Universidade de Brasília - UnB
Faculdade do Gama - FGA
Engenharia de Software

iFlow: Ferramenta para Apoio ao Processo de Engenharia de Requisitos

Autores: Lucas Fellipe Carvalho Moreira
Rogério Soares Caixeta
Orientadora: Profa. Dra. Milene Serrano

Brasília, DF
2022

Lucas Fellipe Carvalho Moreira
Rogério Soares Caixeta

iFlow: Ferramenta para Apoio ao Processo de Engenharia de Requisitos

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade do Gama - FGA

Orientador: Profa. Dra. Milene Serrano

Coorientador: Prof. Dr. Maurício Serrano

Brasília, DF

2022

Lucas Fellipe Carvalho Moreira
Rogério Soares Caixeta

iFlow: Ferramenta para Apoio ao Processo de Engenharia de Requisitos/ Lucas Fellipe Carvalho Moreira e Rogério Silva dos Santos Júnior. – Brasília, DF, 2022-
136 p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Milene Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade do Gama - FGA , 2022.

1. Engenharia de Requisitos. 2. Requisitos de Software. I. Profa. Dra. Milene Serrano. II. Universidade de Brasília. III. Faculdade do Gama. IV. iFlow: Ferramenta para Apoio ao Processo de Engenharia de Requisitos

CDU 02:141:005.6

Lucas Fellipe Carvalho Moreira
Rogério Soares Caixeta

iFlow: Ferramenta para Apoio ao Processo de Engenharia de Requisitos

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 27 de Setembro de 2022:

Profa. Dra. Milene Serrano
Orientadora

Prof. Dr. Maurício Serrano
Coorientador

Prof. Dr. Edson Alves da Costa Júnior
Examinador 1

Prof. Dr. Renato Coral Sampaio
Examinador 2

Brasília, DF
2022

Este trabalho é dedicado às nossas famílias, à nossa fonte de amor, suporte e esperança, sem as quais este trabalho não seria possível. Vocês foram e são a luz que nos guiam a cada dia e nos dão forças para ir sempre além.

Agradecimentos

Primeiramente, gostaríamos de agradecer à Deus, que nos proporcionou a graça de estarmos aqui hoje concluindo a nossa graduação, em meio a um contexto onde tantos não tiveram a mesma oportunidade. Agradecemos pela inteligência e pela força que nos proporcionou para trilhar esse caminho.

Agradecemos a todos os professores da Faculdade UnB Gama que não medem esforços para nos transmitir e proporcionar o melhor de si, para termos o melhor aprendizado possível. Sabemos que tivemos uma grande oportunidade na Universidade de Brasília, e que hoje saímos mais humanos e capazes de enfrentar os desafios que vierem pela frente.

Gostaríamos de deixar um agradecimento em especial aos nossos orientadores e professores da banca, verdadeiros exemplos de profissionais. Vocês sempre nos conduziram para que déssemos o nosso melhor, não mediram esforços para nos entregar a melhor educação possível, e nos inspiraram para podermos, algum dia, ser profissionais tão incríveis quanto vocês. A dedicação de vocês nos impressiona e nos motiva a sempre querer conquistar o próximo nível. O nosso mais sincero obrigado.

Aproveitamos a oportunidade de deixar o mais sincero agradecimento à professora Milene Serrano, o terceiro membro desse trabalho. Você nunca mediu esforços em nos auxiliar e nos dar simplesmente o seu melhor. Muito obrigado pelas suas correções impecáveis, nas quais não media esforços para nos orientar no melhor possível. Você é um exemplo de dedicação e profissional, e nós agradecemos por termos tido a honra e o privilégio de termos sido os seus alunos.

Eu, Rogério, gostaria de agradecer às duas pessoas mais importantes da minha vida que já se foram, a minha Mãe e minha Avó. Foram elas que me ensinaram tudo o que sei, mostrando-me os caminhos que me permitiram chegar até aqui. Deram-me o incentivo e a sabedoria para enfrentar todas as adversidades e continuar fiel ao caminho correto. Se hoje estou aqui, prestes a me tornar um Engenheiro de *Software*, foi graças a determinação e carinho de vocês. A minha mais profunda gratidão por ter tido a grande oportunidade de ser seu filho e neto.

Gostaria ainda de agradecer à minha família, que sempre me incentiva a dar o meu melhor. Vocês nunca deixaram de me incentivar e dar forças para eu poder concluir

mais essa etapa da minha carreira profissional. O meu mais sincero agradecimento a vocês.

Eu, Lucas, gostaria de agradecer, primeiramente, à minha Mãe e à minha Avó, que sempre estiveram comigo e nunca mediram esforços para me dar a melhor educação e o maior amor possível. Gostaria de agradecer ao meu Irmão, que sempre me ensinou e sempre me inspirou, além de ser uma referência para mim. Gostaria de agradecer ao meu Pai, que sempre confiou em mim e que também não mediu esforços para me dar a melhor educação. Gostaria, também, de agradecer à Minha Namorada, que não me deixou desistir do curso e sempre esteve comigo nos meus momentos mais difíceis. Sem vocês, eu não teria chegado aqui. Meu sentimento é de gratidão. O meu mais sincero agradecimento a vocês.

"Cada sonho que você deixa para trás,
é um pedaço do seu futuro que deixa de existir."
(Steve Jobs)

Resumo

A Engenharia de Requisitos é um processo essencial no desenvolvimento de um produto de *software*, possuindo diversas etapas a serem seguidas, e sendo fundamental para a elaboração do produto. Dada a relevância da área, e as diferentes vertentes de atuação demandadas para cumprimento de suas atividades, os Engenheiros de Requisitos e profissionais afins revelam dificuldades, portanto, necessidades de recursos que os auxiliem. Visando corroborar nesse contexto, esse trabalho propõe um apoio guiado, chamado *iFlow*, orientado à literatura especializada, que procura automatizar parte desse processo. O *iFlow* é um *software* web desenvolvido em *NodeJS* e *ReactJS*, direcionado a um público técnico, e centrado nas principais atividades da Engenharia de Requisitos, sendo elas: Pré-rastreabilidade, Elicitação, Modelagem, Análise (foco em Verificação e Priorização), e Pós-Rastreabilidade. A automatização no *iFlow* orienta-se por um processo de planejamento sistemático, conhecido como *House of Quality*, cuja implementação é viabilizada com uso de Grafos, e Algoritmos de Menor Caminho. O iFlow é capaz de gerar uma versão preliminar de um Produto Mínimo Viável.

Palavras-chave: Engenharia de Requisitos. Elicitação. Modelagem. Análise. Rastreabilidade. *House of Quality*. Grafos. Algoritmos de Menor Caminho.

Abstract

Requirements Engineering is an essential part in software product development. It has many steps to be followed and is fundamental in product elaboration. Given the importance of this field, and the different action strands demanded to fulfil the activities, Requirement Engineers and correlated professionals reveal some difficulties and, therefore, require resources for assistance. In this perspective, this paper proposes a guided support, called *iFlow*, based on specialized literature that aims to automatize part of this process. The *iFlow* is a web software developed in NodeJS and ReactJS, directed to a specialized audience, and centered in the main activities of Requirements Engineering: Pre-traceability, Elicitation, Modeling, Analysis (focused in Verification and Prioritization) and Post-traceability. The *iFlow* automatization is guided by a systematic planned process, known as House of Quality, that will be implemented using Graphs, and Single Shortest Path algorithms. The idea is to generate a preliminary version of the Minimum Viable Product.

Key-words: Requirements Engineering. Elicitation. Modeling. Analyze. Traceability. House of Quality. Graphs. Single Shortest Path Algorithm.

Lista de Ilustrações

Lista de Tabelas

Lista de Quadros

Lista de Códigos

Lista de Abreviaturas e Siglas

TI	Tecnologia da Informação
MVP	*Minimum Viable Product* (Produto Mínimo Viável)
BPMN	*Business Process Model and Notation*
BPMI	*Business Process Management Iniative*
DE-R	Diagrama Entidade-Relacionamentos
DL	Diagrama Lógico
TCC1	Trabalho de Conclusão de Curso 1
TCC2	Trabalho de Conclusão de Curso
SAL	Serviço de Ambulância de Londres

Sumário

1 Introdução

Neste capítulo, contextualizaremos a pesquisa realizada neste trabalho, sendo a mesma focada na Engenharia de Requisitos e seus processos. A partir da contextualização (Seção 1.1), será apresentada uma problemática (Seção 1.2) no domínio da Engenharia de Requisitos. A partir da problemática, serão apresentadas as questões de pesquisa (Seção 1.3) que norteiam a revisão da literatura, respondidas ao final do trabalho. Adicionalmente, são descritos os principais objetivos (1.4) atingidos ao longo da pesquisa. Por fim, tem-se a organização desta monografia (Seção 1.5).

