UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS RUSSAS

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ALEXANDRE VILA NOVA ALBUQUERQUE LIMA

SISTEMA DE GERENCIAMENTO DE REQUISITOS PARA A COMUNIDADE ACADÊMICA: UMA ABORDAGEM ÁGIL COM CONTROLE DE APROVAÇÃO E VINCULAÇÃO DE ARTEFATOS

RUSSAS

2025

ALEXANDRE VILA NOVA ALBUQUERQUE LIMA

SISTEMA DE GERENCIAMENTO DE REQUISITOS PARA A COMUNIDADE ACADÊMICA: UMA ABORDAGEM ÁGIL COM CONTROLE DE APROVAÇÃO E VINCULAÇÃO DE ARTEFATOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marcos Vinicius de Andrade Lima.

RUSSAS

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

L696s Lima, Alexandre Vila Nova Albuquerque.
Sistema de gerenciamento de requisitos para a comunidade acadêmica: uma abordagem ágil com controle de aprovação e vinculação de artefatos / Alexandre Vila Nova Albuquerque Lima. – 2025.
48 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Ciência da Computação, Russas, 2025.
Orientação: Prof. Dr. Marcos Vinicius de Andrade Lima.

1. Sistema de gerência de requisitos. 2. Desenvolvimento web. 3. Gestão de requisitos. 4. Requisitos e artefatos. I. Título.

CDD 005

ALEXANDRE VILA NOVA ALBUQUERQUE LIMA

SISTEMA DE GERENCIAMENTO DE REQUISITOS PARA A COMUNIDADE ACADÊMICA: UMA ABORDAGEM ÁGIL COM CONTROLE DE APROVAÇÃO E VINCULAÇÃO DE ARTEFATOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: ____/____/_____

BANCA EXAMINADORA

_____
Prof. Dr. Marcos Vinicius de Andrade Lima (Orientador)
Universidade Federal do Ceará (UFC)

_____
Profa. Dra. Patrícia Freitas Campos de Vasconcelos
Universidade Federal do Ceará (UFC)

_____
Profa. Dra. Marília Soares Mendes Albuquerque
Universidade Federal do Ceará (UFC)

Dedico a Deus, minha família, meus amigos e meu orientador. Obrigado por tudo.

AGRADECIMENTOS

Expresso minha gratidão ao Prof. Dr. Marcos Vinicius de Andrade Lima, pela orientação excepcional, dedicação e estímulo durante todo o desenvolvimento deste projeto.

Agradeço também aos professores que integraram a banca examinadora Profa. Dra. Patrícia Freitas Campos de Vasconcelos e a Profa. Dra. Marília Soares Mendes Albuquerque, pelo tempo dedicado e pelas valiosas contribuições e sugestões.

Sou grato aos meus colegas e amigos da faculdade, pelo apoio, reflexões, críticas construtivas e pela amizade cultivada ao longo dessa trajetória.

Finalmente, agradeço à minha família pelo suporte incondicional, paciência e incentivo constante, fatores fundamentais para a conclusão deste trabalho.

“O desenvolvimento de software deve ser considerado um processo de evolução contínua, no qual as mudanças são inevitáveis e necessárias para atender às necessidades dos usuários e às condições do ambiente.” (SOMMERVILLE, 2011, p. X).

RESUMO

No desenvolvimento de um software são envolvidas diversas fases, como a obtenção de requisitos com a ajuda de *stakeholders*, planejamento de atividades para implementação dos requisitos, testes das funcionalidades desenvolvidas pelo time, entre outras. Dentre essas fases, a que mais causa impacto caso não seja realizada corretamente, é a de obtenção de requisitos, pois ela tem a responsabilidade de auxiliar o restante do fluxo durante a esteira de desenvolvimento, já que, ao não ser realizado corretamente, pode acarretar em erros de desenvolvimento e a realização de testes sem necessidade. Por isso, a gerência dos requisitos possui grande relevância em qualquer projeto de software. Mas, observa-se que as principais ferramentas que auxiliam no gerenciamento de requisitos, não são livres. Isso faz com que diversos projetos acadêmicos realizem a gestão de requisitos de forma manual, trabalhosa e ineficaz. Nesta pesquisa é apresentado um software livre, desenvolvido segundo princípios das metodologias ágeis, que tem como objetivo facilitar o gerenciamento de requisitos em projetos de desenvolvimento de software. Essa solução surge como uma resposta aos problemas identificados em diversas disciplinas de graduação dos cursos de Ciência da Computação e Engenharia de Software do Campus Russas da Universidade Federal do Ceará, que necessitam gerenciar requisitos, automatizar tarefas de desenvolvimento e facilitar o processo que hoje é feito manualmente. O sistema desenvolvido possui arquitetura web, utiliza tecnologias *React* e *NodeJS* e mantém organização dos requisitos e artefatos por projeto, necessitando que seus utilizadores façam uso de autenticação de usuários. Embora a primeira versão tenha alcançado seus objetivos, pretende-se aprimorar a experiência de usuário do software, para deixar cada vez mais o processo de gerir requisitos fluidos para qualquer usuário. Pretende-se também realizar a expansão do público alvo da plataforma, firmando parcerias para implantação e aprimoramento do software em outras instituições.

Palavras-chave: Sistema de gerência de requisitos; Desenvolvimento web; Gestão de requisitos; Requisitos e artefatos.

ABSTRACT

Software development involves several phases, such as obtaining requirements with the help of stakeholders, planning activities to implement the requirements, testing the functionalities developed by the team, among others. Among these phases, the one that causes the most impact if not performed correctly is the requirements obtaining phase, as it is responsible for assisting the rest of the flow during the development process. If not performed correctly, it can lead to development errors and unnecessary testing. Therefore, requirements management is highly relevant in any software project. However, it is observed that the main tools that assist in requirements management are not free. This means that several academic projects perform requirements management manually, in a laborious and ineffective manner. This research presents free software, developed according to the principles of agile methodologies, which aims to facilitate requirements management in software development projects. This solution was developed as a response to problems identified in several undergraduate courses in Computer Science and Software Engineering at the Russas Campus of the Federal University of Ceará, which require requirements management, automation of development tasks and facilitation of the process that is currently performed manually. The system developed has a web architecture, uses React and NodeJS technologies and maintains organization of requirements and artifacts by project, requiring its users to use user authentication. Although the first version has achieved its objectives, the aim is to improve the user experience of the software, to make the process of managing requirements increasingly fluid for any user. The aim is also to expand the target audience of the platform, establishing partnerships for the implementation and improvement of the software in other institutions.

Keywords: Requirements management system; Web development; Requirements management; Requirements and artifacts.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

No contexto do desenvolvimento de uma solução computacional, seja a construção de uma grande aplicação bancária ou até mesmo uma atividade acadêmica, a gestão dos requisitos dessas soluções é uma atividade crucial para avaliar a aplicação, definir os próximos passos, criar atividades específicas, pensar em melhorias e sugestões, além de aumentar a taxa de sucesso da entrega da solução (Barros, 2018). A gestão de requisitos é um desafio que afeta, em grande parte, as equipes de engenharia de software (Santos, 2015). Consequentemente, a comunidade acadêmica, que frequentemente desenvolve softwares, também enfrenta esse problema devido à falta de um gerenciamento eficiente de requisitos, muitas vezes realizado de forma manual (Azevedo, 2017).

Para a gestão de requisitos, existem algumas ferramentas que oferecem um período de teste ou uma versão limitada para seus usuários, mas que só disponibilizam todas as funcionalidades mediante a aquisição de planos pagos. Entre essas ferramentas, podemos citar: o Jira, um sistema de gerenciamento de requisitos que, além de lidar com a gestão de requisitos no dia a dia, oferece um quadro Kanban ágil (Ortu *et al*., 2015); a ReQtest, que permite o gerenciamento de requisitos e o monitoramento de testes por meio de uma estrutura em nuvem, proporcionando rastreabilidade plena das mudanças ágeis ao longo do processo (Siddiqui *et al.*, 2020); e a DOORS, que possibilita a criação de artefatos para requisitos, a interligação entre eles, uma melhor visualização de fluxos e documentações, além de oferecer colaboração entre a equipe e utilizar inteligência artificial para aprimorar a qualidade dos requisitos (Lottin; Dalfovo, 2016).

Trazendo esse contexto para o meio universitário, temos um exemplo prático em nossa universidade. A disciplina ministrada pelo orientador deste trabalho tem enfrentado essa problemática, pois a ausência de uma aplicação acessível para a gestão de requisitos torna o processo mais trabalhoso. Atualmente, a revisão de documentos ocorre de forma manual, exigindo a análise de diversos arquivos em PDF enviados pelos alunos. Caso sejam necessárias correções, novos arquivos são solicitados, e o processo precisa ser refeito até que tudo esteja validado corretamente. Esse fluxo consome tempo e esforço devido à falta de uma ferramenta adequada para esse tipo de avaliação.

A gestão de requisitos não é um desafio exclusivo do ambiente acadêmico, mas também de diversas empresas que enfrentam dificuldades para organizar seus requisitos, manter versões, aprovar ideias sem perder a rastreabilidade das versões anteriores, e realizar atualizações de forma sistemática (Santos, 2015). Neste trabalho, abordamos especificamente

o contexto acadêmico, com foco em disciplinas que envolvem a documentação e o desenvolvimento de software, uma vez que os problemas mencionados afetam tanto os alunos, no desenvolvimento de suas aplicações, quanto os professores, na avaliação dos requisitos necessários para o progresso dos projetos.

Os desafios decorrentes dessa problemática incluem a falta de otimização e praticidade no gerenciamento de requisitos, a ausência de controle de versão, a necessidade de controle de acesso para monitoramento das ações dos usuários, entre outros. Um dos principais problemas na obtenção de requisitos é o fato de, muitas vezes, ser realizada manualmente, tornando-se suscetível a erros humanos, o que exige retrabalho para corrigir inconsistências e falhas no processo (Azevedo, 2017).

Com base na problemática identificada, formulamos a seguinte questão de pesquisa: como uma ferramenta de software livre pode auxiliar o processo de gerenciamento de requisitos, especialmente em projetos acadêmicos? Para responder a essa questão, estabelecemos os seguintes objetivos:

1.1 Objetivo Geral

Desenvolver uma aplicação web para gerenciamento de requisitos voltada à comunidade acadêmica, com foco em disciplinas que envolvem o desenvolvimento de soluções e/ou documentos relacionados a software.

1.2 Objetivos Específicos

- Realizar a análise do negócio para identificar os requisitos funcionais e não funcionais do sistema desenvolvido;
- Projetar a estrutura e a arquitetura do sistema de gerenciamento de requisitos, garantindo que ele atenda às necessidades identificadas;
- Planejar o desenvolvimento do sistema utilizando a metodologia Scrum, dividindo o trabalho em sprints e garantindo entregas iterativas e incrementais;
- Implementar o sistema especificado por meio de uma arquitetura cliente-servidor, utilizando uma linguagem de programação moderna.

1.3 Estrutura do trabalho

Este trabalho está organizado em seis seções, incluindo a Seção de introdução previamente apresentada. Na Seção 2, apresenta-se o referencial teórico, abordando os conceitos de desenvolvimento ágil, metodologia Scrum, tecnologias utilizadas e trabalhos relacionados. A Seção 3 descreve a metodologia e o desenvolvimento do projeto, detalhando o planejamento, a definição de requisitos, a estrutura das sprints e a implementação do software. Na Seção 4, apresentam-se os resultados obtidos, incluindo a avaliação das sprints, os desafios encontrados e uma comparação com aplicações que fornecem similaridades. Na Seção 5, abordam-se as considerações finais, as limitações do trabalho e sugestões para trabalhos futuros. Por fim, a Seção 6 apresenta as referências bibliográficas, com todas as fontes utilizadas para a elaboração deste trabalho.

