FlowUp* que é chamado de *Entity*, ou Entidade, que indica uma tabela de banco de dados usada no *FlowUp*. Este conceito será aplicado para três propriedades da modelagem de dados que serão usadas para o desenvolvimento da funcionalidade, elas serão descritas no Apêndice [6].

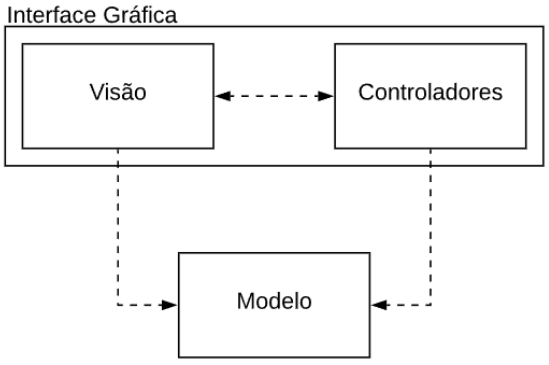

Figura 4: Figura ilustrativa da arquitetura Model-View-Controller.
Fonte: Valente (2020).

3 Metodologia

A solução apresentada está fundamentada nos conhecimentos adquiridos ao longo deste trabalho. O objetivo é fornecer uma compreensão clara e sistemática das abordagens adotadas, provendo visões sobre as decisões tomadas ao longo do projeto e a sua contribuição para a resolução do problema em questão.

3.1 Design Thinking

O Design Thinking (DT) é descrito por Vianna *et al.* (2012)[11] como uma metodologia voltada para a resolução de problemas de maneira criativa e inovadora, com foco central no usuário e na consideração de aspectos humanos durante o processo de solução. Originalmente, Vianna *et al.*[11] definem o DT como composto por quatro fases principais: Imersão, Análise e Síntese, Ideação e Prototipação, que não seguem uma sequência linear, mas constituem um ciclo contínuo de aprendizagem e aperfeiçoamento, onde os insights obtidos em uma etapa podem influenciar e modificar as demais. Palma (2022)[12] sugere a adição de uma quinta fase, a Implementação, que incorpora o teste e o desenvolvimento, integrando elementos de metodologias ágeis para aprimorar a construção e a validação de soluções. A figura 5 ilustra o clico do DT começando pela fase da Imersão.

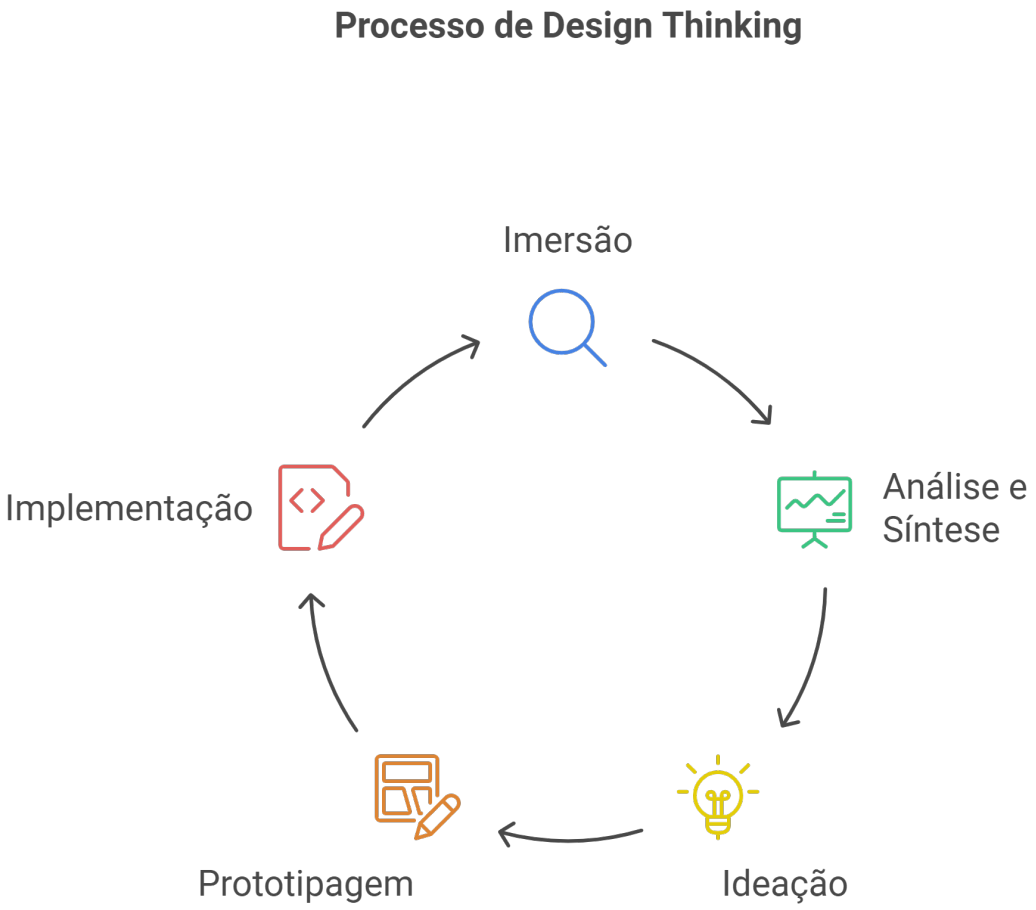

Figura 5: Figura ilustrativa do fluxo de Design Thinking.
Fonte: Própria (2024).

Descrição das Etapas do Design Thinking

- Imersão: Esta fase envolve a compreensão aprofundada do contexto do problema a ser resolvido, dividindo-se em Imersão Preliminar e Imersão em Profundidade. A Imersão Preliminar busca um entendimento inicial por meio de pesquisa bibliográfica, entrevistas com usuários e reuniões de alinhamento, enquanto a Imersão em Profundidade se foca na identificação de necessidades e oportunidades a partir das experiências e perspectivas dos usuários.
- Análise e Síntese: Nesta etapa, o objetivo é organizar e interpretar as informações coletadas nas fases anteriores para identificar padrões e oportunidades que orientarão a geração de soluções. Utilizam-se diversos artefatos, como mapas conceituais, personas e histórias de usuário, que auxiliam na definição do uso do software a partir da perspectiva do usuário final ou cliente.
- Ideação: A fase de Ideação é dedicada à geração de ideias através de atividades colaborativas, como brainstorming, bem como à revisão dos artefatos desenvolvidos para a formulação da solução. Essa etapa visa explorar diferentes soluções para o problema em questão, assegurando que elas estejam em conformidade com as necessidades identificadas dos usuários.
- Prototipação: Consiste na criação de protótipos da solução proposta, seguidos de testes e sugestões de melhorias para atender às necessidades identificadas nas etapas anteriores. Os protótipos oferecem uma oportunidade para testar e validar as propostas em ambientes reais ou simulados, facilitando a obtenção de feedbacks dos usuários ou clientes.
- Implementação: É a fase na qual ocorre o planejamento e a construção do software, integrando metodologias ágeis para promover o desenvolvimento incremental de serviços e funcionalidades. Esta etapa abrange o planejamento de requisitos funcionais e não funcionais, a definição de tecnologias, a arquitetura do software e a modelagem do banco de dados. Durante o desenvolvimento, realizam-se ciclos de sprints para garantir a entrega contínua de melhorias. A fase de testes assegura a validação das funcionalidades e tecnologias, enquanto a colaboração com usuários finais e stakeholders é essencial para alinhar as soluções às necessidades reais, promovendo um processo iterativo de refinamento e validação contínua da solução proposta.

3.2 Design Thinking no FlowUp

Nesta seção, será apresentado como foi aplicado o Design Thinking para o desenvolvimento deste projeto dentro do FlowUp.