1.1 Contextualização

A Engenharia de Requisitos é um ramo da Engenharia de *Software* que se preocupa com os objetivos reais, funções e restrições dos sistemas de *software*, fazendo parte do ciclo de desenvolvimento do *software*, e sendo fundamental para o sucesso geral do sistema desenvolvido. Esse ramo também se preocupa com a relação desses fatores com caracterizações exatas do comportamento do *software* e com a sua evolução temporal (ELLIOTT, 2012).

Desempenhando um papel significativo no desenvolvimento de aplicativos de *software*, a Engenharia de Requisitos fornece uma maneira de entender e descrever os problemas que requerem uma solução de *software*. Isso é relevante para compreender as necessidades dos usuários e especificar tais necessidades como requisitos de *software* (ELLIOTT, 2012). Trata-se de uma área de suma importância, pois é necessária para melhorar as possibilidades de sucesso de um projeto.

A elicitação, a modelagem e a análise (verificação e validação) são processos essenciais na Engenharia de Requisitos, os quais ajudam a cumprir o propósito de um *software*. Além disso, a Engenharia de Requisitos documenta o sistema, identifica todas as partes interessadas e suas preocupações, e apresenta essas informações de uma forma que possam ser analisadas e, eventualmente, implementadas (ELLIOTT, 2012).

1.2 Problemática

Um dos grandes casos estudados na área de Engenharia de Requisitos é o sistema de *software* desenvolvido para a automatização do Serviço de Ambulância de Londres (SAL). Este serviço tinha uma taxa de 2000 a 2500 ligações diárias, com uma frota de 750 veículos e área coberta de cerca de 1600 km². Foi operado manualmente até o ano de 1987, quando o sistema foi posto em funcionamento (ADAMU et al., 2010).

O sistema ficou em operação por menos de 10 dias, e se mostrou totalmente ineficiente para as demandas do serviço que propunha automatizar. Foi causa da morte de cerca de 30 pessoas; gerou um prejuízo em torno de £1.5 milhões; casou a resignação do chefe executivo do SAL; instaurou um inquérito na equipe que desenvolveu o produto, e deixou em voga a responsabilização ética para profissionais de TI, bem como para outras áreas, dentre elas: direito e medicina (ADAMU et al., 2010).

Os principais pontos de erros levantados no trabalho de (ADAMU et al., 2010), considerando os problemas de elicitação do projeto, foram:

- Prolemas na definição do cronograma de execução do projeto, dado que foi definido um prazo irreal para o desenvolvimento do sistema, treinamento de pessoal e testes do *software*;
- O comitê responsável teve um papel mínimo no desenvolvimento do sistema, visto que, metade das pessoas ficaram responsáveis por grande parte do trabalho, e não consideraram a visão dos funcionários que trabalhavam na ambulância. Isso levou a um cenário de forte introspecção, com poucos pontos de vista do sistema, criando algo que não atendia as necessidades e também não considerava o contexto inserido;
- Não considerou situações de erro na captação de dados da localização das ambulâncias ou mesmo de recursos necessários para que o sistema responsável por enviar ambulâncias em locais de incidentes pudesse funcionar corretamente. Em períodos com sobrecarga de chamados, o sistema colapsava e levava bastante tempo para executar;
- Realizaram o treinamento inadequado, de forma que, os funcionários tiveram que se adaptar e compreender um sistema que ainda não estava totalmente pronto. Vale ressaltar que o *software* teve uma série de modificações antes de ser finalizado, e

- O sistema foi testado, apenas, no contexto de cada módulo, ou seja, em separado. Contudo, não foi testada a integração dos módulos, assim como não realizados os *testes de estresse*, adequados no contexto de uma aplicação tão crítica.

A partir do exposto, nota-se o quão importante é uma Engenharia de Requisitos elaborada adequadamente, assim como o quanto está diretamente associada ao sucesso do sistema desenvolvido. Impacta diretamente nos custos e no resultado do produto. Corroborando com esse ponto, há um estudo realizado por Donald Firesmith, mostrando que as empresas americanas têm um prejuízo acima de 30 bilhões de dólares por ano, acarretando projetos fracassados (KING; MARASCO, 2008).

1.3 Questão de Pesquisa

Para guiar as pequisas e o desenvolvimento deste estudo, foram definidas as seguintes indagações:

1. Como podemos desenvolver uma ferramenta que apoie o Engenheiro de Requisitos na sintetização de um produto de *software* abstrato para um que seja concreto?;
2. Como podemos facilitar o desenvolvimento criativo do Engenheiro de Requisitos, de tal forma que a rastreabilidade das fontes dos requisitos sejam mantidas?, e
3. Tendo posse dos requisitos amadurecidos e sua importância para o produto de *software*, como podemos correlacionar esses dados para gerar um possível *Minimum Viable Product* (MVP)?.

1.4 Objetivos

1.4.1 Objetivo Geral

Desenvolver uma ferramenta que viabilize automatizações e auxilie no processo de Engenharia de Requisitos, para embasar a construção de uma aplicação e mitigar retrabalhos e desperdícios de tempo. A ferramenta funciona conferindo ênfase nas etapas da Engenharia de Requisitos, de forma que, uma vez concluída uma etapa, a próxima é desbloqueada, permitindo que o usuário volte em etapas anteriores para que haja um refinamento da mesma.

1.4.2 Objetivos Específicos

No intuito de cumprir com o objetivo geral deste trabalho, foram planejados e concluídos os seguintes objetivos específicos:

1. Automatizar a *hiperlinkagem* entre os artefatos de requisitos;
2. Viabilizar a geração do MVP (versão preliminar) com base nos artefatos gerados;
3. Prover uma *interface* para a ferramenta que guie o usuário no processo de criação de alguns artefatos de requisitos pré-definidos;
4. Viabilizar a criação da *interface*, mencionada no item anterior, de modo a proporcionar uma experiência de usuário satisfatória, visando facilitar o processo da Engenharia de Requisitos, e
5. Realizar uma primeira análise dos resultados obtidos, usando como base uma amostra do público alvo.

Adicionalmente aos objetivos específicos listados, embasou-se a pesquisa considerando a literatura especializada, bem como se conduziu a pesquisa orientando-se por uma metodologia adequada. Portanto, foram realizadas atividade de levantamento bibliográfico e definição de uma metodologia, ambas acordadas no Capítulo de Metodologia (Capítulo 4).

1.5 Organização da Monografia

A monografia está estruturada conforme os seguintes capítulos:

- Capítulo 2 - Embasamento Teórico: apresenta o que foi encontrado na literatura, no que permeia e oferece apoio, em termos conceituais, ao tema do trabalho;
- Capitulo 3 - Referencial Tecnológico: descreve as tecnologias que foram utilizadas para o desenvolvimento do trabalho na totalidade;
- Capítulo 4 - Metodologia: define como foi realizada a condução do trabalho, desde a pesquisa, até a construção da ferramenta e a análise dos resultados;

- Capítulo 5 - *iFlow*: descreve o produto final deste trabalho, no caso, a ferramenta *iFlow*, que consiste em gerar um Produto Mínimo Viável, definindo sua estrutura e dispondo de artefatos e provas de conceito;
- Capítulo 6 - Análise de Resultados: apresenta a análise de resultados coletados durante o desenvolvimento do trabalho, sendo esses analisados, e
- Capítulo 7 - Conclusão: apresenta as considerações finais deste projeto, bem como propostas para trabalhos futuros.

2 Embasamento Teórico

Neste capítulo, são apresentados os principais conceitos utilizados para embasamento da ferramenta desenvolvida, sendo essa focada na Engenharia de Requisitos e suas principais atividades/etapas. Primeiramente, terá uma seção descrevendo, de forma mais conceitual e abrangente, a Engenharia de Requisitos (Seção 2.1). Logo em seguida, será apresentado o tópico de Rastreabilidade (Seção 2.2), definindo conceitos importantes no domínio da Engenharia de Requisitos. Serão apresentadas as técnicas referentes ao tópico de Elicitação (Seção 2.3). Adicionalmente, será apresentada uma seção sobre Modelagem (Seção 2.4), definindo os principais modelos para a ferramenta. Posteriormente, têm-se os tópicos de Verificação (Seção 2.5) e Priorização (Seção 2.6). Logo em seguida, têm-se as definições de *House of Quality* (Seção 2.7), bem como as definições de Grafos, fundamentais para a ferramenta. Por fim, tem-se o resumo do capítulo (Seção 2.8). Como há um vasto conjunto de conceitos, a Figura 1 confere um roteiro de leitura, procurando auxiliar na compreensão do conteúdo exposto.