2 REFERENCIAL TEÓRICO

O gerenciamento de requisitos em equipes que trabalham diariamente com desenvolvimento de sistemas, é um tema bastante discutido em Engenharia de Software, tendo em vista que a análise, definição e visualização desses requisitos, tende influenciar o sucesso de um projeto, tendo em vista que a extração correta desses requisitos permite que as necessidades dos usuários que irão utilizar um software sejam atendidas além das expectativas de stakeholders envolvidos, minimizando riscos de falhas e garantindo uma entrega com qualidade (Santos, 2015).

A seguir, serão apresentados os temas mais relevantes para nosso trabalho, trazendo uma revisão da literatura existente sobre o tema da gestão de requisitos em projetos tecnológicos.

2.1 Engenharia de Requisitos

Requisitos são compreendidos como definições documentadas que detalham o funcionamento de um software, aplicação ou projeto, além de abranger as ações possíveis e os casos de uso que podem surgir durante sua utilização (Wiltgen, 2022). É importante mencionar que existem requisitos bem elaborados, cujas definições, a partir da realização de testes de caso de uso, podem levar a melhorias e à evolução da documentação (Wiltgen, 2022). A rastreabilidade dos requisitos é um aspecto essencial, pois permite acompanhar a vida útil de cada requisito, validando sua trajetória desde a origem até a implantação e uso (Gaspardine Júnior; Sobrinho, 2022).

A gestão de requisitos consiste na ação de entender, definir e documentar as demandas de um sistema. Este procedimento é crucial para o entendimento e a implantação do software, pois é nesse estágio que todos os possíveis problemas são analisados. Se algum erro não for identificado nessa análise, ele poderá impactar o software durante a sua implantação, já que a forma como é apresentado e interpretado pela equipe de desenvolvimento pode resultar em falhas (Pressman; Maxim, 2021).

2.2 Requisitos de Software

Os requisitos de software representam um conjunto de características e funcionalidades que definem o que um sistema deve fazer para atender às demandas e

necessidades de usuários e stakeholders. Eles podem ser classificados como requisitos funcionais, que especificam funções específicas que o sistema deve ter, como, por exemplo, “um sistema deve possuir autenticação por dois fatores”. Além disso, existem requisitos não funcionais, que tratam de características que não estão diretamente ligadas às funcionalidades do sistema, como segurança e desempenho (Sommerville, 2011). O autor ainda ressalta que os requisitos podem mudar ao longo do tempo, sempre buscando verificar se os artefatos gerados atendem às exigências especificadas pelo cliente.

De maneira mais específica, um requisito pode ser qualquer demanda expressa por interessados em um software, englobando características ou funcionalidades que devem ser desenvolvidas para o produto final, além de abordar limitações que possam existir (Fernandes, 2017). A definição de requisitos é de suma importância em um cenário onde as remunerações são atreladas às atividades especificadas. A validação dos requisitos, se não for realizada de forma completa, entrega a responsabilidade do requisito a um desenvolvedor, que pode não ter o conhecimento necessário para esclarecer tais demandas, resultando em um aumento de custos, uma vez que o requisito não foi definido adequadamente antes de entrar na esteira de desenvolvimento (Vazquez, 2016). O mesmo autor salienta que, durante o processo de definição de requisitos, é necessário descrever os fluxos operacionais principais e secundários, além de determinar como o sistema deve responder às ações do usuário e as limitações que o software pode apresentar.

Além disso, Vazquez (2016) menciona a existência de um documento entre o cliente e a equipe de desenvolvimento, que descreve as expectativas do cliente e os resultados a serem entregues ao final do processo de desenvolvimento. Esse documento, conhecido como especificação de requisitos, tem como objetivo registrar de forma precisa todas as necessidades do cliente e verificar se elas foram atendidas. Isso permite que o cliente valide os resultados e forneça feedback, possibilitando que, em caso de falhas, o serviço seja corrigido e o requisito atualizado.

2.3 Desenvolvimento de Software

A criação de software, é uma atividade uma serão desenvolvidos protótipos comerciais específicos ou produtos, que serão destinados ao uso de um usuário ou grupo de usuários, que será produzido por um time de desenvolvedores (Sommerville, 2011). Ainda segundo o autor, a produção de software é algo contínuo, que será mantido e alterado ao longo de sua história, tendo grande suporte da engenharia de software, onde o ciclo de

implementação do desenvolvimento, será um estágio onde os requisitos definidos do sistema, serão transformados em um protótipo ou software executável, que sempre envolver processos de concepção e programação.

Um dos grandes desafios para o desenvolvimento de uma aplicação é a dificuldade em se analisar as necessidades do cliente e a obtenção de soluções práticas e possíveis que podem ser desenvolvidas para conseguir assim resolver o problema que o cliente espera que seja resolvido (Vargas, 2016). Ainda segundo o autor, é descrito que existem inúmeros estudos a respeito da taxa de sucesso do desenvolvimento de softwares, que muitas vezes é decorrente de uma má utilização ou a não utilização de metodologias de desenvolvimento, tendo em vista um mal planejamento e um mal gerenciamento do processo.

A qualidade de software também é também algo que deve ser validado em todas as etapas de desenvolvimento de um software, que não se resume necessariamente a suprir ou não as necessidades dos clientes através dos requisitos, mas também atender questões de desempenho, segurança, manutenibilidade e conformidade com as demais necessidade de um cliente (Sommerville, 2011). A qualidade do software também, segundo o autor, é de essencial importância para garantir a manutenção e que o código escrito tenha possibilidade de evolução no futuro.

A combinação de práticas do Scrum se torna um grande parceiro para o desenvolvimento de algum software, tendo em vista que tendem trazer uma maior flexibilidade a algum projeto que esteja sendo desenvolvido, além de trazer maior agilidade e adaptabilidade (Vargas, 2016). Ainda segunda o autor, tal junção possibilita que o resultado esperado com as entregas de valor sejam mais rápidas e com maior qualidade. Dentre as possibilidades de desenvolvimento, temos a abordagem ágil para software, detalhada a seguir.

2.4 Desenvolvimento Ágil de Software

O desenvolvimento ágil de um software é uma metodologia de trabalho que visa a produção do mesmo, de forma iterativa e incremental, visando a rapidez no processo. Tem por objetivo, a valorização da colaboração com o cliente, as mudanças, o próprio software e como os indivíduos interagem durante o processo, visando uma rápida adaptabilidade durante o processo de produção, com foco na entrega ao final (Badoco; Almeida; Lucas, 2018). Além disso segundo o autor, não existe a necessidade de seguir um plano fixo durante o desenvolvimento de um software, tendo em vista as mudanças que podem acontecer, de acordo com as necessidades do cliente, então esse processo ágil auxilia em adaptar uma

solução de forma rápida e eficiente para um determinado cliente, tendo em vista que o sistema funcionando tem mais impacto do que um documento mais extenso além do relacionamento de uma equipe de desenvolvimento com os próprios clientes.

A gestão de um projeto e seus requisitos, traz uma grande responsabilidade e complexidade para uma equipe, tendo em vista que se a obtenção de dados e informações para consolidar ideias e atividades de um projeto, pode influenciar diretamente na gerência dos requisitos previamente definidos, de forma a tornar pouco eficiente e propenso a erros. A gestão de requisitos de uma equipe frente ao desenvolvimento de um software traz vantagens em relação à proteção contra situações de risco que possam acontecer, controlando o que está sendo desenvolvido com uma visão que possibilita a antecipação e prevenção de situações que podem colocar em risco o software (Vargas, 2018). A gerência de forma ótima, traz vantagens quanto a tomada de decisões de forma ágeis e assertivas, tornando a previsibilidade de desenvolvimento mais coerente com as requisições de software e mercado.

Na metodologia de desenvolvimento ágil, um software é produzido de forma incremental e contínua, fazendo com que correções e alterações sejam absorvidas pela aplicação ao longo do tempo por suas versões, sempre com o sistema em pleno funcionamento, para isso foram desenvolvidos alguns princípios, dentre eles podemos citar: valor, flexibilidade, frequência, união, motivação, comunicação, funcionalidade, sustentabilidade, revisão, simplicidade, organização e autoavaliação. Com a equipe de desenvolvimento atuando de forma a seguir esses princípios, os resultados se tornam cada vez mais condizentes com o esperado (Bittar *et al.*, 2018). A partir desses princípios, foram também desenvolvidas algumas metodologias ágeis, como o Scrum, que abordaremos a seguir.

2.5 Scrum

A metodologia Scrum, é um grupo dinâmico que tem por principal objetivo, o gerenciamento de empresas e projetos relacionados ao mercado. O Scrum deve ser simples de entender mas extremamente difícil de se dominar, tendo em vista que seu foco é desenvolver e manter projetos complexos, permitindo diferentes padrões e técnicas adaptativas durante suas práticas, adicionando requisitos e artefatos específicos para cada organização, tornando o processo de desenvolvimento intuitivo, produtivo e criativo mas sem perder a base estrutural principal (Vargas, 2018).

O Scrum é um *framework* que surgiu nos anos 90 e vem sendo amplamente utilizado no desenvolvimento de projetos nos mais variados níveis de complexidade por demonstrar uma alta efetividade dos equipe com seus processos aliada à praticidade de gestão. Além de ser utilizado no desenvolvimento de produtos, o Scrum passou a ser utilizado também na gestão de empresas. Os times que adotam o Scrum, tendem a ter uma maior autonomia ao escolher qual a melhor forma de completar seu trabalho, em vez de serem dirigidos por outros de fora do time, esses times são compostos por um *Product Owner*, o time de desenvolvimento e um *Scrum Master* (Soares *et al.*, 2018).

Ainda segundo Soares *et al.* (2018), o Scrum define quatro eventos sua determinação para inspeção e adaptação: planejamento do *Sprint*, reunião diária, revisão do *Sprint* e retrospectiva do *Sprint*. O *Product Owner* deve organizar uma lista de requisitos necessários para o projeto, chamada de *Product Backlog*, onde seus itens são atualizados diariamente e tem seus níveis de prioridade definidos, sendo que os ciclos de tempo de produção que chamamos de *Sprints*, a lista definida em *backlog* devem ser desenvolvidas e entregues ao final de cada *sprint*. As atividades das *sprints* são estimadas em relação ao tempo durante um evento no qual o *Scrum Master* coordena, realizando o planejamento e equilibrando os interesses entre o *Product Owner* e o time de desenvolvimento.

2.6 Trabalhos Relacionados

A gestão de requisitos, é um tema de vital importância na identificação e validação das necessidades de usuários, de maneira a possibilitar uma organização sobre esses requisitos, visando um melhor acompanhamento e alterações que possam acontecer para os requisitos durante o processo de desenvolvimento, tendo em vista que os erros obtidos a partir da coleta de requisitos tem total impacto na taxa de sucesso de um determinado sistema (Fagundes, 2021).

Este estudo propõe uma abordagem prática da utilização de metodologias ágeis juntamente com um software, como uma forma de explorar a automatização da gerência de requisitos trazendo uma solução eficaz para gerenciar projetos de software de maneira mais flexível e estruturada ao mesmo tempo.