3.2.1 Fase de Imersão

A fase inicial consistiu na condução de uma reunião com o Product Owner (PO), o Scrum Master (SM), desenvolvedores e designers do projeto FlowUp. O objetivo dessa reunião foi compreender em profundidade as necessidades e expectativas do FlowUp. Embora já exista um roadmap com as solicitações dos usuários e as melhorias planejadas para o sistema, o que facilita o trabalho dos

desenvolvedores ao direcionar as prioridades, essa reunião foi fundamental para identificar qual problema específico seria abordado no desenvolvimento do TCC. Assim, assegurou-se o alinhamento entre os objetivos do trabalho de conclusão de curso e as demandas reais do ambiente empresarial, garantindo que as funcionalidades desenvolvidas fossem relevantes e aplicáveis ao contexto do projeto.

3.2.2 Fase de Análise e Síntese

Com base nas informações coletadas durante a fase de imersão, realizou-se uma análise detalhada dos feedbacks e discussões obtidos, juntamente com um estudo sobre benchmarks de práticas recomendadas no desenvolvimento de software. Essa investigação permitiu identificar padrões, lacunas e oportunidades, que foram então traduzidos em user stories, que são descrições curtas e objetivas das funcionalidades desejadas pelos usuários finais.

3.2.3 Fase de Ideação

Na fase de Ideação, foi realizada a priorização das funcionalidades a serem prototipadas, com base nas user stories previamente desenvolvidas. Em seguida, ocorreu uma reunião com a equipe de desenvolvedores para definir a modelagem do banco de dados que integraria a solução, para garantir que a arquitetura de dados estivesse alinhada com os requisitos funcionais e as necessidades específicas identificadas nas etapas anteriores.

3.2.4 Fase de Prototipação

Na fase de Prototipação, foram criados os protótipos das interfaces de usuário seguindo o design padrão do FlowUp. Esses protótipos foram validados pelo PO e pela designer para garantir que atendesse aos requisitos estabelecidos.

3.2.5 Fase de Implementação

A fase de Implementação seguiu a metodologia ágil Scrum, que é utilizada de forma contínua no FlowUp para o desenvolvimento de novas funcionalidades, incluindo a criação de campos personalizados. No FlowUp, as sprints do Scrum possuem uma duração de 15 dias e são iniciadas com uma reunião de planejamento, na qual as tarefas são identificadas, priorizadas e atribuídas aos membros da equipe. As tarefas são organizadas em um quadro de tarefas que contém seis colunas, representando os diferentes estados de progresso: “Novas”, “Em andamento”, “Feitas”, “Em testes”, “Finalizadas” e “Implantadas”, como pode ser visto na figura 6.

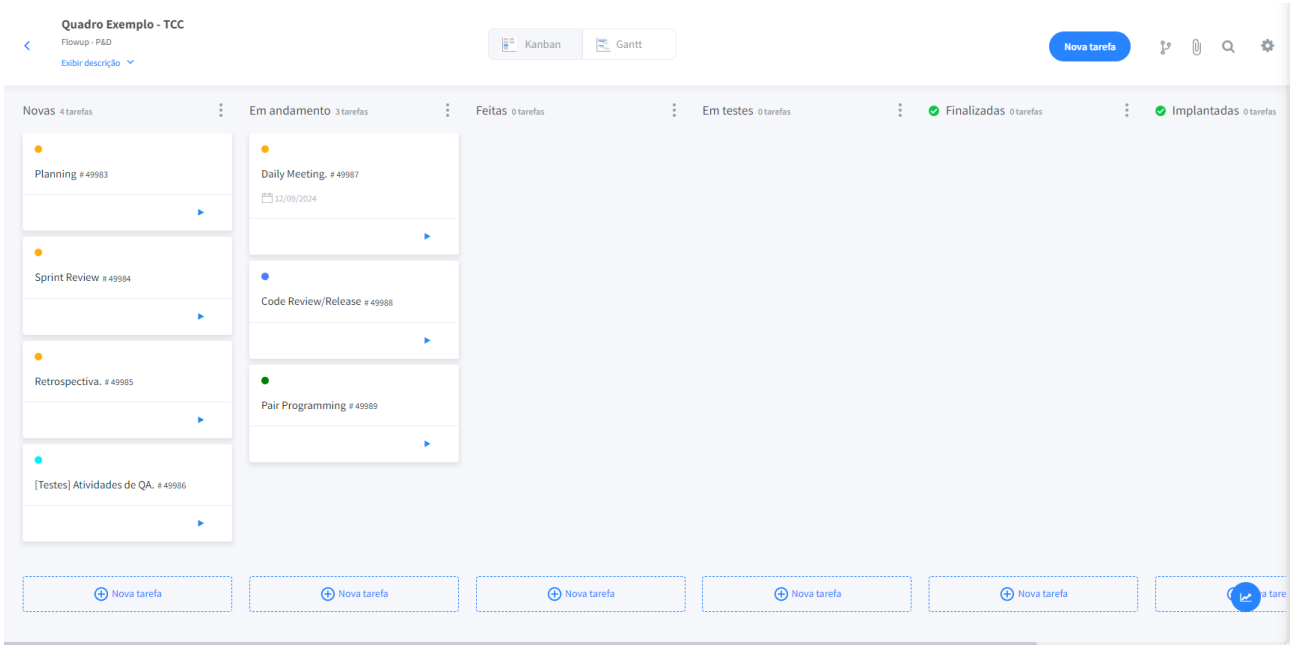

Figura 6: Modelo de quadro Kanban utilizado no FlowUp.
Fonte: Própria (2024).

Durante a sprint, à medida que as tarefas são concluídas e movidas para a coluna “Feitas”, elas passam por um processo de revisão de código e avaliação da funcionalidade, conduzido por um membro designado da equipe. Uma vez aprovadas, as tarefas são integradas ao ambiente de testes e transferidas para a coluna “Em Testes”, onde o Tester (QA) realiza a validação final de seu funcionamento. Caso a tarefa não atenda aos critérios de aceitação, ela é retornada para a coluna “Em andamento”, exigindo ajustes e correções pelo responsável. Se validada com sucesso, a tarefa é movida para “Finalizadas” e, subsequentemente, para “Implantadas”, tornando-se disponível no ambiente de produção.

Este processo de implantação é planejado para ocorrer ao final de cada sprint, garantindo a entrega contínua. No entanto, em casos de necessidades específicas, a implantação pode ser antecipada. Além disso, ao término de cada sprint, são realizadas duas reuniões essenciais: a Sprint Review, destinada a avaliar os resultados alcançados, adaptar o backlog conforme necessário, e a Retrospectiva, que visa analisar os processos e identificar oportunidades de melhoria contínua para o próximo ciclo.

3.3 Pesquisa Qualitativa e Entrevistas

A pesquisa qualitativa é uma metodologia amplamente utilizada para compreender fenômenos complexos em contextos naturais. De acordo com Merriam (2009)[13], essa abordagem foca na interpretação dos significados que os indivíduos atribuem às suas experiências, sendo ideal para estudos que buscam entender processos e contextos específicos. A pesquisa qualitativa adota um caráter indutivo, onde teorias são construídas a partir dos dados coletados, e não previamente testadas.

Segundo Merriam[13], a pesquisa qualitativa possui três características principais:

1. Foco no entendimento dos significados: A pesquisa qualitativa busca compreender como as pessoas interpretam suas vivências, em vez de se concentrar em medir quantidades ou testar relações.

2. Pesquisador como principal instrumento de coleta: O pesquisador desempenha um papel ativo no processo, adaptando-se ao contexto e interagindo diretamente com os participantes, o que permite uma coleta de dados mais profunda e sensível.