A macro área de interesse deste trabalho é a Engenharia de *Software*. A ferramenta desenvolvida compreende, como principal ponto, contribuições no escopo da Engenharia de Requisitos, que será mais bem detalhada ainda nesse capítulo, mas que trata da etapa que define concretamente os requisitos que vão compor um produto de *Software*.

Nesse contexto, para a ferramenta desenvolvida, foram escolhidas algumas etapas de maior relevância para que se tenha um processo mais certeiro na elaboração de um novo produto de *software* com qualidade e os artefatos (documentos) usados para tanto. São eles:

- Pré-rastreabilidade: para viabilizar esta atividade, podem ser utilizados diferentes artefatos, sendo sugeridos, no contexto desse trabalho: *Rich Picture* (Seção 2.2.1.2) e *5W2H* (Seção 2.2.1.1). Esta etapa viabiliza um olhar mais centrado no que tange o contexto de uma aplicação, permitindo esboçar as primeiras ideias em artefatos de cunho mais generalista, e livres de notação formal. Adicionalmente, permite manter um elo, por exemplo, para as fontes de informação;
- Elicitação: para condução desta atividade, há técnicas bem estabelecidas, tais como: *Brainstorming* (Seção 2.3.1), Questionário (Seção 2.3.2), Prototipação (Se-

Figura 1 – Infográfico do Processo da Ferramenta

Fonte: Autores, 2022.

ção 2.3.3), *Storytelling* (Seção 2.3.4), Entrevista (Seção 2.3.5) e Análise de Protocolo (Seção 2.3.6). Neste trabalho, são apresentadas breves definições sobre cada técnica. A partir das técnicas propostas, o principal objetivo é levantar as funcionalidades (requisitos funcionais e não funcionais) de uma aplicação de *software*;

- Modelagem: olhar mais centrado em fazer com que as ideias levantadas na etapa anterior tenham um aspecto mais maduro e que possam ser, de fato, implementado em código. Nesse contexto, existem vários níveis e propostas de modelagem. Entretanto, há ênfase em alguns artefatos, os quais corroboram com a modelagem em diferentes aspectos de um *software*. O *Backlog* do Produto (Seção 2.4.1) permite especificar os requisitos funcionais de um *software*, sendo um artefato muito conhecido e orientado às metodologias ágeis. O *Softgoal Interdependence Graph* (SIG) compreende uma modelagem centrada em requisitos não funcionais, sendo proposta em um *framework* conceitual conhecido como NFR *Framework* (Seção 2.4.2). Além disso, os Léxicos (Seção 2.4.3) são relevantes para compreensão dos

principais termos de um produto de *software*, sejam esses de cunho mais técnico ou de um domínio particular;

- Verificação: Para realização desta atividade, deve-se ter em mente as práticas de Verificação e Validação (Seção 2.5). Esta etapa visa encontrar erros e inconsistências nos modelos e requisitos levantados;
- Pós-rastreabilidade: O olhar desta atividade concentra-se em manter elos entre requisitos e artefatos de níveis mais baixos de abstração (Arquitetura e Código). Nesse ponto, o principal objetivo, é prover um rastro do requisito desde a sua primeira concepção, e
- Priorização/*House of Quality* (Seção 2.6 e 2.7): etapa que visa definir qual a ordem que os requisitos devem ser implementados, que gere maior valor para os seus *Stakeholders*. Acorda uma modelagem gráfica, que tem em vista correlacionar as preocupações entre diferentes *Stakeholders*, sendo muito útil para lidar com interesses conflitantes e visualizar as funcionalidades que são mais relevantes, dados os aspectos qualitativos (requisitos não funcionais) do *software*.

Cabe colocar que a ferramenta desenvolvida orienta o desenvolvimento de um Produto Mínimo Viável de *software*, considerando a Engenharia de Requisitos e suas atividades. Neste sentido, algumas automatizações são realizadas, demandando estabelecer e se apoiar no conceito de um *Minimum Viable Product* (MVP), em sua versão preliminar.

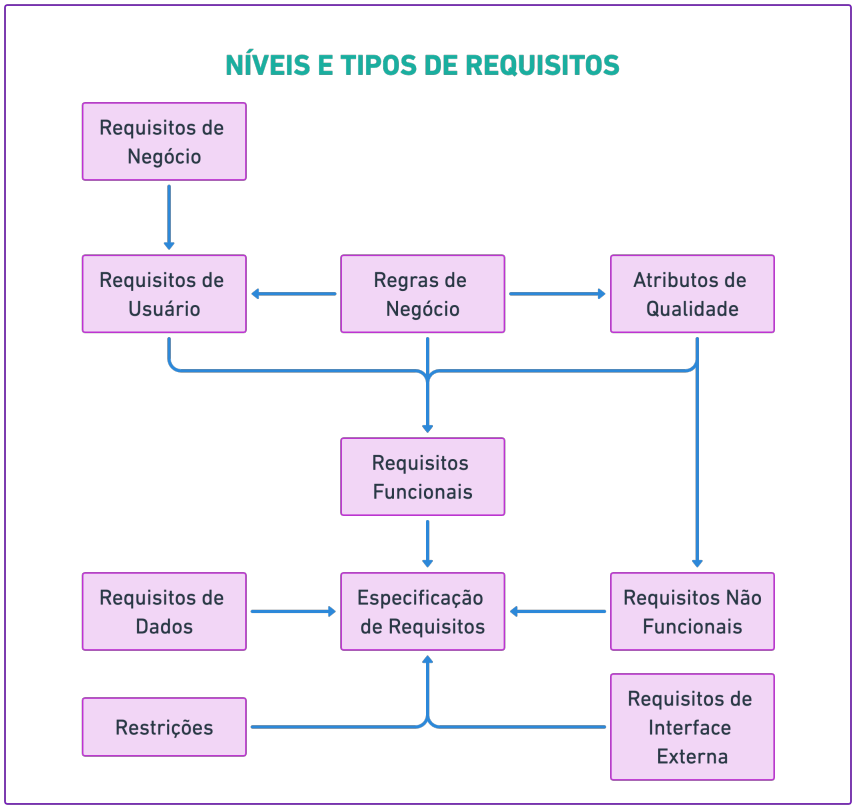
Diante da necessidade de se prover um MVP, e orientando-se pelas etapas e artefatos mencionados anteriormente, em especial pela priorização do *House of Quality*, foram automatizadas algumas partes deste processo. Tendo em vista a *House of Quality* e suas relações finalizadas, é utilizado o conceito de um Grafo, bem como o Algoritmo de Menor Caminho para a geração do MVP, em sua versão preliminar.

2.1 Engenharia de Requisitos

Por se tratar de uma área que se encontra entre o abstrato e o concreto, Westfall (2014), em seu estudo, define os diferentes porquês da existência da Engenharia de Requisitos, sendo:

- O que é?: são os requisitos responsáveis pela definição do que o *software* deve realizar (*requisitos funcionais*[1]) e possuir (*requisitos não funcionais*[2]) para agregar valor para os seus *stakeholders*[3]; Existem vários níveis e categorias de requisitos, mostrados na Figura 2.

Figura 2 – Níveis e Categorias de Requisitos



Fonte: Westfall, 2014

  - Os **requisitos de negócio** definem, diretamente, quais são os problemas que devem ser resolvidos e porque o produto de *software* está sendo desenvolvido;
  - Os **requisitos de usuário** têm o objetivo de visualizar as funcionalidades do produto de *software* a partir da perspectiva do usuário. Resumidamente,

[1] Declarações de serviços que o sistema deve prover, como tem que reagir para entradas e como precisa se comportar em situações especificas (SOMMERVILLE, 2010).
[2] Restrições aos serviços ou funções oferecidas pelo sistema. Aplicam-se ao sistema na totalidade (SOMMERVILLE, 2010).
[3] Grupo de pessoas interessadas no produto, podendo ser cliente, usuários etc.

eles definem o que o *software* deve realizar para que os usuários atinjam seus objetivos;

- Os **requisitos funcionais** de produto definem as funcionalidades do *software* que devem ser implementadas para que os usuários consigam realizar suas tarefas de forma fácil;
- As **regras de negócio** são declarações sobre a forma da empresa fazer negócio, ou seja, elas refletem as políticas, os padrões, as práticas, os regulamentos e as diretrizes de negócio;
- Os **atributos de qualidade** no nível de usuário delimitam requisitos não funcionais que determinam o produto de *software*. Incluem confiabilidade, disponibilidade, segurança, manutenibilidade, portabilidade, usabilidade, dentre outros;
- Os **requisitos de *interface* externa**, objetivamente, definem o fluxo de informações através de *interfaces* compartilhadas;
- As **restrições** têm o objetivo de mostrar quais destas limitações foram colocadas pelo fornecedor ao projetar e desenvolver o *software*, e
- Os **requisitos de dados** definem os itens de dados específicos que devem ser introduzidos como parte do produto de *software*.