O levantamento dos trabalhos relacionados se deu por dois motores principais de pesquisa, o Google Acadêmico e o Repositório Institucional da UFC, para um tempo estimado de publicação para os últimos dez anos, entre 2015 e 2025. Para a busca foram realizadas pesquisas num período de 10 anos utilizando as palavras chave para o problema

que estava sendo implementando uma solução, então foram utilizadas palavras como gerenciamento de requisitos de software, gerenciamento de requisitos, requisitos de software, etc. Posterior a isso, foram selecionados dentro dos 16500 resultados, os trabalhos que mais se aproximaram da ideia apresentada neste trabalho, trazendo pontos de forma clara e didática, como a gerência de requisitos, a ideia de software para resolver um determinado problema e o desenvolvimento de uma aplicação web como software de resultado. O Trabalho de Barros (2018) foi escolhido por concluir que falhas ao requisito podem comprometer muito o projeto todo. Já para Silva (2017) foi escolhido por trazer uma similaridade na motivação, sendo as dificuldades encontradas entres os alunos e professores para realizar alguma ação dentro do meio universitário. Por fim, o de Muniz (2023) foi escolhido primeiramente por ser um trabalho mais recente e trazer um software que ajuda o meio universitário, demonstrando a importância desses tipos de software para alunos e professores.

O trabalho de Barros (2018), traz a importância da gestão de requisitos para projetos de desenvolvimento de software, no qual o objetivo é a disseminação dos conhecimentos acerca do gerenciamento requisitos para a comunidade acadêmica, levando a identificação das falhas no gerenciamento de requisitos e como elas podem afetar o projeto de desenvolvimento de software. Ainda sobre o que abordado pelo autor, que define sua pesquisa como sendo um processo que tem como foco a descoberta de novas situações e/ou soluções, em diferentes áreas, sendo a natureza do estudo é qualitativa, com o objetivo responder questões levantadas acerca do tema de pesquisa, além da flexibilidade na interpretação, tradução e aprofundamento das questões referentes ao tema, trazendo como conclusão a importância de gestão de requisitos, além da identificação prévia das falhas que dessa gestão, de forma a detectar esses problemas e resolvê-los o mais rápido possível, para que não haja comprometimento do software em desenvolvimento.

Outro exemplo que podemos mencionar, vem de Silva (2017), que trás um sistema de gerenciamento para trabalhos de conclusão de curso, onde sua abordagem vem da substituição de uma aplicação já existente, trazendo uma atualização com alguns requisitos novos, como a geração automática de atas de defesa e gerenciamento de usuários. Segundo o autor, a aplicação antiga foi avaliada para levantar as funcionalidades necessárias que esta executava, para serem integrados novos requisitos que foram definidos a partir das dores dos usuários, onde tinham dificuldades em escolher as temáticas para o desenvolvimento de seus trabalhos de conclusão de curso.

Vale a pena também citar Muniz (2023) que traz como principal objetivo, o desenvolvimento de um sistema web, que tem por finalidade gerenciar e automatizar o

processo de se submeter artigos para o evento encontros universitários da Universidade Federal do Ceará do campus de Russas. No trabalho de Muniz (2023) é utilizado uma metodologia ágil de desenvolvimento, para deixar o processo mais dinâmico além de utilizar das demandas e exigências dos organizadores do evento, passando por etapas de definição, especificação, validação de requisitos, implementação e implantação.

A partir das contribuições fornecidas pelos trabalhos, podemos compreender a importância da automatização e do uso de metodologias ágeis na administração da gestão de requisitos. Ao combinar metodologias ágeis com a utilização de software, este estudo traz uma solução para melhorar o processo de gestão de requisitos, propondo um software web eficaz e inteligente para atender as necessidades presentes em projetos de desenvolvimentos, se aproximando das ideias de Silva (2017) e Muniz (2023), e com o objetivo de demonstrar a importância do processo de gestão de requisitos para o desenvolvimento de um software, que traz como referência Barros (2018).

O Quadro 1, traz uma síntese das características gerais deste trabalho, mostrando um comparativo entre os trabalhos citados acima:

Quadro 1 – Comparativo entre as características dos trabalhos

Características	Silva (2017)	Barros (2018)	Muniz (2023)	Este trabalho
Foi implementado e disponibilizado a um determinado grupo de usuários.	Sim	Não	Sim	Não
Foi hospedado em nuvem ou servidor local.	Local	Não teve	Nuvem	Nuvem
Possui software implementado.	Sim	Não	Sim	Sim
Utilizou metodologia ágil.	Não	Não	Sim	Sim

Fonte: elaborado pelo autor.

Como visto no Quadro 1, este trabalho ainda passará por testes com usuários, já que ainda não se foi disponibilizado para teste para a comunidade acadêmica da Universidade Federal do Ceará, Campus Russas para que haja a possibilidade de obtenção de um *feedback* sobre melhorias que possam acontecer para que assim o software seja integrado aos alunos e professores.

3 METODOLOGIA E DESENVOLVIMENTO

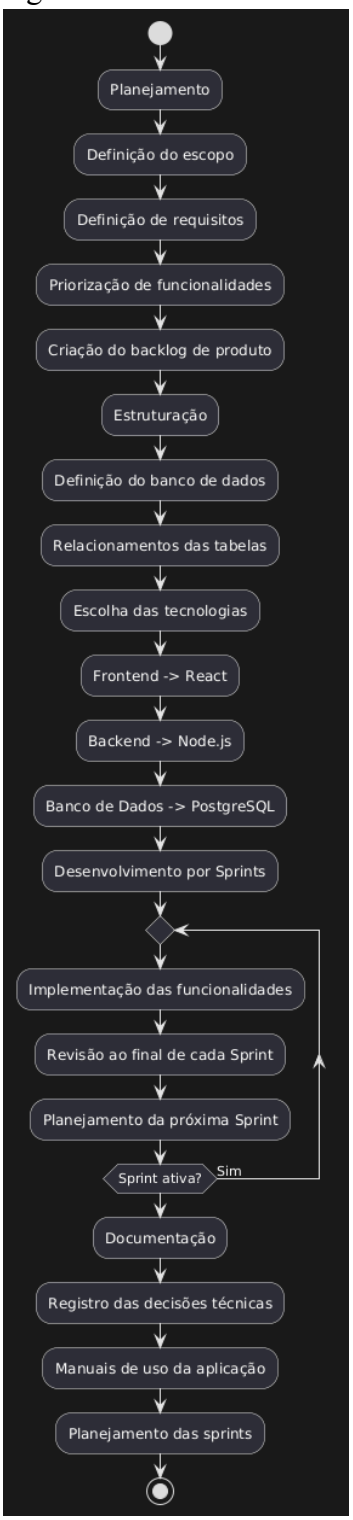
O desenvolvimento da aplicação foi baseado em metodologias ágeis, com destaque para o Scrum adaptado para nosso contexto. O Scrum é um conjunto de práticas ágeis que ajudam no gerenciamento de projetos, ajudando uma equipe a trabalhar de forma ágil e organizada. Embora o desenvolvimento de sistemas geralmente envolve times, neste caso, o trabalho foi realizado individualmente. Ainda sim, a divisão das atividades em *sprints* e a entrega iterativa e incremental de funcionalidades foram mantidas para garantir organização e foco. O processo de desenvolvimento será dividido em quatro etapas principais: o planejamento, a estruturação de suas entidades, o desenvolvimento em si e a documentação. Podemos detalhar cada uma delas da seguinte forma:

- Planejamento: Definição do escopo do projeto, definição de requisitos, priorização de funcionalidades e elaboração do *backlog* do produto, criando assim atividades para o desenvolvimento da aplicação;
- Estruturação: Definição das tabelas do banco de dados assim como seus relacionamentos, visando uma maneira ótima de salvar, organizar e recuperar os dados. Nessa etapa, também foram definidas as linguagens de programação utilizadas para desenvolvimento do *backend* e *frontend*, tais como:
    - *Frontend*: React, para a criação de uma interface moderna e responsiva;
    - *Backend*: Node.js, para a implementação da lógica de negócio e integração com o banco de dados;
    - Banco de Dados: PostgreSQL, para armazenar e gerenciar os dados de requisitos, artefatos e usuários.
- Desenvolvimento por *Sprints*: Cada *sprint* terá duração de uma semana, com foco na entrega de funcionalidades específicas. Ao final de cada *sprint*, será realizada uma revisão para avaliar o progresso e planejar as próximas etapas;
- Documentação: Todas as etapas do projeto serão documentadas, incluindo o planejamento das *sprints*, as decisões técnicas e os manuais de uso da aplicação.

Essa abordagem ágil adaptada do Scrum, foi de vital importância para o desenvolvimento individual do sistema, permitindo uma evolução constante do projeto, com a possibilidade de ajustes e melhorias ao longo do processo, garantindo que a aplicação atenda aos seus requisitos previamente estabelecidos.

Tal aplicação tem como finalidade a criação, edição e aprovação de requisitos, esses que por sua vez podem ter artefatos atrelados a eles, que podem estar em um ou mais requisito e sua versão para cada requisito, pode ser controlada pelo usuário, onde o usuário tem a possibilidade de escolher para quais versões de requisito o artefato será atualizado. O sistema também possui uma estrutura baseada em projetos, onde cada projeto possui seus usuários que possuem suas permissões previamente definidas para conseguirem ministrar os requisitos e artefatos da solução, tendo em vista que nem todos os usuários possuem permissão para aprovar ou reprovar requisitos. Tais funcionalidades tem por finalidade otimizar e dinamizar o processo de gerência de requisitos em ambientes onde a tarefa de organizar e manter requisitos sejam o principal objetivo.

Figura 1 – Processo de desenvolvimento da aplicação



Fonte: elaborada pelo autor.

3.1 Planejamento do Projeto de Software

- Estruturação do projeto antes do início do desenvolvimento: foi pensado inicialmente nas linguagens de programação juntamente com o banco de dados para se obter a melhor combinação para o projeto requisitado;
- Levantamento de requisitos com *stakeholders* (usuários, clientes, equipe técnica): A partir de uma estruturação inicial feita, foi iniciado um processo de escrita dos requisitos, funcionais e não funcionais, juntamente com o professor orientador deste trabalho, que para esse caso, vem a ser o cliente;
- Planejamento das *sprints*: Com base nos requisitos escritos, estabeleceu-se uma definição de atividades e *sprints*, dividindo cada grupo de tarefas em *sprints* de no máximo uma semana.

3.2 Definição dos Requisitos

O levantamento dos requisitos é uma das principais etapas no processo de desenvolvimento de qualquer software, para a nossa aplicação foram definidos alguns requisitos principais, tais como:

- Requisitos funcionais

Quadro 2 – Requisitos funcionais

RF01	O sistema deve manter requisitos de um sistema, informando nome do requisito, descrição, tipo de requisito (funcional ou não funcional) e seus artefatos;
RF02	O sistema deve possibilitar a associação de múltiplos artefatos a um ou mais requisitos.
RF03	O sistema deve permitir criar, editar, deletar e listar artefatos.
RF04	O sistema deve exibir na listagem de artefatos, os requisitos associados aos mesmos.
RF05	O sistema deve permitir a atualização da versão de cada requisito para cada artefato.
RF06	O sistema deve permitir criar, editar e

	deletar projetos, onde a entidade é composta apenas pelo nome.
RF07	O sistema deve permitir criar, editar e deletar usuários, onde a entidade é composta apenas pelo nome, onde sua entidade é composta basicamente por nome, email, telefone, senha e projetos.
RF08	O sistema deve permitir a associação de múltiplos projetos e suas permissões a um mesmo usuário.
RF09	O sistema deve permitir a escolha de um projeto dentre todos os quais o usuário foi associado para iniciar o acesso ao realizar o login.