3. Processo indutivo: As interpretações emergem dos dados à medida que são coletados e analisados, sem hipóteses pré-determinadas, gerando uma descrição rica dos fenômenos investigados.

Para realizar essa pesquisa qualitativa, foram conduzidas entrevistas com o gestor de projeto, a gestora de sucesso do cliente e membros da equipe de sucesso do cliente, seguindo um roteiro estruturado. Durante as entrevistas, os participantes utilizaram a funcionalidade de campos personalizados, passando pelas etapas de cadastro e configuração do campo, edição e registro de uma resposta na tela de criação de projetos. Após essas ações, os entrevistados responderam ao questionário, que teve como objetivo avaliar a implementação dos campos personalizados no FlowUp e verificar se os requisitos mapeados foram atendidos. Essa abordagem permitiu uma análise detalhada da eficácia da funcionalidade, além de identificar possíveis áreas para melhoria.

A seguir, apresentamos as questões que compõem este instrumento de coleta de dados, as quais exploram a experiência geral dos usuários com os Campos Personalizados, além de aspectos específicos relacionados à usabilidade, impacto no fluxo de trabalho, e sugestões para aprimoramento:

1. Como você descreveria a experiência geral com a funcionalidade de Campos Personalizados?

2. Os Campos Personalizados atendem às necessidades e expectativas dos clientes?

3. Você encontrou alguma dificuldade para criar ou configurar os Campos Personalizados?

4. Quais aspectos poderiam ser melhorados para facilitar o uso?

5. O design da interface é intuitivo e fácil de usar?

6. Você notou algum problema de desempenho ou bugs ao usar os Campos Personalizados? Se sim, como isso afetou sua experiência?

7. Há algum outro comentário ou sugestão que você gostaria de compartilhar sobre os Campos Personalizados?

Por meio das respostas a essas questões e utilizando a técnica de *open coding* (Codificação aberta) para a analise das respostas, espera-se obter uma compreensão detalhada da funcionalidade e de sua influência no cotidiano dos usuários, contribuindo para a melhoria contínua do sistema ERP FlowUp. Afim de obter uma compreensão detalhada da funcionalidade e de sua influência no cotidiano dos usuários, foi utilizado a técnica de *open coding* (Codificação aberta) para a análise das respostas, assim, contribuindo para a melhoria contínua do sistema ERP FlowUp.

3.4 Contribuições

As contribuições do autor deste projeto foram fundamentadas nas fases do Design Thinking, conforme previamente descrito, abrangendo as etapas de pesquisa, prototipagem e desenvolvimento do backend. As demais atividades, incluindo o desenvolvimento do frontend e a integração com o Metabase, ficaram a cargo dos outros membros da equipe desenvolvimento. A fase de pesquisa envolveu uma revisão da literatura existente, bem como a realização de reuniões com os stakeholders. As informações obtidas nesse processo foram utilizadas para alinhar as necessidades dos usuários às funcionalidades esperadas. Com base nesses dados, foi desenvolvida a prototipação que foram validados pelo Product Owner e pela equipe de design.

Na etapa de desenvolvimento, o autor do projeto desempenhou um papel fundamental no backend, especificamente na implementação das regras de negócio, na criação e estruturação das tabelas do banco de dados e na definição dos seus respectivos relacionamentos. Embora o desenvolvimento do frontend e a integração com o Metabase tenham sido conduzidos por outros membros da equipe, todas as fases do processo foram feitas em colaboração com o PO.

4 Resultados

4.1 Necessidades

Na construção das user stories para o desenvolvimento deste projeto, foram apresentados apenas três critérios de aceitação para cada uma. Embora houvesse outros critérios relevantes, essa escolha foi feita para evitar que o texto se tornasse excessivamente longo e poluído, priorizando a clareza e objetividade. Assim, os critérios apresentados focam nos aspectos mais importantes para garantir que as funcionalidades atendam às principais necessidades do sistema. As user stories resultantes descrevem as funcionalidades-chave, com ênfase na personalização, flexibilidade dos formulários e integração dos dados com outras ferramentas, conforme identificado nas reuniões e pesquisas realizadas ao longo do projeto.

4.1.1 User Story 1: Campos Personalizados para Projetos

- **Como** um administrador do sistema,
- **eu quero** adicionar novos campos personalizados aos formulários de cadastro de projetos,
- **para que** eu possa registrar informações específicas que são relevantes para os projetos, como coordenadas, localização, data de finalização e tipo de projeto.

Critérios de Aceitação:

1. O sistema deve permitir a adição de novos campos aos formulários de projetos.
2. Os campos devem ser flexíveis, permitindo a captura de diferentes tipos de dados.
3. Os novos campos devem estar disponíveis em todos os cadastros de projetos.

4.1.2 User Story 2: Definir Campos Obrigatórios ou Opcionais

- **Como** um administrador do sistema,
- **eu quero** definir se um campo personalizado é obrigatório ou opcional,
- **para que** eu possa garantir que informações críticas sejam sempre coletadas durante o cadastro.

Critérios de Aceitação:

1. O sistema deve permitir marcar cada campo como “obrigatório” ou “opcional”.
2. Os campos obrigatórios devem impedir o envio do formulário caso não sejam preenchidos.
3. Os campos opcionais devem permitir o envio mesmo que estejam em branco.

4.1.3 User Story 3: Filtrar e Pesquisar Registros

- **Como** um usuário do sistema ERP,
- **eu quero** filtrar e pesquisar registros com base nos campos personalizados,
- **para que** eu possa encontrar facilmente as informações que preciso.

Critérios de Aceitação:

1. O sistema deve permitir a filtragem de registros com base em qualquer campo personalizado.
2. A pesquisa deve ser eficiente, exibindo apenas os registros que correspondam aos critérios definidos.
3. O usuário deve poder combinar múltiplos critérios de filtragem.

4.1.4 User Story 4: Validação de Campos Personalizados

- **Como** um administrador do sistema,
- **eu quero** configurar validações para os campos personalizados,
- **para que** eu possa garantir que os dados inseridos sejam precisos e estejam no formato correto.

Critérios de Aceitação:

1. O sistema deve permitir a definição de regras de validação para cada campo.
2. As validações devem incluir verificações de formato e campos numéricos ou de texto.
3. Campos que não atendam às validações devem exibir uma mensagem de erro clara.

4.1.5 User Story 5: Exportação de Dados

- **Como** um analista de dados,
- **eu quero** exportar dados que incluam campos personalizados,
- **para que** eu possa realizar análises detalhadas e gerar relatórios personalizados.

Critérios de Aceitação:

1. O sistema deve permitir exportar os dados em formatos compatíveis com ferramentas de análise.
2. Todos os campos personalizados devem ser incluídos na exportação.
3. O usuário deve poder selecionar os campos que deseja incluir na exportação.

4.1.6 User Story 6: Campos personalizados em Dashboards de BI

- **Como** um gerente de operações,
- **eu quero** usar campos personalizados em dashboards de BI,
- **para que** eu possa monitorar a eficiência operacional e identificar áreas de melhoria.

Critérios de Aceitação:

1. Os campos personalizados devem ser integrados aos relatórios do BI.
2. O dashboard deve permitir a visualização e análise dos dados com base nos campos personalizados.
3. O sistema deve gerar gráficos e métricas a partir dos dados filtrados pelos campos personalizados.

As *user stories* apresentadas permitiram priorizar as demandas, identificar possíveis desafios e garantir que a solução final estivesse alinhada aos objetivos do projeto, facilitando assim o desenvolvimento da funcionalidade dos campos personalizados..

4.2 Prototipação

As funcionalidades implementadas para a gestão de campos personalizados no FlowUp são apresentadas a seguir. Cada uma delas é ilustrada através de protótipos que mostram as interfaces desenvolvidas.