- Para que serve? : um dos grandes nomes da Engenharia de *Software*, Brooks(1995), afirma que:

> A parte mais difícil da construção de um sistema de *software* é decidir precisamente o que deve ser feito. Nenhuma outra parte do trabalho conceitual é tão custosa quanto estabelecer detalhadamente os requisitos técnicos, incluindo todas as interfaces para os usuários, para os computadores, e para os outros sistemas do *software*. Nenhuma outra parte do trabalho prejudica tanto o sistema se for feita de maneira errada. Nenhuma outra parte é tão difícil de ser retificada posteriormente.

  Assim, pode-se entender o quão crítica é essa fase do processo de desenvolvimento de *software*, contexto que se insere esse trabalho.

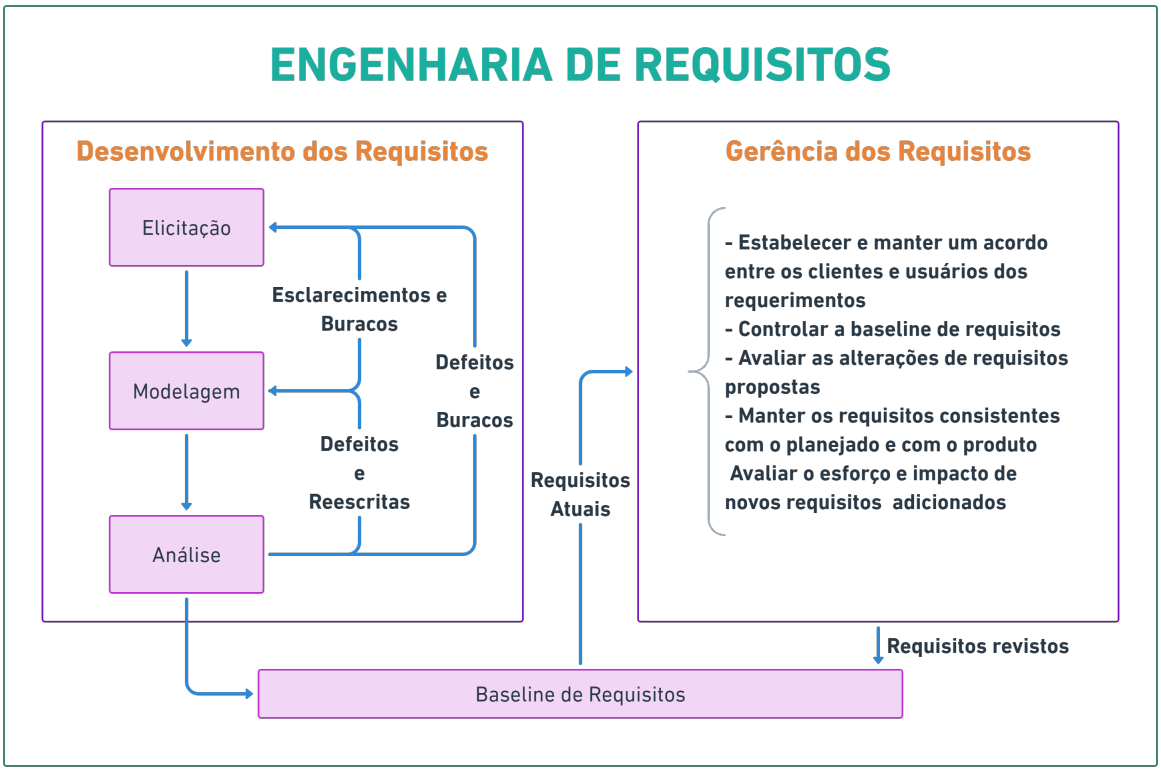
- Para quem?: os *stakeholders* são indivíduos que impactam ou que são impactados pelo produto de *software*. Portanto, possuem um nível de influência sobre o produto. Dentro desse contexto, existem diferentes perspectivas relacionadas à atuação de cada indivíduo, sendo as mais comuns:

  - Os Analistas de Requisitos são essenciais dentro dessa área pelo fato deles serem responsáveis pela elicitação, análise e especificação dos requisitos, além de comunicarem as necessidades aos desenvolvedores e às partes interessadas;
  - Os Arquitetos de *Software* são essenciais por criarem toda a arquitetura do *software* a partir dos requisitos elicitados e por dizerem como será implementado;
  - Os Desenvolvedores são responsáveis pela implementação do produto de *software*, traduzindo os requisitos em funcionalidades concretas;
  - Os Testadores de *Software* têm o objetivo de usar os requisitos elicitados como base, e criar casos de teste para que o produto de *software* seja testado sob condições específicas, afim de detectar defeitos para que, após os testes, passem confiabilidade de que o produto de *software* esteja funcionando de acordo com as características estabelecidas;
  - Os Gerentes de Projeto são responsáveis pelo planejamento e pelo monitoramento de todos os envolvidos no projeto, além de guiar o time de desenvolvimento de *software* para que o produto final seja entregue com todos os requisitos definidos anteriormente, e
  - O Gerente do Produto é um participante fundamental dentro desse processo, pois revisa todas as mudanças propostas; analisa os riscos e os impactos que podem vir a ocorrer, e aprova ou desaprova quaisquer mudanças, garantindo que essas mudanças foram implementadas e validadas.

- Quando?: a maioria das atividades do processo da Engenharia de Requisitos é feita nas primeiras fases do ciclo de vida do produto. Contudo, não deve ser restrito somente a esse período, visto que novos requisitos vão surgindo e o projeto deve se adaptar às novas realidades, pois deve ser um processo iterativo. Esse aspecto está ligado ao gerenciamento de requisitos, o qual necessita avaliar como novas requisições influenciam no desenvolvimento do projeto, assim como elencar os riscos e impactos causados por essas alterações. Posteriormente, após a aprovação, ocorre a implementação da nova funcionalidade. Durante a fase de desenvolvimento, estes requisitos devem ser usados como critérios para testar o produto e validar a correta implementação das funcionalidades. Os requisitos de negócio devem ser os primeiros a serem modelados, seguidos dos requerimentos de usuário e, por fim, os de produto.

- Como?: a Engenharia de Requisitos é um processo bem definido, com etapas claras e repletas de documentações que possam registrar e orientar o desenvolvimento do produto. A especificação e a gerência dos requisitos são dois grandes processos dentro deste escopo. A Figura 3 representa bem essas etapas e subetapas do processo da Engenharia de Requisitos, que vão ser mais bem especificadas no decorrer deste trabalho.

Figura 3 – Fluxo da Engenharia de Requisitos



Fonte: Adaptação de Westfall, 2014

Como descrito anteriormente, a Engenharia de Requisitos é uma área essencial na construção de um produto de *software*. Existem vários processos e várias técnicas utilizadas para que o produto final seja desenvolvido com qualidade, evitando retrabalhos no futuro. Alguns processos e técnicas utilizados na Engenharia de Requisitos serão tratados nas próximas seções.

2.2 Rastreabilidade

2.2.1 Pré-rastreabilidade

A Pré-rastreabilidade preocupa-se, objetivamente, em rastrear um requisito até a sua origem (*Backward-From*), ou da origem até o requisito (*Forward-To*). De forma exemplificada, quando um requisito é alterado, usa-se a pré-rastreabilidade para investigar as informações utilizadas para obtê-lo, ou seja, para delinear a pessoa interessada, quais técnicas ou quais documentos o requisito foi extraído, dentre outros (PINHEIRO, 2004). Pode-se ter o interesse ainda de rastrear da origem até o requisito. O importante é ter em mente que a pré-rastreabilidade está compreendida entre Origem-*Baseline* de Requisitos (sentido avante) ou entre *Baseline* de Requisitos-Origem (sentido reverso). Não é escopo de atuação da pré-rastreabilidade considerar rastros da *Baseline* de Requisitos para outros níveis de abstração mais baixos, tais como Arquitetura e Código.