Fonte: elaborado pelo autor.

- Requisitos não funcionais (desempenho, segurança, usabilidade)

Quadro 3 – Requisitos não funcionais

RNF01	O sistema deve ser intuitivo e de fácil utilização pelos usuários.
RNF02	O sistema deve ser escalável, suportando grande número de requisições por múltiplos usuários.
RNF03	O sistema deve permitir a troca de projetos a qualquer momento.
RNF04	O sistema deve guardar dados de forma segura.
RNF05	O sistema deve possuir um tratamento adequado de erros, informando o usuário de forma entendível o erro que aconteceu.
RNF06	O sistema deve possuir uma lógica de permissões para usuários, tendo possibilidade de múltiplas permissões para que possam ser concedidas.
RNF07	O sistema deve possibilitar que as permissões delimitem as ações que o usuário pode realizar dentro do software.

Fonte: elaborado pelo autor.

3.3 Elaboração do projeto do sistema

O software consiste em um sistema para gerenciamento de requisitos e artefatos, que tem por finalidade auxiliar as disciplinas de Interação Humano Computador, Engenharia de Software, programação orientada a objetos e análise e projetos de sistemas a gerenciarem de forma ótima trabalhos e atividades que precisam ser enviadas por alunos e corrigidos pelos professores de forma a tornar o processo que hoje é feito manual, mais automático, com as facilidades de software web. Um caso de uso simples, seria a criação de um requisito, onde o usuário iria realizar o login, selecionar um projeto, caso não existam projetos listados, ele será redirecionado para a criação de um projeto, logo após o usuário deverá adicionar artefatos a serem associados a requisitos e finalmente poderá criar requisitos, podendo adicionar um ou mais artefatos criados anteriormente.

A elaboração do software, seguiu uma abordagem por meio da metodologia ágil Scrum, permitindo alterações e correções rápidas ao longo do processo de desenvolvimento durante as *sprints*, buscando sempre respeitar a necessidade que o software busca solucionar a partir dos requisitos funcionais e não funcionais, pensando nos usuários finais. O sistema ainda não foi testado por um grupo de usuários, possibilitando uma validação e obtenção de *feedbacks*, sendo esse um dos principais motivos pelo qual o software não possui um gerenciamento eficiente de dados de usuário, sendo essa uma melhoria a ser realizada, a segurança das senhas dos usuários é gerenciada pelo próprio banco de dados, tendo a necessidade de uma implantação inicial para que alunos e professores possam realizam testes para que se possa fazer uma pesquisa para melhorias e correções e a partir disso, o desenvolvimento de todas as necessidades obtidas pela pesquisa para que o sistema possa ser implantado para a comunidade acadêmica. Nos próximos tópicos, serão abordados pontos de fundamental importância para o software, que são eles a Definição de requisitos, Estrutura das *Sprints* e Implementação do Software.

3.4 Estrutura das *Sprints*

As *sprints* foram organizadas de forma a terem no máximo uma semana, tendo em vista os prazos estabelecidos. Inicialmente, o projeto foi estruturado e então seus requisitos e atividades foram definidos, para cada *sprint* foi definido um objetivo específico em foco, inicialmente foi-se trabalhado no desenho inicial das tabelas do banco de dados, para se ter

uma noção das primeiras regras de negócios e conseguir pensar em soluções inteligentes para se integrar tais lógicas utilizando o *frontend* e *backend* do software.

A partir de uma visão inicial do banco de dados, foi implementado uma prototipação do *backend*, onde foram desenvolvidas ações iniciais para requisitos, com a possibilidade de criação, edição, deleção e recuperação desses dados, além de entidades relacionadas a usuários e permissões, permissões essas que evoluíram nas versões futuras e acabaram por se tornar uma tabela de relacionamento entre usuários e projetos.

Com um *backend* inicial que já pudesse ser utilizado, foi desenvolvido o *frontend*, um esqueleto para se ter uma ideia e dimensão do que se precisava ser feito e como deveria ser feito, para que a aplicação ficasse o mais intuitiva e fácil o possível para o uso de um usuário, assim foi-se criado algumas telas iniciais, como telas de login, cadastro, uma *sidebar* entre outras telas, onde todo esse processo aconteceu em duas *sprints* de uma semana.

Com o *frontend* construído, foi hora de voltar ao desenvolvimento do *backend* juntamente com atualização, estruturação e atualização de algumas tabelas do banco de dados, assim como a criação de outras, levando em conta quesitos de segurança. Após uma reestruturação do *backend*, foi pensado posteriormente em se fazer as lógicas, serviços e regras de negócios adicionais para as entidades de requisitos e artefatos, levando em conta todos os serviços de requisitos previamente estabelecidos. Por fim, foi feito o desenvolvimento das entidades relacionadas a projetos além de outros serviços relacionados a usuário, como sua edição e deleção. Ao fim desse processo, foram totalizadas quatro *sprints* para o desenvolvimento, organização e implantação do *backend*.

Ao final, foram definidas duas *sprint* para testes, correções e melhorias de código e regras de negócio para a aplicação, como definições para troca de projeto por parte do *frontend*, mudanças na exibição dos dados de usuário na *sidebar*, estilos e cores por parte de botões entre outras melhorias. Totalizando o processo, todo o desenvolvimento levou cerca de nove *sprints* para ser realizado, abaixo, podemos citar as regras de negócio estabelecidas assim como o processo de implementação no próximo tópico,

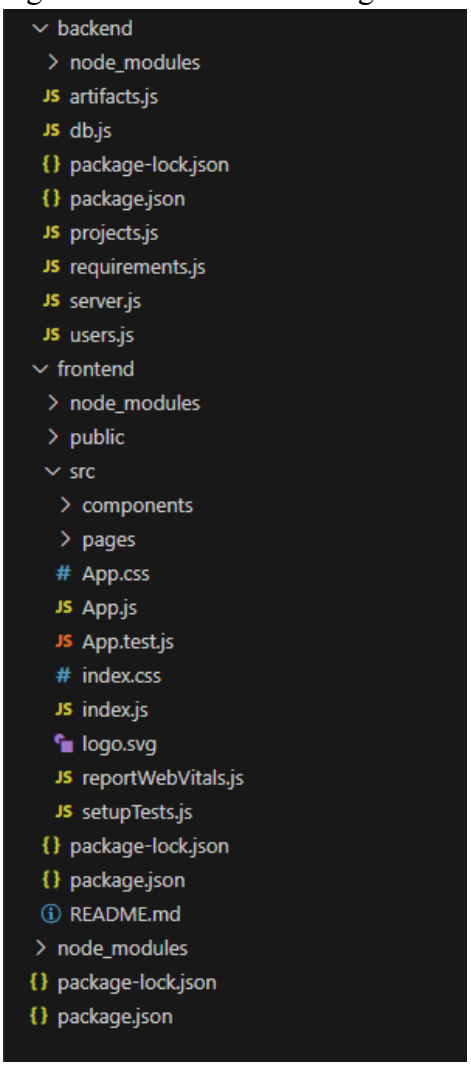
- Cada usuário pode estar associado a múltiplos projetos, mas deve selecionar um para trabalhar ao fazer login;
- Possibilidade de mudar projeto a qualquer momento da navegação;
- Requisitos e artefatos devem possuir controle de versão;
- Permissões devem ser atribuídas por usuário para cada projeto;
- Apenas administradores podem criar e excluir usuários;

- Projetos e requisitos não podem ser deletados se possuírem dependências ativas.

3.5 Implementação do Software

Para a implementação do software foi pensado em uma estrutura conjunta entre *backend* e *frontend*, com o *frontend* sendo escrito em React e o *backend* sendo em Node.

Figura 2 – Estrutura de código do software desenvolvido



Fonte: elaborada pelo autor.

3.5.1 Implementação Front End

A implementação do *frontend* tem por base a estrutura vista na Figura 2, as *pages* são todas as páginas utilizadas pelo sistema e as rotas de caminho de uma para outra usando o *react-router-dom*, uma biblioteca para implantar e gerenciar as rotas de um software web. Em

components, temos a construção de alguns componentes base, como tabelas, modais, *sidebar*, etc. O sistema se divide basicamente em quatro telas, Requisitos, Artefatos, Usuários e Projetos, tendo cada um, sua estrutura de conexões com o banco de dados e suas regras de negócios.

3.5.2 Implementação Back End

Na implementação de *backend* em Node, temos dois arquivos principais, que são o *db.js* e o *server.js*, no primeiro arquivo, estão as configurações de banco de dados para se conectar ao software, já no segundo, é o nosso servidor, que vai ficar escutando todas as chamadas que foram criadas além de instanciar o banco de dados para uso nas demais rotas, que estão contidas em *users.js*, *projects.js*, *artifacts.js* e *requirements.js*. Além disso, no *backend* foram configuradas as rotas para login e criação de usuários, quando logados, todos os usuários precisam enviar o token de identificação, que vai como resposta no login, em todas as chamadas realizadas a API.

3.5.2.1 Rotas da API

- **POST /projects:** Criar um novo projeto - Permite que um usuário crie um novo projeto.
- **GET /projects:** Obter todos os projetos - Permite que um usuário obtenha uma lista de todos os projetos com opções de filtro.
- **GET /projects/user:** Obter projetos de um usuário específico - Permite que um usuário obtenha todos os projetos que ele participa.
- **GET /projects/:id:** Obter um projeto por ID - Permite que um usuário obtenha os detalhes de um projeto específico.
- **PUT /projects/:id:** Atualizar um projeto - Permite que um usuário atualize os detalhes de um projeto existente.
- **DELETE /projects/:id:** Deletar um projeto - Permite que um usuário delete um projeto existente.
- **POST /requirements:** Criar um novo requisito - Permite que um usuário crie um novo requisito, validando se já existe um requisito ativo com o mesmo nome.