Uma das funcionalidades principais implementadas é a visualização dos campos personalizados adicionados ao sistema. O protótipo desenvolvido demonstra essa funcionalidade através de uma

tela que exibe os campos em formato de tabela, organizada de maneira a facilitar o gerenciamento das informações. A tabela contém três colunas principais: “Nome do Campo”, que identifica o campo; “Tipo do Campo”, que especifica o formato de dados associado; e “Obrigatoriedade”, que indica se o preenchimento do campo é obrigatório. Essa funcionalidade está representada na Imagem 7, nessa tela temos acesso a criação dos campos personalizados, que é um dos requistos para o User Story 1.

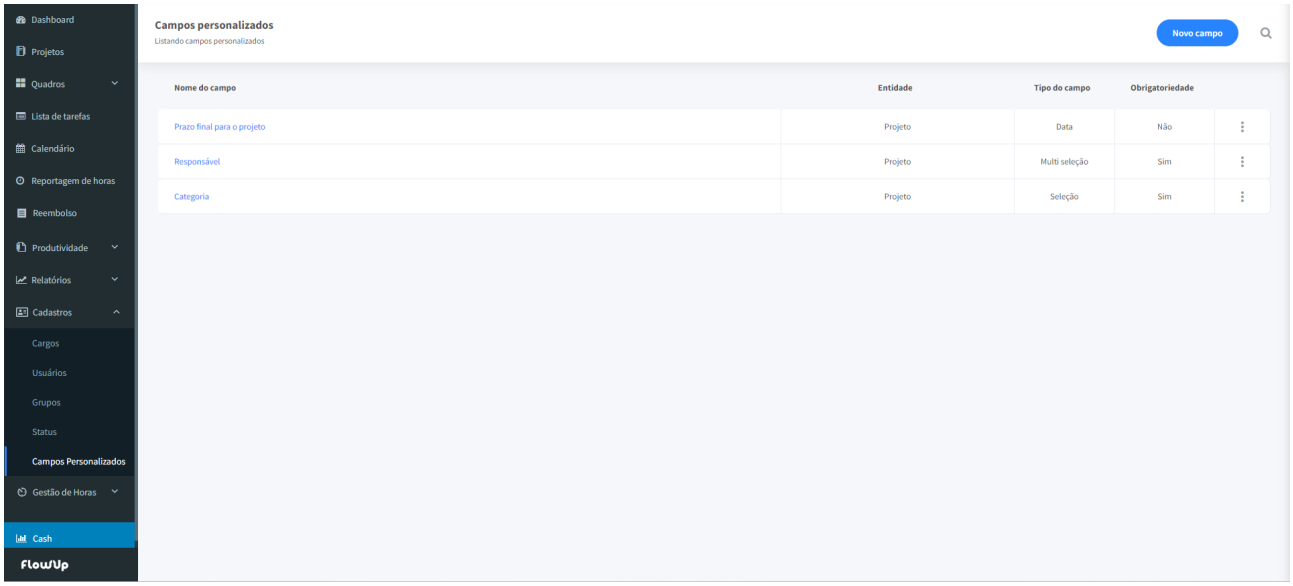

Figura 7: Listagem de campos personalizados criados.
Fonte: Própria (2024).

A imagem 8 representa o protótipo da funcionalidade de criar, que é necessário para satisfazer todos os requisitos da User Story 1. Esta tela composta por um formulário, onde o usuário pode definir detalhes como o “Nome” do campo, o “Tipo” de dado (texto, número, seleção, entre outros), o que é necéssario para a User Story 4, e a “Entidade” à qual o campo será aplicado, essa tela também contempla a User Story 2, com um checkbox que permite ao usuário determinar a obrigatoriedade do campo. Além disso, no caso de campos de seleção ou multisseleção, um botão adicional possibilita a configuração de valores de resposta predefinidos. Essa interface é projetada também para a edição de campos existentes, apresentando os dados previamente preenchidos no formulário para facilitar a atualização.

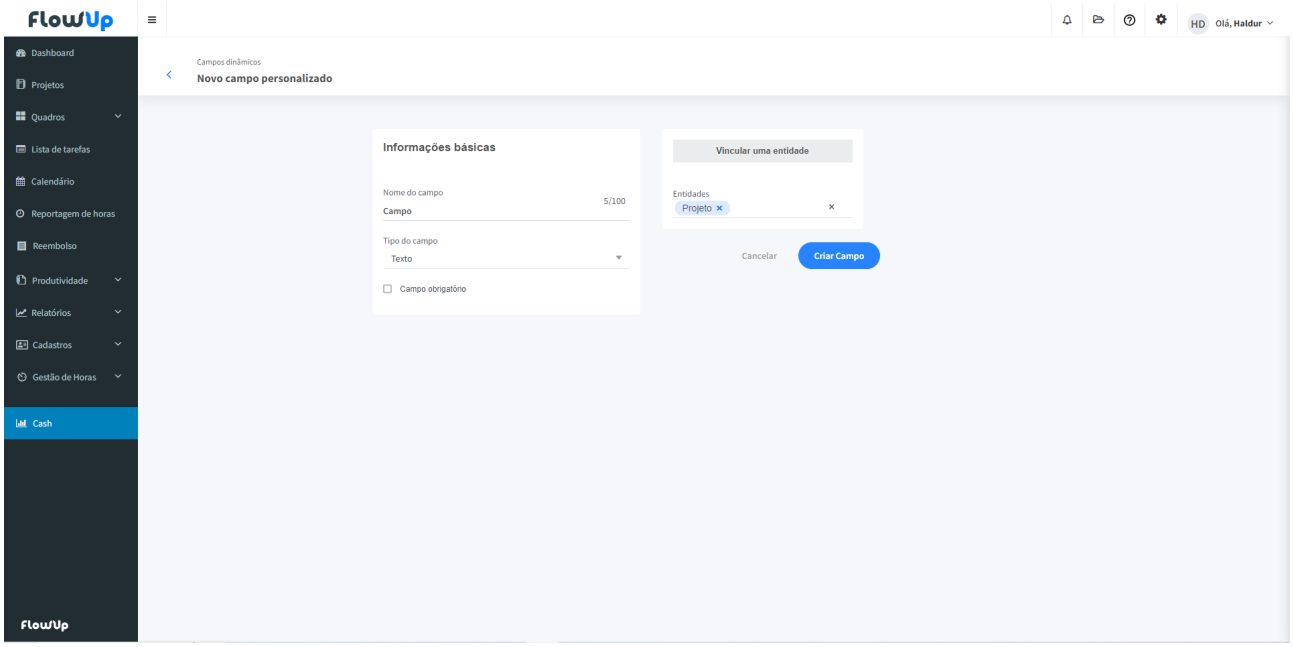

Figura 8: Cadastro de campo personalizado.
Fonte: Própria (2024).

Após a configuração dos campos personalizados, estes são integrados à tela de cadastro de projetos. O protótipo desenvolvido demonstra essa integração por meio de uma seção dedicada, intitulada “Outros Campos”, onde os campos previamente criados são exibidos para serem preenchidos pelos usuários, e ao preenchimento de cada campo é feito a validação de cada tipo de campo com sua respectiva configuração, assim satisfaz a User Story 4. A integração dos campos personalizados diretamente na tela de cadastro assegura que todas as informações essenciais estejam centralizadas, facilitando o gerenciamento dos dados de forma eficiente e personalizável. Essa funcionalidade, ilustrada na Imagem 9, é onde os campos configurados estão aplicados diretamente ao formulário de cadastro de projetos.