2.2.1.1 5W2H

5W2H é um acrônimo em inglês que representa as principais perguntas que devem ser feitas e respondidas para investigar e relatar um fato ou situação: *Who*, *What*, *Where*, *When*, *Why*, *How* e *How Much*. O 5W2H é uma ferramenta empregada no planejamento estratégico de empresas, análise de negócios, elaboração de planos de negócios e outras áreas de gestão. Tem o objetivo de descrever um plano de ação com as atividades que precisam ser desenvolvidas com o máximo de clareza possível. Essa ferramenta baseia-se na elaboração de um artefato formado por sete perguntas: O quê? Por quê? Quem? Onde? Quando? Como? Quanto custa? (RABUSKE, 2016). Seguem as perguntas, as quais estão apresentadas em português. Entretanto, em inglês, cabe ressaltar, fica mais compreensível o nome atribuído ao *Framework* Conceitual 5W2H, com: *What*, *Why*, *Who*, *Where*, *When*, *How*, e *How Much*, respectivamente.

- O quê?: que ação será executada;
- Por quê?: porque a ação será executada;
- Quem?: quem irá participar da ação;
- Onde?: onde será executada a ação;
- Quando?: quando a ação será executada;
- Como?: como a ação será executada, e

Quadro 1: 5W2H do Aplicativo iGado

O quê?	Um aplicativo para o gerenciamento e gestão da pecuária.
Por quê?	Atualmente, no mercado, tem-se uma escassez de aplicativos para a gestão da pecuária. Além disso, muitos proprietários de fazenda ainda utilizam ferramentas arcaicas, como papel e caneta, para o gerenciamento da fazenda e, principalmente, da pecuária.
Quem?	Alunos de Engenharia de *Software* da Universidade de Brasília do Campus FGA-Gama que estão cursando a disciplina Arquitetura e Desenho de *Software*. Haverá a participação de 2 *stakeholders*, que irão auxiliar no processo de desenvolvimento do *software*.
Onde?	O projeto será realizado remotamente devido à pandemia. O aplicativo será disponibilizado para dispositivos mobile - *iOS* e *Android*
Quando?	O usuário poderá acessar o aplicativo em qualquer momento. Quando houver a necessidade de anotar informações, visualizar dados e gerar relatórios.
Como?	Por meio do desenvolvimento de um aplicativo mobile — iOS e Android, para facilitar o controle da gestão rural e permitir a geração de relatórios. O usuário poderá cadastrar as informações dos bovinos no aplicativo, além de ter uma janela para que o usuário gere os relatórios especificados por ele.
Quanto Custa?	R$ 226.551,64

Fonte: Autores, 2022.

- Quanto custa?: quanto custa para executar a ação.

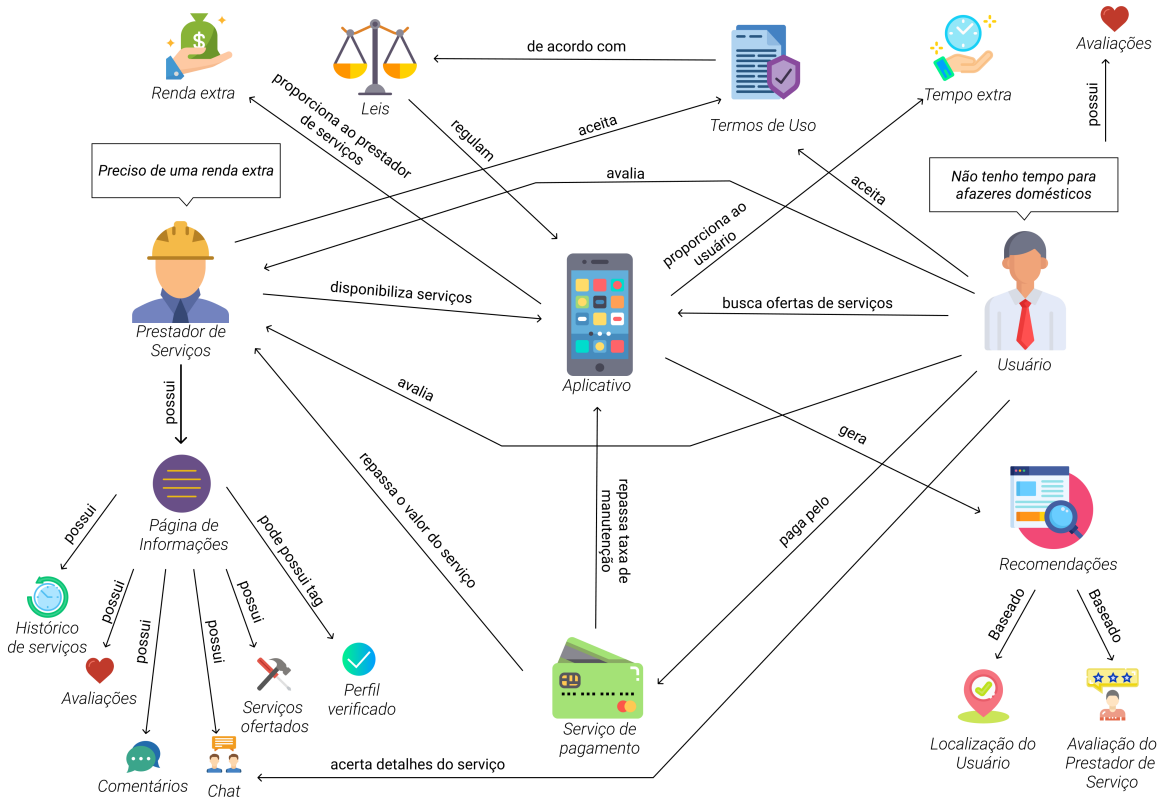
A Tabela 1 ilustra um exemplo de 5W2H, realizado para a aplicação *iGado* (Moreira et al. (2022)), que tem o objetivo de facilitar o gerenciamento rural de forma que o proprietário possa ter um controle sobre os dados dos seus bovinos. Ressalta-se que como esse artefato permite rastros para vários aspectos relevantes do *software*, incluindo *Stakeholders*, fontes, funcionalidades, dentre outras, há pré-rastreabilidade anotada.

2.2.1.2 *Rich Picture*

Rich Picture é uma ilustração que visa identificar as partes interessadas(*stakeholders*), suas preocupações e partes subjacentes. É uma ferramenta para registrar e raciocinar sobre os aspectos do contexto de trabalho e como eles afetam o produto (MONK; HOWARD, 1998). Além disso, ela se baseia em rascunhar desenhos e usar textos curtos e objetivos para expressar um momento, um desejo, uma atividade, dentre outras

necessidades. A Figura 4, desenvolvido para a aplicação *Pax* (Caixeta et al. (2022)), ilustra um exemplo de *Rich Picture* no contexto de uma aplicação cujo foco é conectar prestadores de serviços aos seus possíveis clientes. Cabe ressaltar que esse artefato, por revelar fontes de informação (ex. Leis e Termos de Uso); *Stakeholders* (ex. Prestador de Serviços e Usuário), fluxos de atividades (ex. Prestador de Serviços aceita Termos de Uso), dentre outras informações do Universo de Informações (LEITE, 2007), permite conferir pré-rastreabilidade.

Figura 4 – Rich Picture do Aplicativo PAX



Fonte: Autores, 2022.

2.2.2 Pós-rastreabilidade

A Pós-rastreabilidade consiste em rastrear um requisito para a implementação de um projeto. Exemplificadamente, quando um requisito é alterado, usa-se a pós-rastreabilidade para investigar o impacto desta mudança, no nível de implementação. Portanto, *Forward-from*, consistindo em ligar o requisito do sistema ao código, e man-

tendo o conhecimento de onde o requisito está implementado, e qual foi o caminho percorrido por ele, da *baseline* de requisitos, passando pelas etapas de desenho e arquitetura, teste e código. Há ainda, a possibilidade de realizar o sentido reverso, *Backward-to*. Neste caso, e dentro do contexto de Pós-rastreabilidade, permite-se ligar o código aos requisitos do sistema (PINHEIRO, 2004).

2.3 Elicitação

A Elicitação de Requisitos é o processo relacionado às atividades que permitem a compreensão de metas, objetivos e motivos para a construção de um novo sistema de *software*, e a identificação dos requisitos associados às partes interessadas, no intuito de compreender o sistema a ser desenvolvido (ELLIOTT, 2012).

Além disso, as informações coletadas durante a elicitação de requisitos, geralmente, precisam ser modeladas e analisadas (verificadas e validadas), antes do início da implementação do sistema (NUSEIBEH; EASTERBROOK, 2000).