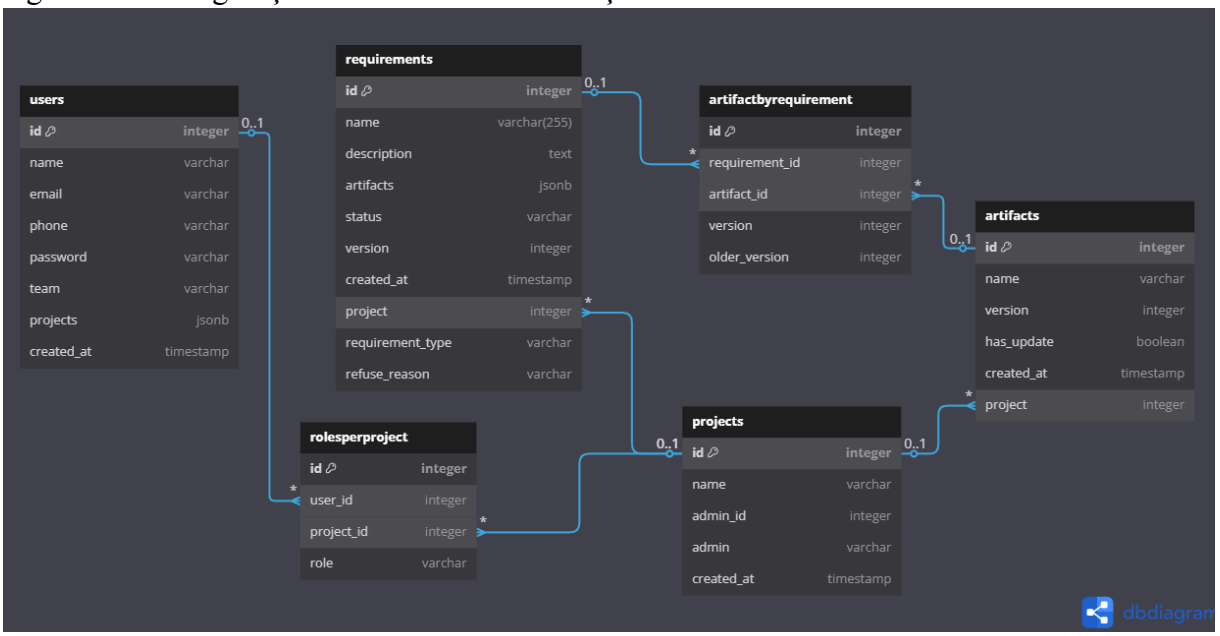
- **GET /requirements:** Obter todos os requisitos - Permite que um usuário obtenha uma lista de todos os requisitos com opções de filtro.
- **GET /requirements/:id:** Obter um requisito por ID - Permite que um usuário obtenha os detalhes de um requisito específico.
- **PUT /requirements/:id:** Atualizar um requisito - Permite que um usuário atualize um requisito existente.
- **DELETE /requirements/:id:** Deletar um requisito - Permite que um usuário delete um requisito existente.
- **PUT /requirements/reactivate/:id:** Ativar um requisito e desativar outro com o mesmo nome - Permite que um usuário ative um requisito, desativando outro requisito ativo com o mesmo nome.
- **PUT /requirements/deactivate/:id:** Desativar um requisito e ativar outro com o mesmo nome - Permite que um usuário desative um requisito e ative outro com o mesmo nome.
- **POST /register:** Registrar um novo usuário - Permite que um novo usuário se registre, verificando se o email já está cadastrado.
- **POST /login:** Login de usuário - Permite que um usuário faça login, verificando as credenciais e gerando um JWT.
- **GET /users:** Obter todos os usuários - Permite que um usuário autenticado obtenha uma lista de todos os usuários.
- **POST /users:** Criar um novo usuário - Permite que um usuário admin crie um novo usuário, incluindo a atribuição de projetos e funções.
- **DELETE /users/:id:** Deletar um usuário - Permite que um usuário admin delete um usuário, verificando se o usuário não possui entidades relacionadas.
- **GET /users/user:** Obter projetos de um usuário específico - Permite que um usuário autenticado obtenha todos os projetos associados a um usuário específico.
- **PUT /users/:id:** Atualizar informações de um usuário - Permite que um usuário admin atualize os dados de um usuário existente.
- **POST /artefacts:** Criar um novo artefato - Permite que um usuário crie um novo artefato associado a um projeto.
- **GET /artefacts:** Obter todos os artefatos com requisitos relacionados - Permite que um usuário obtenha uma lista de todos os artefatos, incluindo requisitos relacionados.

- **GET /artefacts/:id:** Obter um artefato por ID - Permite que um usuário obtenha os detalhes de um artefato específico.
- **PUT /artefacts/:id:** Atualizar um artefato - Permite que um usuário atualize um artefato existente, incrementando sua versão.
- **DELETE /artefacts/:id:** Deletar um artefato - Permite que um usuário delete um artefato existente.
- **PUT /artefacts/update/:id:** Atualizar o atributo *version* de um artefato - Permite que um usuário atualize a versão de um artefato e suas relações com requisitos.

3.5.3 Configuração banco de dados PostgreSQL

As entidades presentes no banco de dados estão presentes na Figura 3, nela são exemplificadas de como ficou definido cada entidade e suas relações. Para garantir uma maior escalabilidade e rapidez na recuperação de dados, foram criadas duas tabelas de relacionamento, para facilitar as interações entre tabelas que detinham de relacionamentos de muitos para muitos, em exemplo temos a entidade de requisitos, que pode ter vários artefatos associados e a entidade de artefatos podem estar em muitos requisitos, então para facilitar esses relacionamentos e na eficácia em recuperar dados para visualização além de guardar dados relacionados a versão, foi criado a tabela *artifactsByRequirements*.

Figura 3 – Configuração das tabelas e suas relações no banco de dados



Fonte: elaborada pelo autor.

Relação entre as Tabelas

- Usuários (*users*) podem estar associados a múltiplos projetos (*projects*) através da tabela rolesperproject, que define a função de cada usuário dentro de um projeto.
- Cada projeto (*projects*) pode ter vários requisitos (*requirements*) e artefatos (*artifacts*) vinculados a ele.
- Os requisitos (*requirements*) podem estar associados a múltiplos artefatos (*artifacts*) através da tabela intermediária *artifactbyrequirement*, que também armazena o controle de versão.
- Cada artefato (*artifacts*) pode pertencer a um único projeto (*projects*), mas pode ser utilizado em diversos requisitos (*requirements*).

3.6 Tecnologias Utilizadas

Para o desenvolvimento do software para o gerenciamento de requisitos, foram utilizadas tecnologias modernas e ferramentas que garantem a eficiência, escalabilidade e uma boa experiência de usuário, tanto para *backend* quanto para *frontend*. A escolha dessas tecnologias foi baseada em características de popularidade de mercado e melhor adequação aos requisitos do projeto, visando uma melhor manutenção e conformidade para futuras melhorias em seus requisitos.

No *frontend*, foi utilizado o *framework* React, uma biblioteca JavaScript onde segundo Boduch e Derks (2020) é amplamente adotada para a criação de interfaces de usuário interativas e responsivas, permitindo a construção de componentes reutilizáveis, o que facilita sua manutenção e a escalabilidade da aplicação, além de apresentar a possibilidade de integração com outras bibliotecas e ferramentas, como o *React Router* para gerenciamento de rotas, contribuiu para uma experiência de usuário fluida e dinâmica.

Para o desenvolvimento do backend, a escolha foi o Node.js, que para Casciaro e Mammino (2020) é um ambiente de execução em JavaScript que permite a criação de aplicações eficientes e de alta performance, tendo a capacidade de lidar com múltiplas operações assíncronas e sua possibilidade de integração com bibliotecas disponíveis através do npm. Além disso, foi escolhido o Node pela possibilidade da linguagem de programação base tanto para o backend quanto para o frontend, o JavaScript. Foi também utilizado o framework Express junto ao Node.Js, para facilitar e simplificar o gerenciamento de rotas e requisições HTTP.

Para o armazenamento de dados e seu gerenciamento, foi utilizado o PostgreSQL, um banco de dados relacional de código aberto e altamente confiável. O PostgreSQL foi a principal escolha por sua robustez, suporte a operações complexas e capacidade de lidar com grandes volumes de dados (Garrido; López; Constante, 2020). Além disso, sua integração com o Node.js através de bibliotecas como o pg facilitou a manipulação de dados de forma segura e eficiente.

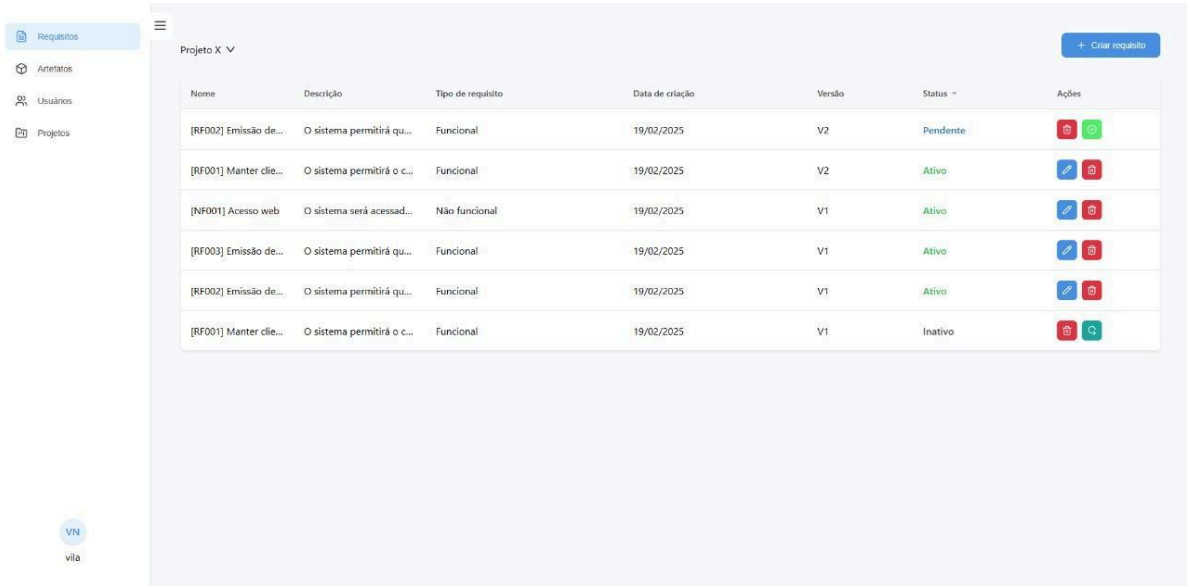
4 RESULTADOS

Nesta Seção são apresentados os resultados obtidos a partir do desenvolvimento do software de gestão de requisitos, destacando as principais telas e sua integração com a API. Os requisitos funcionais e não funcionais foram em sua totalidade implementados no software, assim apresentaremos os requisitos com o foco nas telas, já que é o principal foco do trabalho.

4.1 Requisitos

Ao se logar no sistema e escolher um projeto, o usuário é automaticamente direcionado a página de requisitos, onde estarão listados para ele, todos os requisitos que foram criados para aquele projeto e com a possibilidade de criação de novos.

Figura 4 – Tela de listagem de requisitos

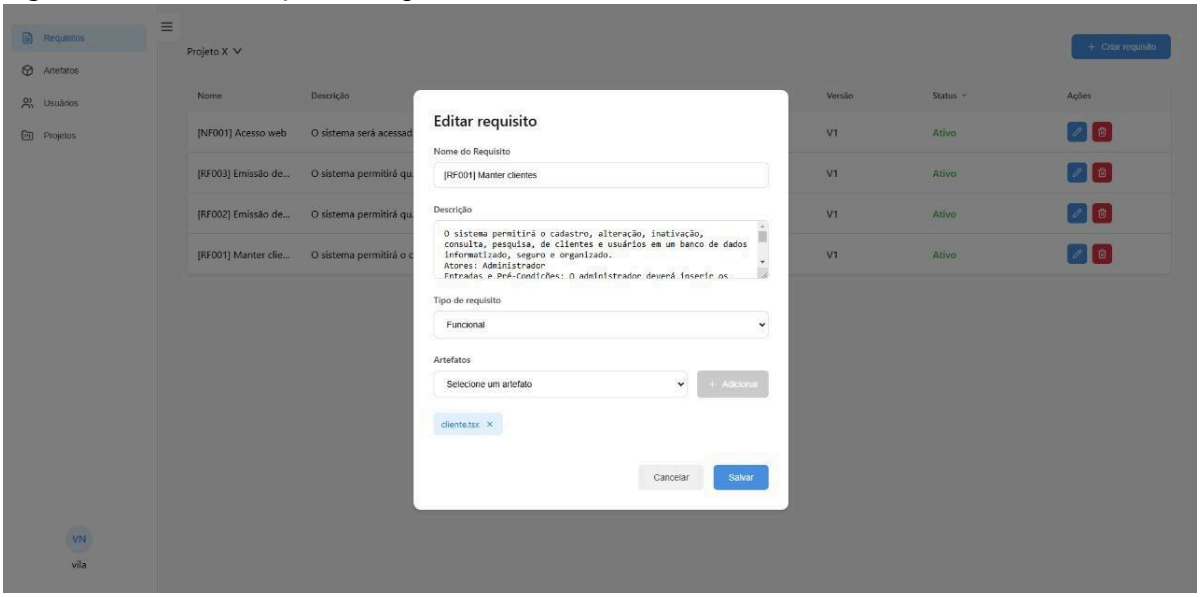


Fonte: elaborada pelo autor.