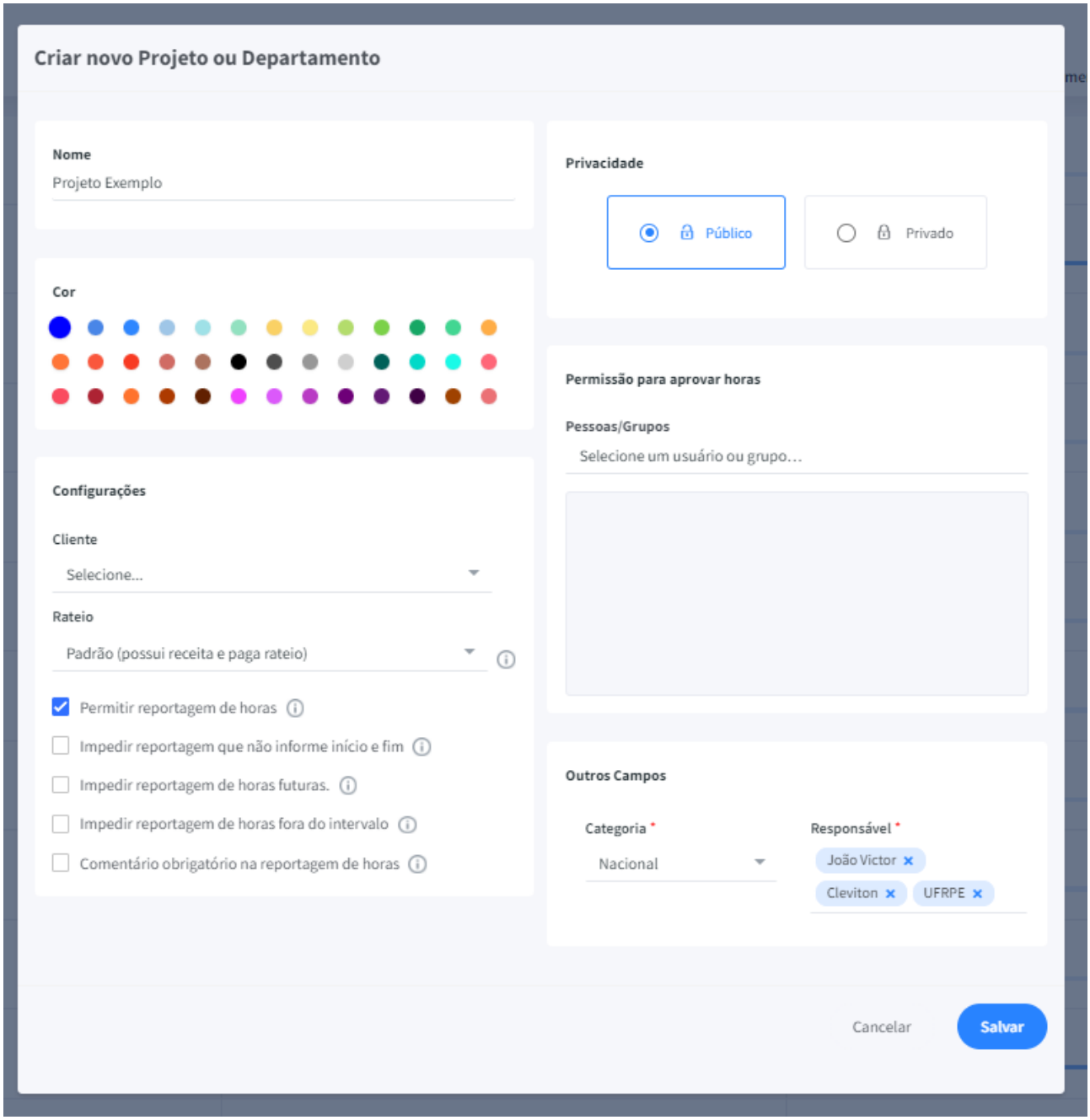

Figura 9: Cadastro de projeto com campos personalizados.
Fonte: Própria (2024).

Para consolidar as informações inseridas nos campos personalizados, foi inserido uma seção na tela de projetos, chamada “Outros campos”, permitindo a visualização dos dados já preenchidos. Essa funcionalidade facilita a consulta posterior e garante que os dados estejam acessíveis e bem organizados para acompanhamento. A exibição das respostas preenchidas nos campos personalizados é apresentada na Imagem 10.

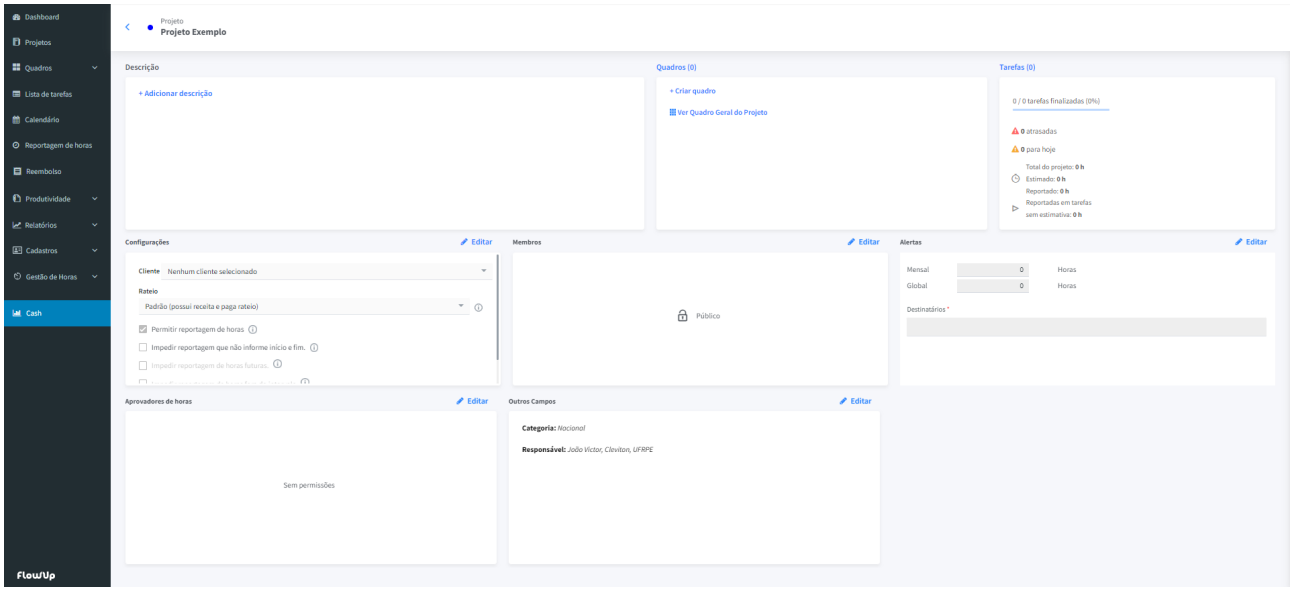

Figura 10: Tela de projeto com resposta dos campos personalizados.
Fonte: Própria (2024).

Por fim, O projeto também contemplou a integração com o Metabase, uma ferramenta de Business Intelligence (BI), utilizada para a criação de dashboards personalizados. A construção dos dashboards foi realizada diretamente no Metabase, através da montagem de consultas ao banco de dados da empresa. Isso permite aos usuários visualizar e analisar os dados de forma mais eficiente. Os campos personalizados do FlowUp foram incluídos nas consultas, oferecendo uma visão das informações específicas de cada projeto, conforme a necessidade do usuário, observe a coluna “Campo personalizado” na imagem 11.