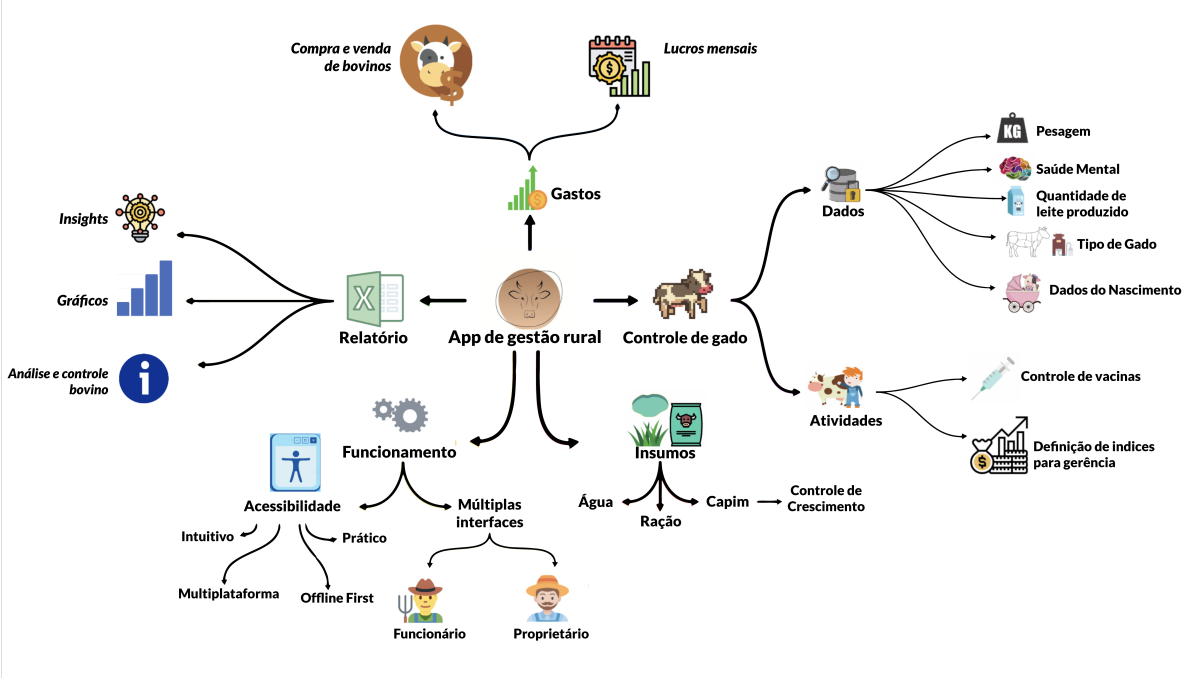
2.3.1 *Brainstorming*

O *Brainstorming* ou "Tempestade de Ideias", por ser uma técnica de grupo, tem o objetivo de coletar o maior número de ideias acerca de um determinado tema ou questão. É utilizada, geralmente, na fase de planejamento de um projeto. Propõe que um grupo de pessoas se reúnam de modo a colaborar para uma "tempestade de ideias", onde cada ideia é conferida e acrescentada, para gerar um longo processo de sugestões e discussões. Vale ressaltar que nenhuma ideia, inicialmente, é criticada ou julgada (MAZZOTTI; BROEGA; GOMES, 2012).

O *Brainstorming* é usado para gerar novas ideias, deixando a mente livre para aceitar todas as ideias sugeridas e, assim, permitir a liberdade para a criatividade (BATISTA; CARVALHO, 2003). Uma dificuldade é documentar um *Brainstorming*. Entretanto, pode-se usar desde gravações em vídeos e áudios, e até mesmo Mapas Mentais e outros modelos que permitem rápidas anotações.

A Figura 5, desenvolvida para a aplicação *iGado* (Moreira et al. (2022)), mostra um *Mapa Mental*, especificado a partir de um debate no contexto de elicitação de um *software* no domínio de uma aplicação *mobile*, que visa facilitar o gerenciamento rural de proprietários de fazendas, em que a equipe usava a técnica de *Brainstorming*.

Figura 5 – Mapa Mental do Aplicativo iGado



Fonte: Autores, 2022.

2.3.2 Questionário

O Questionário é uma técnica de coleta de informações que permite aos envolvidos no projeto estudar atitudes, crenças, comportamentos e características do público alvo que podem ser afetados pelo sistema atual e pelo sistema proposto (KENDALL; KENDALL, 1992). Deve ser realizada com muita cautela. A elaboração das perguntas, que vão constituir o questionário, é um processo engenhoso, complexo e delicado. Um questionário mal elaborado pode, facilmente, levar a conclusões errôneas, que acabam sendo prejudiciais para o projeto que está sendo desenvolvido (JUNIOR, 2005).

De acordo com Segundo et al. (2017), um questionário deve ser elaborado tendo em mente alguns aspectos-chave, dentre eles:

1. Coleta de informações sobre o perfil dos participantes;
2. A informação que se quer descobrir, onde devem ser considerada, primeiramente, uma mensagem introdutória, direcionando o participante sobre o questionamento, seja essa de múltipla escolha ou mais discursiva, e

3. Análise dos resultados obtidos, procurando apresentá-los usando gráficos, tabelas, e quadros bem claros e objetivos, com base, preferencialmente, em diferentes pontos de vista.

2.3.3 Protótipo de Baixa Fidelidade

Um Protótipo, segundo Sommerville (2010), "é uma versão inicial de um sistema de *software*, usado para demonstrar conceitos, experimentar opções de projeto e descobrir mais sobre o problema e suas possíveis soluções". É muito significativo no processo de Engenharia de Requisitos pelo fato de ajudar na elicitação dos requisitos de *software*, e para estudar soluções específicas do *software*.

2.3.4 *Storytelling*

Storytelling, ou simplesmente História, é uma técnica que busca, por meio da análise da descrição de algum processo ou tarefa realizado em algum contexto, extrair *features* do sistema. Vale ressaltar que não existe um formato ou estrutura pré-definida, mas há descrições de altos níveis de uso do sistema. As *Storytellings* são muito usadas, visando conferir um panorama do sistema (SOMMERVILLE, 2010).

O próprio cérebro humano, segundo estudos, é mais propenso ao aprendizado proveniente de histórias. Por retratarem a experiência de quem conta, sendo essa validada pelo próprio orador, as narrativas proporcionam um retrato realista e lógico de algo vivido por seu interlocutor. Isso proporciona um entendimento com uma carga de sentimentos e fatos que podem ser usados para a extração de requisitos mais ricos (BOULILA; HOFFMANN; HERRMANN, 2011).

2.3.5 Entrevista

Entrevista é uma das técnicas mais utilizadas por possuir uma comunicação mais natural entre as pessoas. As entrevistas são usadas para a obtenção de conhecimento sobre um determinado tema/domínio, por perguntas feitas aos usuários especialistas, além de trazer muitas informações aos desenvolvedores do projeto (BATISTA; CARVALHO, 2003). De acordo com Sommerville (2010), elas podem ser de dois tipos:

- Fechadas: seguem um conjunto de perguntas pré-definidas, e
- Abertas: não se tem uma agenda pré-definida.

Na prática, geralmente, as entrevistas são híbridas, orientando-se por ambos os tipos.

2.3.6 Análise de Protocolo

A Análise de Protocolo consiste em pedir a uma pessoa para que ela se envolva em uma tarefa, e fale em voz alta o seu processo de pensamento. Sendo assim, têm-se insumos para ocorrer a extração de possíveis requisitos do sistema (GOGUEN; LINDE, 1993). Usuários alegam que essa categoria de linguagem pode ser considerada uma "verbalização direta do processo cognitivo específico". Vale ressaltar que a Análise de Protocolo está sujeita a problemas de interpretações pelos analistas (BELGAMO; MARTINS, 2000).

2.4 Modelagem

Esta parte do processo da Engenharia de Requisitos consiste em estruturar os requisitos levantados no período da elicitação (Seção 2.3) de forma sistemática. De acordo com Sommerville (2010), os requisitos "devem ser claros, inequívocos, fáceis de entender, completos e consistentes".

Vale mencionar que os documentos gerados, nesta fase, devem ser de fácil entendimento para os *stakeholders* do projeto, sem existirem detalhes técnicos de implementação ou arquitetura do sistema (SOMMERVILLE, 2010).

2.4.1 *Backlog*

O *Backlog* do Produto é um artefato essencial na Engenharia de Requisitos, na metodologia ágil e no ciclo de desenvolvimento de um produto, pois os requisitos elicitados constam nele. Em uma definição mais concreta, o *Backlog* é uma lista ordenada e emergente do que é necessário no produto (segundo os *Stakeholders* consultados), ou seja, é um artefato que agrupa os requisitos (CAROLI, 2021). As funcionalidades são descritas no *Backlog*, sendo esse o que o cliente espera receber ao final do projeto.