Os requisitos são definidos como entidades únicas por nome, então só pode existir uma entidade ativa por requisito, logo as demais, serão inativadas. Ao se listar os requisitos, algumas ações podem ser realizadas em cada entidade dependendo de seu status, como:

- Editar: Usuário tem a possibilidade de criar outra versão de um requisito utilizando de seus mesmo dados, ao criar a nova versão, o status fica pendente de aprovação ou reprovação;

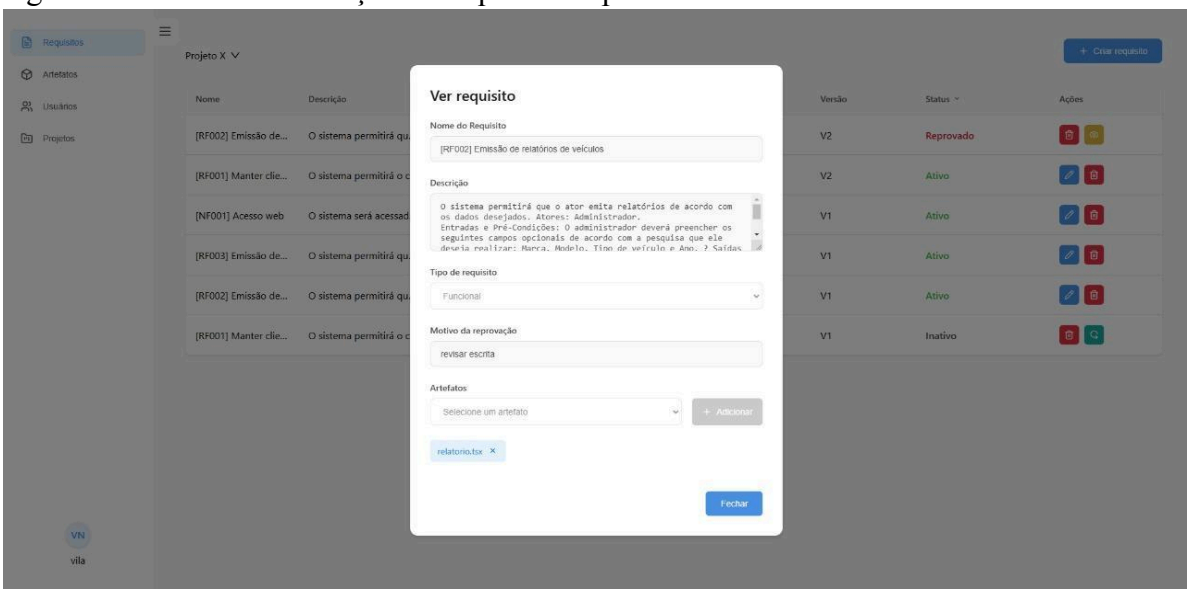
Figura 5 – Tela de edição de requisitos



Fonte: elaborada pelo autor.

- Deletar: Usuário tem a possibilidade de deletar um requisito;
- Visualizar: Para entidade que foram reprovadas, elas conseguem ser visualizadas a qualquer momento, trazendo em seus dados, o motivo da reprovação;

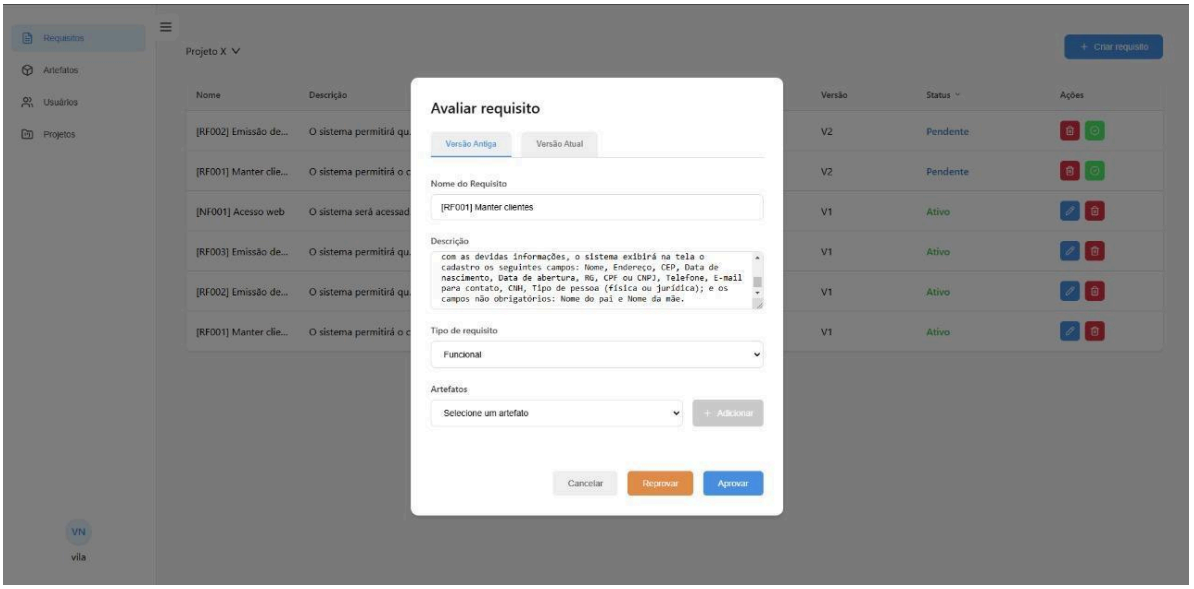
Figura 6 – Tela de visualização de requisitos reprovados



Fonte: elaborada pelo autor.

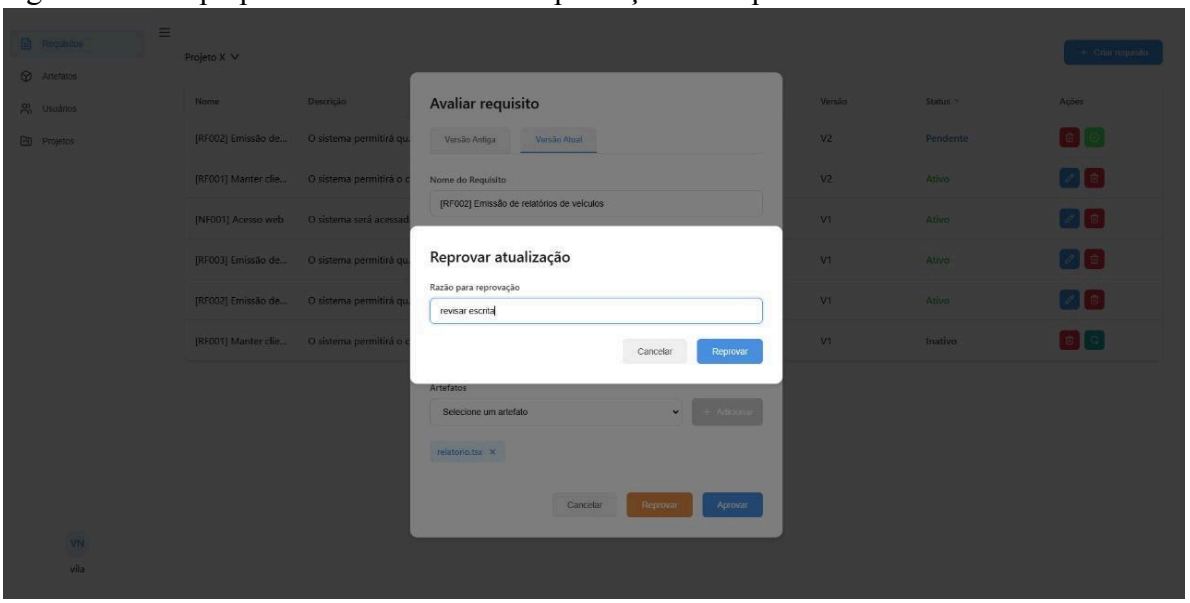
- Aprovar/Reprovar: Quando uma entidade está com status pendente, o usuário consegue aprovar ou reprovar a solicitação de atualização de versão da mesma, tornando aquela entidade como a versão ativa, caso reprove, um motivo para reprovação será requisitado e a solicitação de atualização requisito será reprovada.

Figura 7 – Tela de aprovação/reprovação de um requisito



Fonte: elaborada pelo autor.

Figura 8 – Campo para inserir motivo de reprovação de requisito



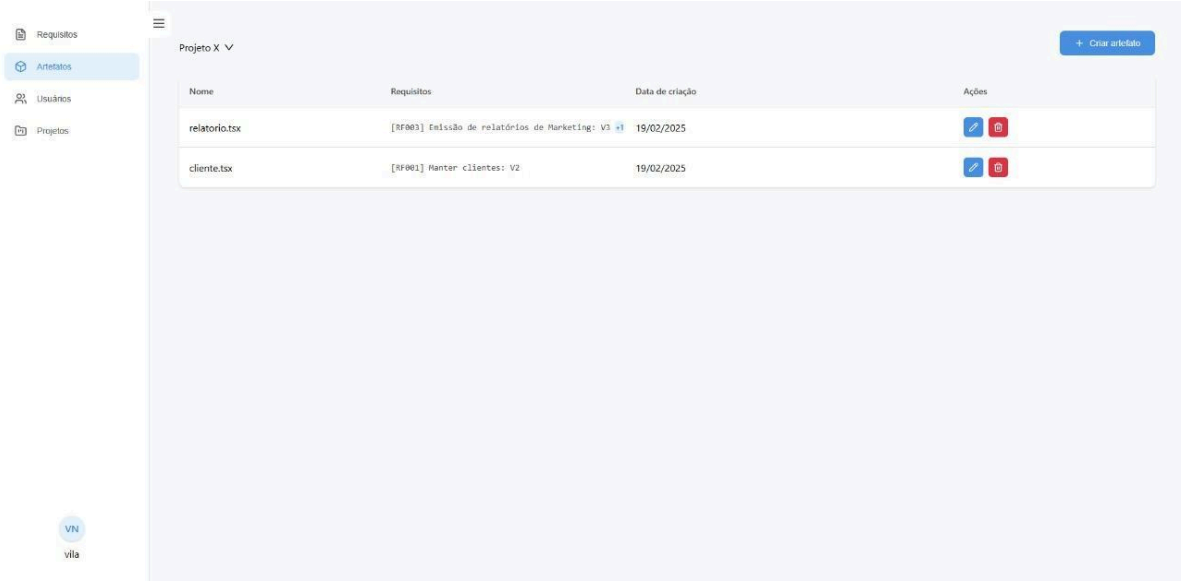
Fonte: elaborada pelo autor.

- Reativar: Para entidades com status inativas, versões anteriores, existe a possibilidade de ativá-las novamente, tornando essas entidades a versão atual do requisito.