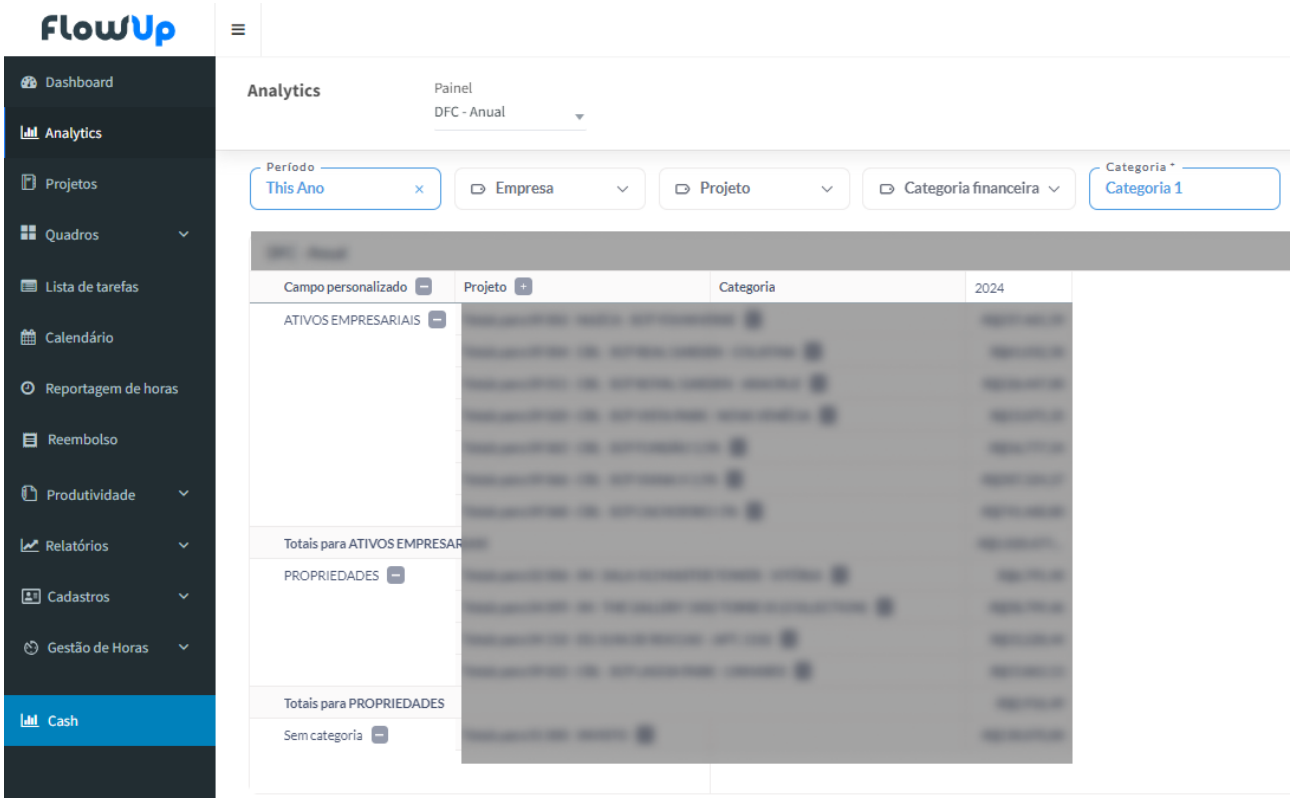

Figura 11: Tela com o componente do Metabase com os campos personalizados configurados.
Fonte: Própria (2024).

4.3 Desenvolvimento e Desafios

Durante o desenvolvimento do projeto, o objetivo central foi a criação de campos personalizados na entidade “Projetos” dentro do sistema FlowUp. Essa funcionalidade permite que os usuários adaptem os campos de acordo com as necessidades específicas de cada projeto, proporcionando flexibilidade na coleta e manipulação de dados. A decisão de começar com a entidade “Projetos” se deu pelo fato de ser uma classe central no fluxo de trabalho do sistema, sendo possível testar e ajustar o comportamento dos campos personalizados antes de expandir essa funcionalidade para outras entidades do sistema.

Para complementar a criação dos campos personalizados, além da integração com o metabase, foi melhorado a funcionalidade de exportação do extrato financeiro com os projetos, para incluir os campos , contemplando a User Story 5. Essa opção permite aos usuários extrair os dados completos para realizar análises externas ou gerar relatórios personalizados, oferecendo mais flexibilidade na utilização dos dados para finalidades específicas fora do ambiente do FlowUp.

Uma das principais dificuldades durante o desenvolvimento foi a escassez de literatura acadêmica especializada sobre a aplicação de tecnologias LC/NC em sistemas ERP. Embora existam revisões sobre LC/NC em um sentido mais geral, poucos estudos exploram sua implementação prática e desafios específicos no contexto de ERPs. Essa falta de material especializado dificultou a fundamentação teórica do projeto e limitou o acesso a referências que poderiam ter auxiliado em decisões técnicas e de design.

Outra dificuldade significativa esteve relacionada à coleta de feedbacks dos usuários. A funcionalidade foi inicialmente disponibilizada para um número limitado de empresas, sendo acessível apenas aos administradores do sistema. No entanto, esses administradores geralmente ocupam cargos de alta responsabilidade e possuem agendas bastante preenchidas, o que tornou difícil agendar entrevistas e reuniões para obter suas percepções detalhadas. Como alternativa, as entrevistas realizadas focaram nos membros da equipe FlowUp, conforme mencionado na seção dedicada aos questionários.

Além disso, a complexidade da implementação de um filtro único para os campos personalizados no FlowUp representou um obstáculo técnico significativo. A variedade dos tipos de dados nos campos tornou inviável a criação de um filtro dinâmico para todos os tipos de dados, o que pode limitar a flexibilidade do usuário na hora de manipular e buscar informações específicas dentro do sistema utilizandos os campos personalizados.

4.4 Análise das Entrevistas

As entrevistas realizadas com os membros da equipe FlowUp, que lidam diretamente com clientes, forneceram importantes percepções sobre a funcionalidade de “Campos Personalizados”. A análise das respostas revela uma visão consistente entre os entrevistados, destacando aspectos positivos e algumas sugestões de melhorias para o desenvolvimento futuro da ferramenta.

4.4.1 Experiência Geral com a Funcionalidade

De modo geral, os entrevistados descreveram a funcionalidade como positiva e intuitiva, enfatizando a facilidade de uso e a autonomia que ela proporciona para a criação de novos campos personalizados. Todos os participantes concordam que a ferramenta é fácil de implementar e auxilia na gestão dos projetos ao permitir a criação de campos específicos para diferentes necessidades. No entanto, uma crítica comum foi a limitação das entidades às quais os campos podem ser aplicados. Atualmente, eles estão restritos ao projeto, mas os usuários sugerem que essa funcionalidade poderia ser expandida para incluir quadros e tarefas, o que aumentaria sua flexibilidade e aplicabilidade.

4.4.2 Atendimento às Necessidades dos Clientes

Embora a funcionalidade seja considerada útil, houve consenso de que ainda não atende completamente às expectativas dos clientes. A principal queixa foi a restrição de uso dos campos personalizados a uma única entidade (projeto). Os clientes manifestaram interesse em aplicá-los também em outras áreas, como vendas, despesas, quadros e tarefas, além de aprimorar os filtros de pesquisa para permitir a busca por campos personalizados. Isso indica que, embora a ferramenta seja vista como promissora e expansível, ela ainda precisa de ajustes para se alinhar melhor às demandas práticas do dia a dia dos usuários.

4.4.3 Dificuldades de Criação ou Configuração

Nenhum dos entrevistados relatou dificuldades na criação ou configuração dos campos personalizados. Pelo contrário, todos enfatizaram que a usabilidade da ferramenta é intuitiva, fácil de entender e sem complexidades técnicas aparentes. Essa simplicidade de uso foi vista como uma das principais vantagens da funcionalidade, sugerindo que a curva de aprendizado é baixa e que a interface é adequada para os usuários atuais.