Dentro desta construção, Caroli (2021) delimita alguns conceitos usados para uma definição mais consistente dos requisitos, sendo elas:

1. **Épico**: é uma grande parte de trabalho que, geralmente, é dividida em tarefas menores, ou seja, histórias de usuário. Trata-se de um trabalho realizado em semanas ou meses;

2. ***Feature***: é uma funcionalidade que faz parte de um módulo, possuindo seus requisitos funcionais e suas regras de negócio. Trata-se de alguma ação ou interação do usuário com o produto, e

3. **História de Usuário**: é uma função da *feature*, e está associado a ela. Objetivamente, equivale aos requisitos funcionais de uma *interface*. Além disso, trata-se de uma pequena parte da funcionalidade que faz sentido para os *stakeholders*.

Dessa forma, foi construído o *backlog* da ferramenta proposta, podendo ser observado no Apêndice A.

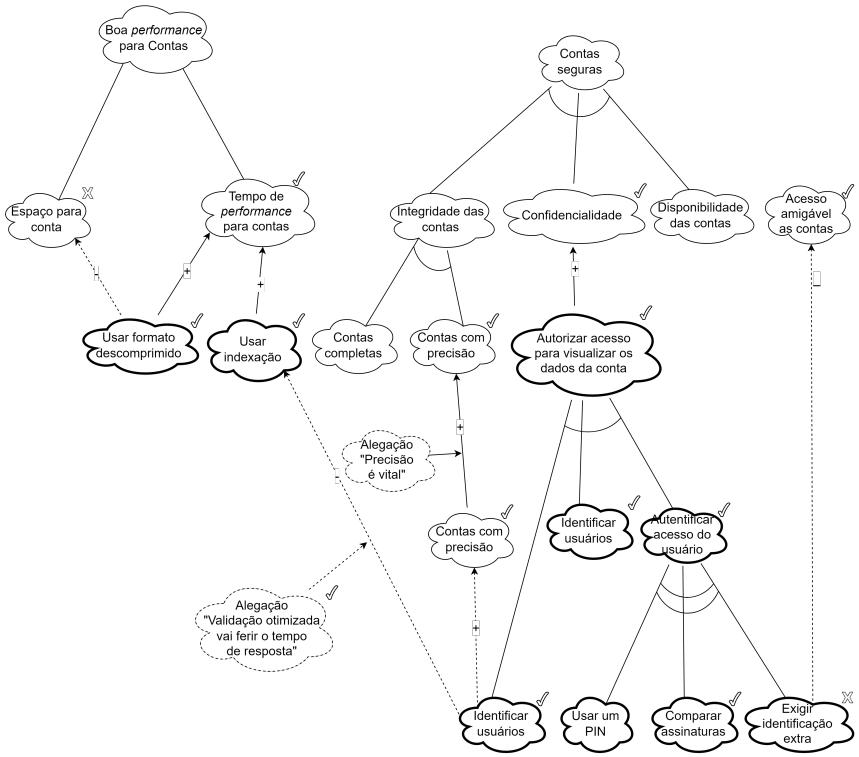
2.4.2 NFR

O NFR *Framework* (*Non-Functional Requirements*) é um artefato que visa tratar os requisitos não funcionais, expressando-os sistematicamente, e usando-os para guiar o processo de desenvolvimento do *software*.

Segundo COUTO e MARTINS (2008), trata-se de uma abordagem que confere operacionalizações, considerando os requisitos não funcionais. Esses requisitos são especificados como metas a serem atingidas. Já as operacionalizações são vistas como ações concretas, que uma vez realizadas, colaboraram com o cumprimento das metas a serem atingidas.

Além disso, de acordo com Chung et al. (2012), o "NFR *Framework* usa os requisitos não funcionais como segurança, precisão, desempenho e custo para conduzir o processo geral de *design*. O Framework visa colocar os requisitos não funcionais em primeiro lugar na mente do desenvolvedor". A Figura 6 ilustra a mesma modelagem, em seu rigor de notação.

Figura 6 – NFR com o Rigor da Notação

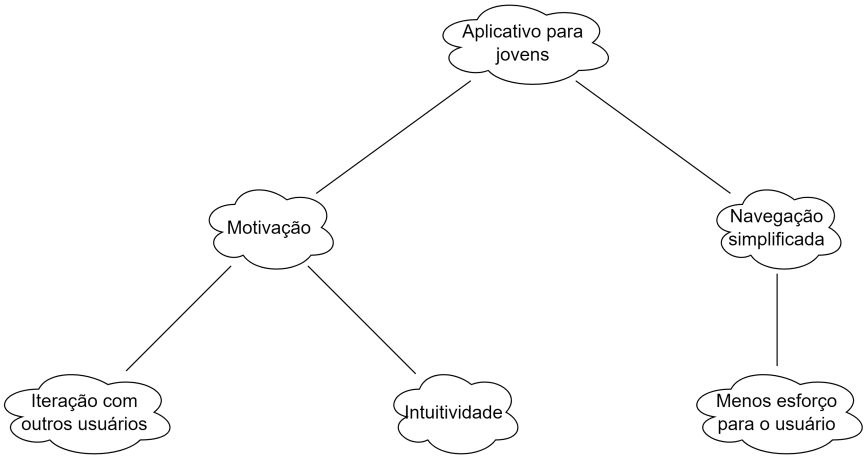


Fonte: Adaptado de Ullah, Iqbal e Khan, 2011.

A intenção do presente trabalho foi usar os princípios desse modelo conceitual, visando um tratamento mais adequado dos requisitos não funcionais. Não foi utilizada a notação, em seu rigor. Entretanto, foi utilizado um processo guiado, orientando-se por esse modelo conceitual, e procurando facilitar o processo de modelagem dos requisitos não funcionais por parte dos usuários da ferramenta. Dessa forma, o NFR foi utilizado para definir os requisitos não funcionais de forma mais lapidada, ou seja, no seu segundo e/ou terceiro nível. Sendo assim, os requisitos não funcionais, na sua forma de operacionalização, foram modelados no *House of Quality* (Seção 2.7).

A Figura 7 ilustra a adaptação usada na ferramenta, elaborada pelos autores, e orientada pelos princípios do NFR *Framework*. O intuito é ser uma notação mais simples e compreensível aos usuários da ferramenta.

Figura 7 – NFR Adaptado Proposto



Fonte: Autores, 2022.

2.4.3 Léxicos

O Léxico é definido como um conjunto de vocábulos de uma língua, dispostos em ordem alfabética e com as respectivas significações. Uma notação na modelagem de requisitos que usa a descrição de termos via léxico é o LAL (Léxico Ampliado da Linguagem). Trata-se de uma técnica que consiste em definir os símbolos de uma linguagem. Esses símbolos são descritos em **noções**[4] e **impactos**[5] (LEITE; FRANCO, 1993). Cada Léxico pode ser definido em somente um tipo, sendo eles:

- Objetos: são aqueles que se relacionam com outros objetos;
- Verbos: são as ações, e
- Estados: são gerados a partir de ações (verbos).

Considerando um *software* no domínio de um aplicativo *mobile*, denominado Pró-Espécies Peixes, com a finalidade de aumentar a quantidade de informações disponíveis sobre as espécies de peixes no cerrado brasileiro, tem-se um exemplo de especificação de léxico, Tabela 2, para o termo **Georreferência**, sendo esse do tipo **Objeto**.

[4]Significa o símbolo, é colocado no sentido denotativo da linguagem.
[5]Efeito/uso/ocorrência do símbolo na aplicação, que pode ser colocado no sentido conotativo da linguagem.

Quadro 2: Exemplo de léxico (Objeto)

Nome	Georreferência
Classificação	Objeto
Sinônimos	Geolocalização
Noção	Funcionalidade de mapeamento e fornecimento da localização do usuário em um registro
Impacto	A Georreferência, ou Geolocalização, permite fornecer a localização de onde ocorreu um determinado registro.

Fonte: Autores, 2022.

2.5 Verificação

Verificação consiste em uma etapa rigorosa do processo da Engenharia de Requisitos, onde os artefatos gerados na modelagem (Seção 2.4) são revisados. Essa etapa é essencial, pois no contexto do desenvolvimento de um *software*, uma refatoração é mais custosa que a inspeção de um artefato previamente à implementação. São usadas diversas técnicas nesta etapa (SILVA; CHAPETTA; TRAVASSOS, 2004), a depender da necessidade.

Nesse contexto, cabe uma ressalva. A etapa de análise, inerente na Engenharia de Requistos, envolve, principalmente, duas sub-atividades, sendo: Verificação e Validação. A verificação compreende analisar se o *software* está correto, conforme foi especificado/planejado/modelado. Já a validação analisa se o *software*, obtido até o momento da análise, atende às expectativas do cliente.