4.2 Artefatos

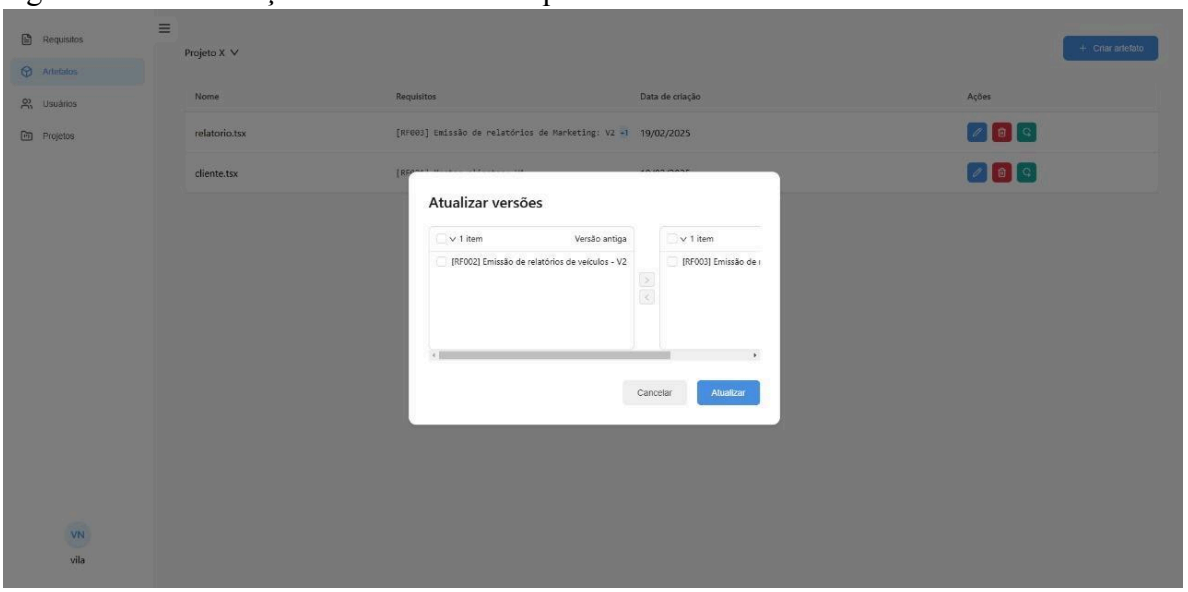
Artefatos são entidades que podem ser associadas a um ou mais requisitos, sua estrutura se baseia basicamente por name. Ao se criar ou aprovar requisitos, são adicionados dados em uma tabela de relacionamento entre requisitos e artefatos a versão daquele requisito, logo, cada artefato possui uma lista de requisitos junto com a sua versão, o que pode ser observado na Figura 9. Cada artefato possui a possibilidade de edição e deleção, caso um requisito tenha sua versão atualizada, uma opção de atualizar os requisitos de um artefato fica disponível, tendo a liberdade de escolher quais requisitos terão quais versões para aquele artefato como mostrado na Figura 10.

Figura 9 – Listagem de artefatos



Fonte: elaborada pelo autor.

Figura 10 – Atualização das versões de requisitos de um artefato



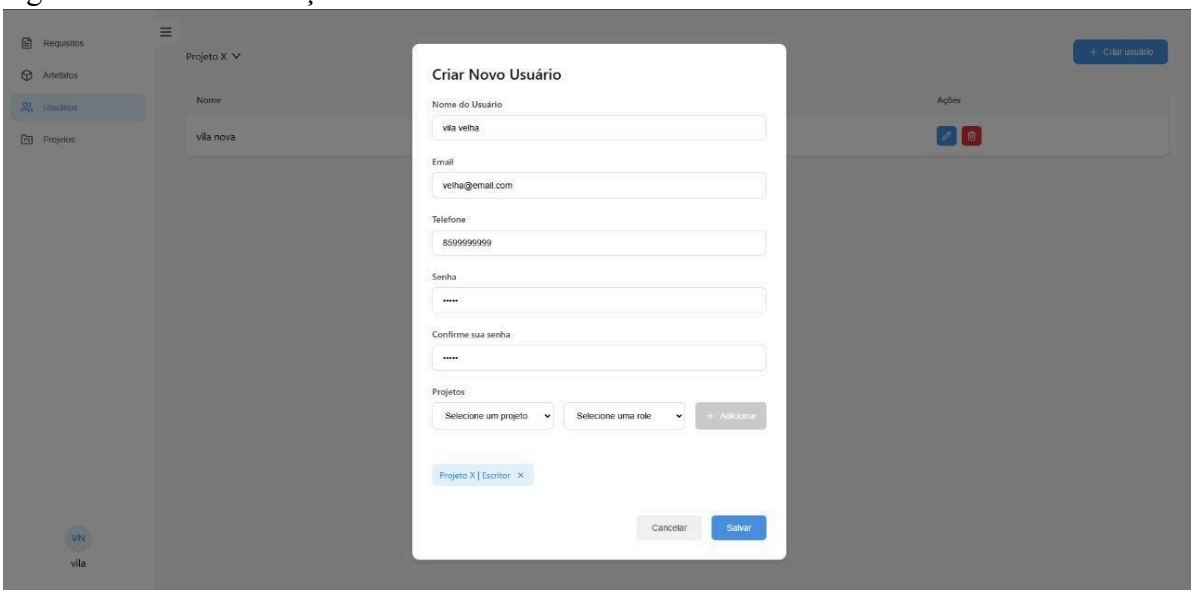
Fonte: elaborada pelo autor.

4.3 Usuários

A página de usuários é responsável por exibir e manter os dados de usuário, podendo ser realizado a criação de novos usuários, editando antigos e podendo também serem deletadas. Para a criação de usuários, são definidos os projetos daquele usuário e suas permissões, para cada projeto, o novo usuário possui uma permissão específica, alimentando uma tabela de relação de usuários e projetos, tais permissões de usuário, podem ser:

- Administrador: Possui acesso total ao sistema, incluindo a criação, edição e remoção de usuários, requisitos, artefatos e projetos.
- Gerente de Projetos: Pode criar, editar e gerenciar requisitos e artefatos dentro dos projetos que administra.
- Usuário escritor: Pode visualizar e gerenciar requisitos e artefatos conforme suas permissões dentro dos projetos.

Figura 11 – Tela de criação de usuários

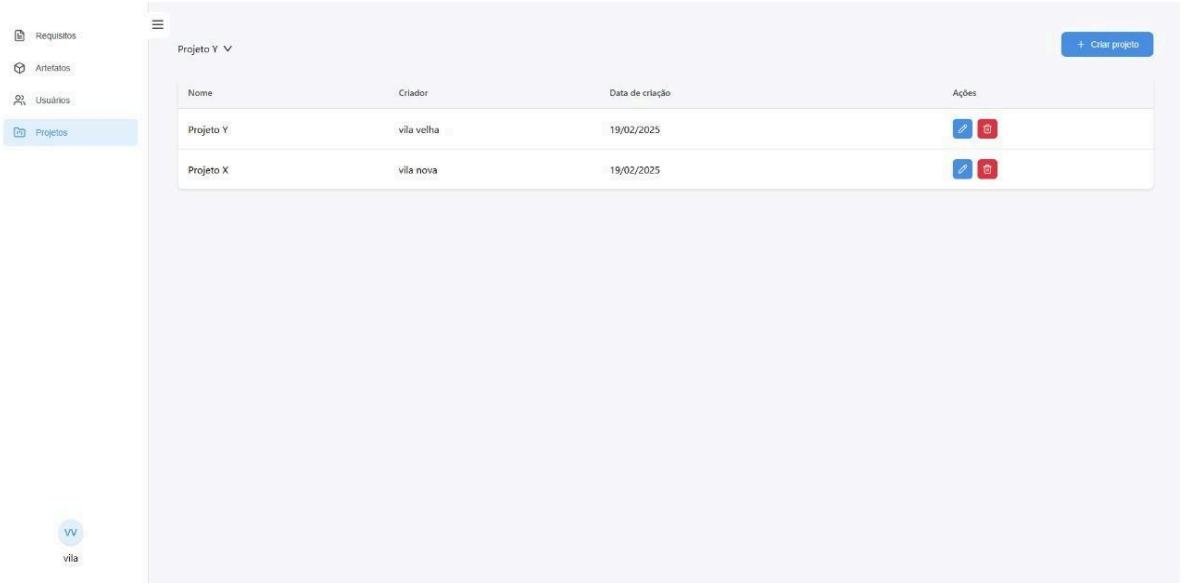


Fonte: elaborada pelo autor.

4.4 Projetos

A página de projetos tem por finalidade, criar entidades e criar escopos para os dados criados dentro desses projetos, então para cada entidade criada em um projeto, seja ela requisito, artefato ou usuário, ela será apenas exibida para aquele usuário, caso ele esteja naquele projeto em específico.

Figura 12 – Tela de listagem de projetos



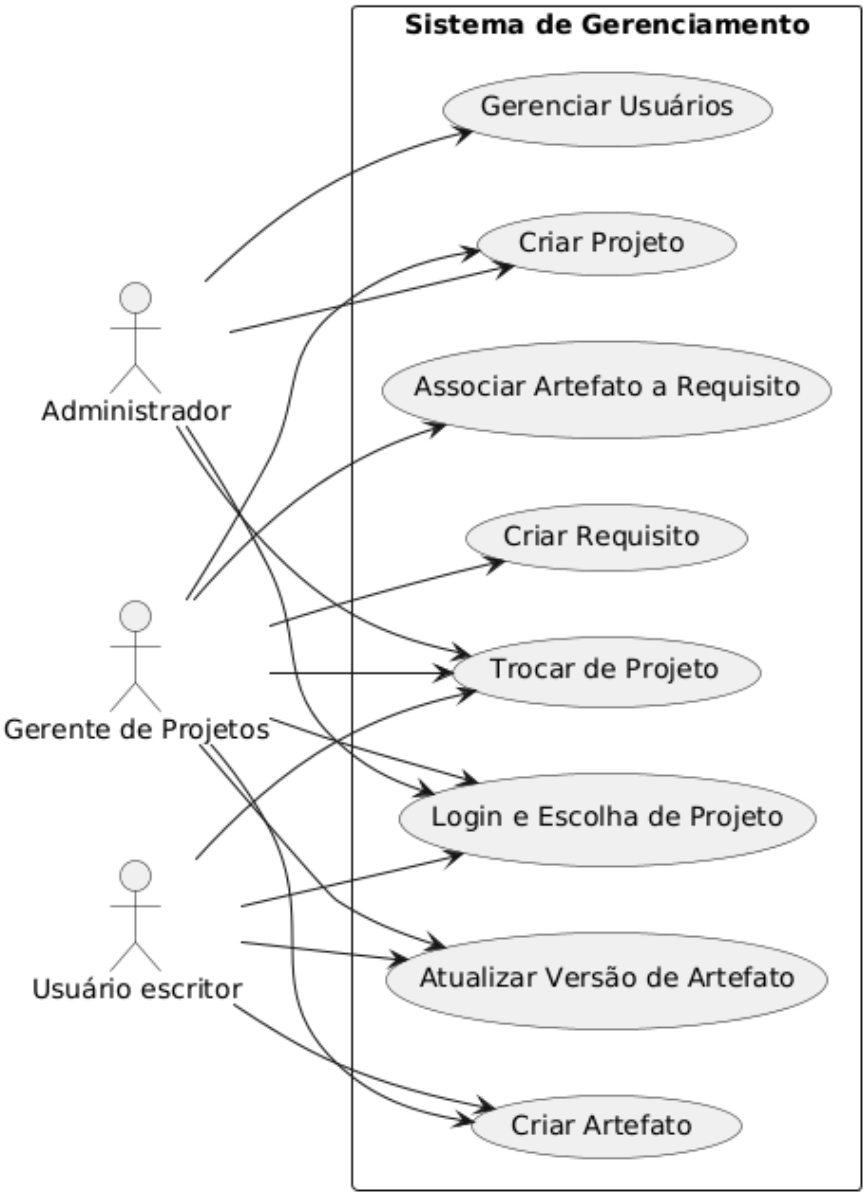
Fonte: elaborada pelo autor

4.5 Casos de uso

O sistema de gerenciamento de requisitos e seus artefatos, tem como principal objetivo manter os dados de um requisito sempre atualizado, permitindo sua rastreabilidade, manutenção e acompanhamento dos requisitos durante seu ciclo de vida. Além de permitir a criação e manutenção de requisitos e artefatos, também possibilita a criação, edição e deleção de usuários e projetos, permitindo um maior controle de versões e dados sobre toda a plataforma com um maior gerenciamento de acessos.