4.4.4 Aspectos a Melhorar

As melhorias mais sugeridas foram relacionadas à ampliação do escopo da funcionalidade. Além das limitações já mencionadas quanto às entidades, foi sugerido que a interface poderia ser mais visível em algumas áreas, como na aba de produtividade, o que facilitaria o acesso e o uso dos campos personalizados. No entanto, outros entrevistados não identificaram a necessidade de grandes mudanças, refletindo uma satisfação geral com a usabilidade atual da ferramenta.

4.4.5 Desempenho e Bugs

Todos os entrevistados relataram que não enfrentaram problemas de desempenho ou bugs ao utilizar os campos personalizados. Isso se deve ao fato de que, durante o processo de implementação, foram identificados e corrigidos bugs relacionados à interface visual, antes que a funcionalidade fosse disponibilizada no ambiente de produção. A correção desses problemas garantiu que, no momento do uso pelos administradores e demais usuários, a funcionalidade estivesse estável e bem implementada, proporcionando uma experiência fluida e sem interrupções técnicas.

4.4.6 Comentários Finais

Nos comentários finais, os entrevistados destacaram que a funcionalidade de campos personalizados possui grande potencial de expansão e pode ser uma ferramenta essencial para diversos setores, desde que mantida sua flexibilidade e autonomia para os usuários. A percepção é de que, com as melhorias sugeridas, a funcionalidade poderia se tornar ainda mais versátil, atendendo a um público mais amplo e diversificado.

5 Impactos da formação no trabalho

A formação acadêmica do autor deste projeto desempenhou um papel fundamental no desenvolvimento das minhas atividades na empresa. As habilidades e conhecimentos adquiridos em disciplinas como Algoritmos e Programação com *Python* Orientado a Objeto forneceram uma base sólida para resolver problemas de software. A capacidade de pensar logicamente e estruturar códigos de forma clara e eficaz, adquirida nessas disciplinas, tem sido essencial na criação e manutenção de

projetos. Além disso, a experiência com AWS e Introdução ao armazenamento de dados adquirida, auxiliou na compreensão sobre a infraestrutura de nuvem e gerenciamento de dados.

Finalmente, a experiência adquirida em projetos de disciplinas, projetos de extensão como *Seed a Bit* e disciplinas de modelagem e teoria da computação, proporcionou uma visão abrangente e uma abordagem multidisciplinar para a resolução de problemas, o que tem sido fundamental para o sucesso dos projetos em que estou envolvido na empresa. Além disso, a colaboração dos projetos e a participação na *Seed a Bit*, permitiu desenvolver habilidades como comunicação e trabalho em equipe. Essas experiências permitiram-me contribuir significativamente para o crescimento da empresa.

6 Conclusão

O trabalho desenvolvido focou na implementação de funcionalidades No-Code no sistema FlowUp, com o objetivo de aumentar a flexibilidade e a personalização para as empresas que o utilizam. A partir do uso da metodologia Design Thinking, foi possível entender principais as necessidades dos usuários, resultando na criação de campos personalizados que permitem uma adaptação mais eficiente do sistema às demandas específicas de cada organização. Este recurso, além de reduzir a dependência de desenvolvedores, promove uma maior autonomia dos usuários, alinhando-se às tendências atuais de personalização e automação nos sistemas ERP.

As entrevistas realizadas durante o processo reforçaram o valor da funcionalidade implementada, demonstrando que os campos personalizados atenderam às expectativas de uso e simplificaram o fluxo de trabalho. No entanto, o estudo também revelou que a funcionalidade poderia ser expandida para outras entidades além dos projetos, como quadros e tarefas, o que agregaria ainda mais valor ao sistema.

Para trabalhos futuros, é importante considerar a expansão da funcionalidade para outras entidades, como foi observado nas entrevista, mas também o uso de outras entidades como respostas em campos de seleção, o que traria uma maior integração à ferramenta. Além disso, encontrar soluções para as limitações nos filtros baseados em campos personalizados pode melhorar ainda mais a eficiência do sistema. Por fim, realizar estudos de caso em empresas que utilizam o FlowUp poderia oferecer insights valiosos para aprimorar a experiência dos usuários e validar o impacto dessas funcionalidades na otimização dos processos empresariais.

Referências Bibliográficas

[1] C. A. de Souza, “Sistemas integrados de gestão empresarial: estudos de casos de implementação de sistemas erp,” *Dissertação (Mestrado em Métodos Quantitativos) - Faculdade de Economia, Administração e Contabilidade, Universidade de São Paulo, São Paulo*, vol. doi:10.11606/D.12.2000.tde-19012002-123639, 2000, acessado em: 2024-02-25.

[2] Quixy, “Boosting your traditional erp with no-code low-code: The essential upgrade,” Disponível em:<https://quixy.com/blog/traditional-erp-with-no-code-low-code-erp-modernization/#1-ease-of-customization-and-maintenance>, 2023, acessado em: 2024-02-28.

[3] K. Rokis and M. Kirikova, “Challenges of low-code/no-code software development: A literature review,” *In: Nazaruka, Ē., Sandkuhl, K., Seigerroth, U. (eds) Perspectives in Business Informatics Research. BIR 2022. Lecture Notes in Business Information Processing*, vol. 462. Springer, Cham., 2022.

[4] Sydle, “Low-code erp: How it works and the advantages of using it,” Disponível em:<https://www.sydle.com/blog/low-code-erp-639c6fabe3d59040cdf94ece>, 2023, acessado em: 2024-02-28.

[5] J. R. Pereira, “Ferramenta para otimizar a importação dos dados de novos clientes do sistema flowup,” *Universidade Federal Rural de Pernambuco – UFRPE Departamento de Estatística e Informática Curso de Bacharelado em Sistemas de Informação*, 2021.

[6] S. Matende and P. Ogao, “Enterprise resource planning (erp) system implementation: A case for user participation,” *Procedia Technology*, vol. 9, pp. 518–526, 2013, cENTERIS 2013 - Conference on ENTERprise Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ HCIST 2013 - International Conference on Health and Social Care Information Systems and Technologies. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212017313002120

[7] E. J. Martins and F. P. Belfo, “Major concerns about enterprise resource planning (erp) systems: A systematic review of a decade of research (2011-2021),” *Procedia Computer Science*, vol. 219, pp. 378–387, 2023, cENTERIS – International Conference on ENTERprise Information Systems / ProjMAN – International Conference on Project MANagement / HCist – International Conference on Health and Social Care Information Systems and Technologies 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050923003125

[8] U. Frank, P. Maier, and A. Bock, “Low code platforms: Promises, concepts and prospects. a comparative study of ten systems,” University Duisburg-Essen, Institute for Computer Science and Business Information Systems(ICB), Essen, ICB-Research Report 70, 2021, urn:nbn:de:hbz:464-20211228-082939-0. [Online]. Available: http://hdl.handle.net/10419/248826

[9] Cigam, “Tecnologia low code,” Disponível em:<https://www.cigam.com.br/blog/384/o-que-e-erp-low-code>, 2021, acessado em: 2024-03-15.

[10] M. T. Valente, *Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade*. Editora: Independente, 2020.

[11] M. Vianna, Y. Vianna, I. K. Adler, B. Lucena, and B. Russo, *DesignThinking: Inovação em negócios*. Rio de Janeiro: MJV Press, 2012.

[12] J. G. Palma, R. T. de Araújo, and J. A. Souza, “Uma abordagem de design thinking no desenvolvimento de software,” *Conjecturas*, vol. 22, no. 5, pp. 1–15, 2022.

[13] S. B. Merriam, *Qualitative Research: A Guide to Design and Implementation*, 3rd ed. San Francisco, CA: Jossey-Bass, 2009.