Abaixo, serão apresentados alguns casos de uso, sobre as ações que podem ser realizadas dentro da plataforma, de forma a especificar o que pode ser feito.

Figura 13 – Caso de uso do sistema



Fonte: elaborada pelo autor

4.5.1 Casos de uso: Criar um requisito

Ator Principal: Gerente de Projetos

Fluxo Principal:

1. O usuário acessa a funcionalidade de criação de requisitos;
2. Insere os detalhes do requisito (nome, descrição, tipo e artefatos relacionados);
3. Confirma a criação;
4. O sistema valida se já existe um requisito com o mesmo nome ativo;
5. O sistema salva o requisito e o associa aos artefatos indicados.

4.5.2 Casos de uso: Criar um artefato

Ator Principal: Gerente de Projetos ou Usuário Comum

Fluxo Principal:

1. O usuário acessa a funcionalidade de criação de artefatos;
2. Insere os detalhes do artefato (nome, descrição, requisitos relacionados);
3. Confirma a criação;
4. O sistema salva o artefato e o associa aos requisitos indicados.

4.5.3 Casos de uso: Gerenciar Usuários

Ator Principal: Administrador

Fluxo Principal:

1. O administrador acessa a funcionalidade de gerenciamento de usuários;
2. Insere os dados do novo usuário (nome, email, telefone, senha e projetos);
3. Define permissões para o usuário;
4. Confirma a criação;
5. O sistema salva o usuário e associa aos projetos e permissões concedidas.

4.5.4 Casos de uso: Criar Projeto

Ator Principal: Administrador ou Gerente de Projetos

Fluxo Principal:

1. O usuário acessa a funcionalidade de criação de projeto.

2. Insere o nome do projeto.
3. Confirma a criação.
4. O sistema cria o projeto e o adiciona à lista de projetos disponíveis.

4.5.5 Casos de uso: Login e Escolha de Projeto

Ator Principal: Todos os usuários

Fluxo Principal:

1. O usuário acessa a página de login.
2. Insere as credenciais (email e senha).
3. O sistema valida as credenciais e gera um token JWT.
4. O sistema exibe a lista de projetos associados ao usuário.
5. O usuário seleciona um projeto para iniciar a navegação no sistema.

5 CONCLUSÃO E TRABALHOS FUTUROS

Por meio do desenvolvimento do software para gerenciamento de requisitos, foi possível perceber como a análise e gestão de requisitos podem ser potencializadas com o auxílio de uma ferramenta automatizada. A solução desenvolvida atende de maneira eficiente e eficaz os requisitos estabelecidos pelo professor orientador, considerando funcionalidades essenciais para a avaliação de requisitos e seus artefatos. Além disso, a ferramenta oferece funcionalidades adicionais que tornam a experiência de escrita de novos requisitos mais fluida e incentivam a colaboração entre os envolvidos no processo.

Durante o desenvolvimento do software, foi realizada uma análise de negócio detalhada para coletar os requisitos necessários, permitindo a construção da estrutura e da arquitetura do sistema. A implementação ocorreu em ciclos iterativos, seguindo o padrão SCRUM, garantindo uma abordagem ágil e organizada. Como resultado, o sistema final apresenta uma linguagem moderna e de fácil compreensão, facilitando sua adoção e utilização. Além disso, ao longo do projeto, foram adquiridos conhecimentos valiosos sobre a importância da gestão de requisitos, o impacto de uma boa documentação em um projeto e conceitos fundamentais para o desenvolvimento utilizando Node.js e metodologias ágeis.

Com os resultados obtidos, identificou-se a necessidade de trabalhos futuros para aprimorar ainda mais a aplicação, principalmente em aspectos de segurança, como a implementação de um sistema mais robusto para gerenciamento de usuários e proteção de dados. Também se considera essencial a realização de testes no campus da Universidade Federal do Ceará, em Russas, envolvendo alunos e professores, a fim de coletar *feedbacks* sobre o uso cotidiano da aplicação e identificar possíveis melhorias.

Dessa forma, o software desenvolvido demonstrou ser uma ferramenta eficiente para otimizar um processo que antes era manual e monótono, facilitando a gestão de requisitos. Espera-se que, com o uso do sistema, a experiência dos usuários seja mais intuitiva e produtiva, reduzindo o tempo e esforço despendidos na criação e manutenção de requisitos. Além disso, o aprendizado adquirido ao longo do projeto será fundamental para futuras melhorias da plataforma, possibilitando sua expansão para um público mais amplo. Assim, acredita-se que este trabalho contribui significativamente para a automação e otimização da gestão de requisitos, beneficiando tanto o meio acadêmico quanto outros setores interessados nessa abordagem.

REFERÊNCIAS

AZEVEDO, Fernando Luiz Ferreira de. **Definição e aplicação de um processo para gerenciamento de requisitos a sistemas aeroespaciais**. 2017. Dissertação (Mestrado em Engenharia e Gerenciamento de Sistemas Espaciais) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2017. Disponível em: http://urlib.net/8JMKD3MGP3W34P/3P27HKH. Acesso em: 18 fev. 2025.

BADOCO, Maurício; ALMEIDA, Renato Inácio de; LUCAS, Carlos Alberto de. Metodologia ágil na gestão de projetos de software. **Revista EduFatec: educação, tecnologia e gestão**, Franca, v. 1, n. 1, p. 1-18, jan./jun. 2018.

BARROS, Ricardo Correia. **A Importância da Gestão de Requisitos para Projetos de Desenvolvimento de Software.** 2018. Tese de Doutorado. BS Thesis, Campus São Paulo (IFSP), Instituto Federal de Educação, Ciência e Tecnologia, São Paulo, Brasil.

BODUCH, Adam; DERKS, Roy. **React and React Native: A complete hands-on guide to modern web and mobile development with React. js**. Packt Publishing Ltd, 2020.

BITTAR, Thiago; LOBATO, Luanna; SANTOS, Larissa; SILVA, Patrícia. **Experiências usando PERT-CPM para desenvolvimento ágil de software**. In: ESCOLA REGIONAL DE INFORMÁTICA DE GOIÁS (ERI-GO), 2018, Goiânia. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2018. p. 177-190.

CASCIARO, Mario; MAMMINO, Luciano. **Node. js Design Patterns: Design and implement production-grade Node. js applications using proven patterns and techniques.** Packt Publishing Ltd, 2020.

FAGUNDES, Priscila Basto. **FIRMa**: Uma proposta baseada nos instrumentos utilizados pela gestão da informação para auxiliar o processo de gestão de requisitos de software. 2021. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/226834/PCIN0262-T.pdf?sequence=1&isAllowed=y. Acesso em: 26 fev. 2025.

FERNANDES, João M.; MACHADO, Ricardo J. **Requisitos em projetos de software e de sistemas de informação**. Novatec Editora, 2017.

GASPARDINE JÚNIOR, Anderson Heraclito; SOBRINHO, Rita de Cassia Ramos. **Análise da influência da qualidade de requisitos na construção de matrizes de rastreabilidade de software**. 2022.

LOTTIN, Poline; DALFOVO, Oscar. **Estudo de caso para gestão de requisitos utilizando a ferramenta doors ng aplicado em uma empresa de softhouse.** 10.13140/RG.2.1.2815.8328. 2016.

MUNIZ, Luis Fernando Lopes. **Zeroc**: uma plataforma para gerenciamento de submissão de artigos para os Encontros Universitários da UFC - Campus Russas. 2023. Trabalho de Conclusão de Curso (Graduação em Engenharia de Software) - Campus de Russas, Universidade Federal do Ceará, Russas, 2023.

ORTU, Marco; DESTEFANIS, Giuseppe; ADAMS, Bram; MURGIA, Alessandro; MARCHESI, Michele; TONELLI, Roberto.**The jira repository dataset**: Understanding social aspects of software development. In: Proceedings of the 11th international conference on predictive models and data analytics in software engineering. 2015. p. 1-4.

GARRIDO, Anabel Pilicita; LÓPEZ, Yolanda Borja; CONSTANTE, Gonzalo Gutiérrez. **Rendimiento de MariaDB y PostgreSQL**. Revista Científica y Tecnológica UPSE (RCTU), v. 7, n. 2, p. 9-16, 2020.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software-9**. McGraw Hill Brasil, 2021.

SANTOS, Jorge Almeida dos. **O gerenciamento de requisitos e a sua importância em projetos de desenvolvimento de software**. 2015. Disponível em: https://repositorio.ufu.br/handle/123456789/19409 .

SIDDIQUI, Shams; AHMAD, Md; SHUAIB, Mohammed. **Cloud based requirements management tools should be selected on the basis of project requirements.** Int J Comput Sci Eng , v. 8, n. 10, p. 83-88, 2020.

SILVA, Matheus Rodrigues Rosado da. **Projeto e desenvolvimento de um sistema para gerenciamento de trabalhos de conclusão de curso**. 2017. 36 f. Trabalho de Conclusão de Curso (Graduação em Sistemas de Informação) – Universidade Federal de Uberlândia, Uberlândia, 2017.

SOARES, Silviane Lawall; WERLANG, Ricardo; MOESCH, Beatriz; OZORIO, Jhon Lenon; HOLSCHER, Matheus. **Experiência de gestão de equipes de educação a distância através da metodologia ágil Scrum**. Revista Connect EAD, [S.l.], v. 1, n. 1, p. 43-57, abr. 2018. Disponível em: https://uceff.edu.br/revista/index.php/connectead/article/view/266. Acesso em: 17 fev. 2025.

SOMMERVILLE, Ian. **Engenharia de software (ed.)**. América: Pearson Education Inc, 2011.

VARGAS, Letícia Marques. **Gerenciamento Ágil de Projetos em Desenvolvimento de Software: um estudo comparativo sobre a aplicabilidade do Scrum em conjunto com PMBOK e/ou PRINCE2**. Gestão e Projetos: GeP, v. 7, n. 3, p. 48-60, 2016.

VARGAS, L. M. Project Agile Management for Software Development: A Comparative Study on the Applicability of Scrum Together with Pmbok and / or Prince2. **Revista de Gestão e Projetos**, v. 7, n. 3, p. 48-60, set./dez. 2016. ISSN 2236-0972.

VAZQUEZ, Carlos Eduardo; SIMÕES, Guilherme Siqueira. **Engenharia de Requisitos**: software orientado ao negócio. Brasport, 2016.

WILTGEN, Filipe. Projetos Baseados em Requisitos. **Revista de Engenharia e Tecnologia,** v. 14, n. 1, 2022.