Apêndice A

Tabelas de dados

O modelo de banco de dados, ilustrado na imagem 12, foi projetado para generalizar o uso de campos personalizados no FlowUp, permitindo que diferentes entidades ter novos campos sem a necessidade de modificar diretamente as tabelas principais.

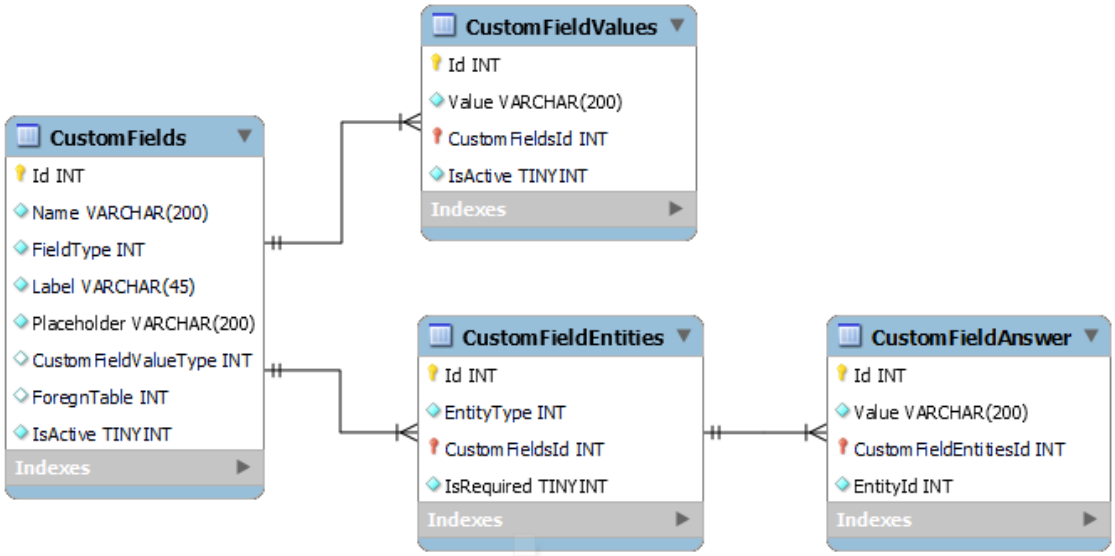

Figura 12: Modelagem de banco de dados dos campos personalizados .
Própria.

A tabela CustomFields armazena a definição dos campos personalizados, como nome, tipo, label, e se o campo está ativo. Esses campos podem ser reutilizados em várias entidades, e é através da tabela CustomFieldEntities que ocorre o vínculo entre os campos e as entidades específicas do sistema (como usuários, produtos, pedidos, etc.). Dessa forma, cada campo pode ser associado a qualquer entidade, o que oferece flexibilidade e escalabilidade para a aplicação dos campos em diferentes partes do sistema.

Para campos que envolvem seleção ou multisseleção, a tabela CustomFieldValues guarda os valores possíveis. Isso permite que, ao criar um campo de seleção, os valores predefinidos sejam armazenados e associados ao campo correspondente, facilitando a reutilização e manutenção das opções de escolha.

Por fim, a tabela CustomFieldAnswer é onde os valores preenchidos pelos usuários são armazenados. Ela faz a ligação entre a entidade que responde, o campo específico, e o valor inserido, garantindo que as respostas sejam corretamente associadas à entidade e ao campo correspondente.

Essa modelagem possibilita uma abordagem flexível e eficiente para o uso de campos perso-

nalizados no FlowUp, facilitando a escalabilidade e manutenção, sem necessidade de constantes alterações no esquema principal do banco de dados.

A seguir será descrito as colunas de cada tabela. A sua representação será pelo nome da coluna em negrito e seu tipo em seguida em parênteses, logo depois uma descrição do uso da coluna da tabela.

CustomFields

A *CustomFields*, ela é responsável por armazenar os valores principais que definem o tipo de campo personalizado, assim como outras propriedades de customização. Cada coluna representa:

- **Id** (*Int*): Identificador chave de cada campo personalizado, sendo Id único e autoincremento.
- ***Name*** (*String*): Nome do campo personalizado, utilizado para filtrar os campos na tela que listará os campos personalizados criados pelos usuários
- ***FieldType*** (*Enum*): o tipo de campo personalizados, estes foram mapeados podendo ser campo de texto, seleção, multipla seleção, checbox, switch, data, número e booleano.
- ***Label*** (*String*): a Label é a informação que especifica a qual input se refere e também ajuda na experiencia do usuário.
- ***Placeholder*** (*String*): o Placeholder é um texto auxiliar que ajuda ao usuário identificar com o que o input precisa ser preenchido.
- ***CustomFieldValueType*** (*Enum*): este atributo somente para campos numéricos, representa o qual unidade numérica o campo se refere. Exemplo: porcentagem, dinheiro, número e etc..
- ***ForeignTable*** (*Enum*): este atributo poderá ser utilizado para campos do tipo seleção ou seleção múltipla. E armazenará a entidade cujo qual será as opções destes tipos de campos.
- ***IsActive*** (*Bool*): este atributo foi adicionado para que a exclusão do campo seja somente lógica, ou seja, está variável apenas indicara se o campo aparecerá ou não na tela.

CustomFieldEntities

A *CustomFieldEntities*, esta tabela é responsável pelo vinculo de um campo personalizado com uma ou mais entidades. Assim, cada coluna da tabela é representada da seguinte forma:

- **Id** (*Int*): Identificador chave único e autoincremento.
- ***EntityType*** (*Enum*): Representa qual tipo de entidade deverá mostra o campo personalizado. Apesar de existirem diversas entidade no sistema, foram mapeados algumas, levando em consideração algumas demandas do cliente e outras entidades que a equipe de desenvolvimento considerou relevante, sendo elas a entidade Projeto, Quadro, Tarefa, Usuário, Venda de Serviço, Venda de Produto, Receita, Despesa e Clientes.

- ***CustomFieldId*** (*Int*): Este atributo foi criado para relacionar um campo com uma entidade, ou seja, este atributo junto com o *EntityType* será utilizado para identificar em quais telas os campos personalizados criados devem ser exibidos.
- ***IsRequired*** (*Bool*): Este atributo é para indicar se o campo criado é obrigatório.

CustomFieldValues

A *CustomFieldValues*, está tabela está associado aos tipos de campos de seleção e multiseleção quando o usuário não quiser utilizar alguma entidade do FlowUp para exibir as opções. Assim, o usuário terá que cadastrar valores para serem utilizados nesses tipos de campos. Cada coluna é representado da seguinte forma:

- **Id** (*Int*): Identificador chave único e autoincremento.
- ***Value*** (*String*): Este atributo será o valor exibido como opção para os campos de seleção e múltipla seleção.
- ***CustomFieldId*** (*Int*): Este atributo representa o relacionamento entre CustomField e *CustomFieldValues*.
- ***IsActive*** (*Bool*): Esta coluna é para a exclusão lógica do valor, assim, mesmo que o valor não seja mais uma opção, não vai gerar problemas quando este valor já estiver sido escolhido como valor de resposta para o campo personalizado.

CustomFieldAnswer

A *CustomFieldAnswer* é responsável por armazenar as respostas dos campos personalizados enviados pelos dos usuários. Esta tabela tem suas colunas da seguinte forma:

- **Id** (*Int*): Identificador chave único e autoincremento.
- ***Value*** (*String*): Este atributo é o valor escolhido como resposta para o campo.
- ***CustomFieldEntityId*** (*Int*): O valor desta coluna define o relacionamento entre *CustomFieldAnswer* e *CustomFieldEntity*.
- ***EntityId*** (*Int*): Aqui será armazenado o Id da entidade, assim ao criar uma entidade que tenha campos personalizados será possível resgatar as repostas dessa entidade.