É possível perceber que a validação é mais complicada de ser realizada, na Engenharia de Requisitos, visto que, nesta etapa do ciclo de vida, não há um incremento de *software* avançado, tampouco um produto concluído. Quando muito, tem-se um protótipo ou um incremento de *software* tido como uma prova de conceito. Portanto, costuma-se validar esse protótipo ou essa prova de conceito usando um grupo mais específico de *Stakeholders*.

Em contrapartida, a verificação é comumente realizada na Engenharia de Requisitos, sendo utilizada, principalmente, a Técnica de Inspeção (FAGAN, 1976) (KALINOWSKI; SPíNOLA, 2008). Neste trabalho, a verificação foi priorizada.

2.6 Priorização (MVP)

O MVP (*Minimum Viable Product*) é definido como a versão mais simples de um produto que pode ser colocado para negociação. Determina quais são as funcionalidades mais importantes, que agregam mais valor ao produto para ele ser validado e efetivado pelo usuário final (CAROLI, 2018). Faz-se uso deste conceito, no sentido da ferramenta elaborada auxiliar os Engenheiros de *Software* na obtenção de um MVP, sendo esse orientado pelas boas práticas da Engenharia de Requisitos, mitigando problemas, mencionados no Capítulo 1.

Além disso, o MVP é usado para validar o retorno de determinado investimento, mesmo antes de o produto estar completamente finalizado, trazendo uma versão mais condensada, mas que, no entanto, já é suficiente para resolver o problema para o qual foi desenvolvido (ZANETTE, 2020).

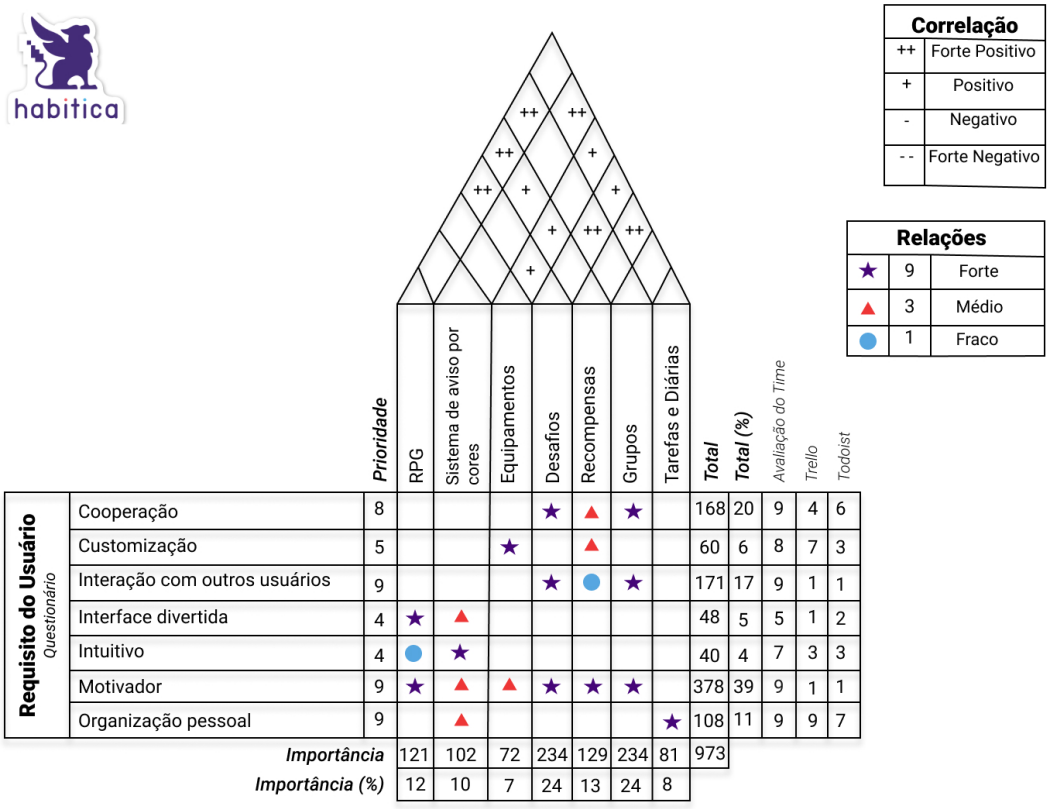
2.7 *House of Quality*

House of Quality ou Casa da Qualidade é um processo de planejamento sistemático, elaborado para compreender de forma explícita a "voz do cliente" no *design* do produto. Esta pode ser vista como uma ferramenta gráfica para compreender e avaliar a relação entre as preocupações do cliente e as do desenvolvedor (HOWARD, 1999), conforme consta na Figura 8.

Trata-se de um artefato composto por cinco partes, sendo elas:

- No centro, são comparados os requisitos técnicos propostos com as necessidades do cliente, classificando sua relação;
- À direita, tem-se uma comparação dos concorrentes com relação ao produto da empresa, sobre as necessidades do cliente. Nela, é possível comparar os diferentes parâmetros dos concorrentes com o produto, bem como ter uma noção da qualidade do produto;
- À esquerda, têm-se as necessidades dos clientes acordadas com um peso de prioridade, para ser possível atender a maioria dessas necessidades;
- Na parte inferior, são comparados todos os requisitos dos produtos concorrentes com o da empresa, da mesma forma que com as urgências, e

Figura 8 – *House of Quality* do Aplicativo Habitica



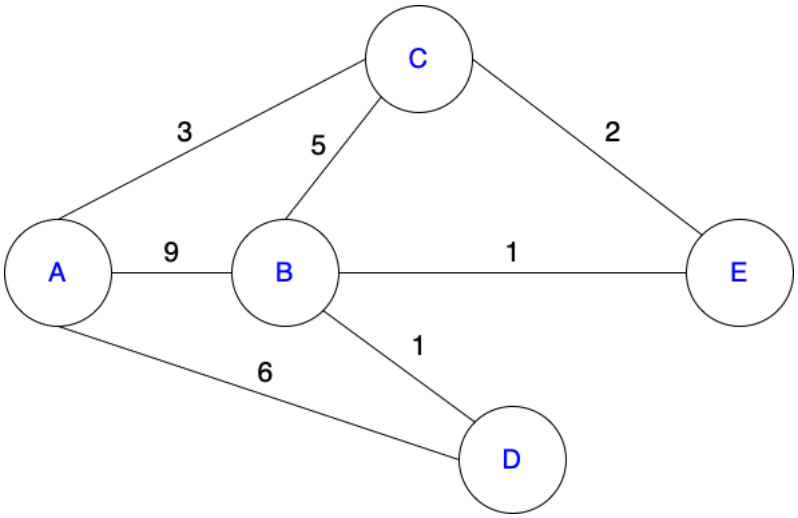
Fonte: Autores, 2022.

- Na parte superior, têm-se as correlações dos requisitos técnicos, sendo onde são determinados os parâmetros conflitantes. Neles, que a equipe deve ponderar e descobrir o meio-termo no desenvolvimento.

2.7.1 Grafos

Grafo é um ramo da matemática que estuda as relações entre os objetos de um determinado conjunto. Portanto, um Grafo G é definido por um conjunto $V(G)$ de elementos denominados Vértices; um conjunto $E(G)$ de elementos chamados Arestas, onde cada aresta associa um ou dois vértices. A Figura 9 mostra a representação de um grafo com os vértices (A, B, C, D, E), suas respectivas arestas e os pesos $(3, 5, 9, 2, 1, 1, 6)$.

Figura 9 – Exemplo de Grafo



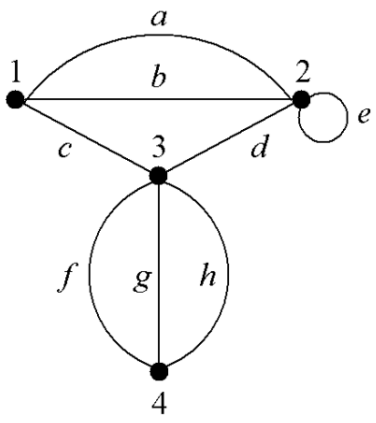
Fonte: Autores, 2022.

Na computação, a lista de adjacências é uma das formas de representação de um grafo, existindo também a representação por matriz. Na lista de adjacências, têm-se as ligações de vértices e arestas com seus pesos respectivos. É uma maneira de codificar relações emparelhadas entre um conjunto de objetos. Consiste em uma coleção V de nós e uma coleção E de arestas, onde as arestas juntam dois nós (KLEINBERG; TARDOS, 2006).

De acordo com Prestes (2016), um grafo $G = (V, A)$, com $|V| = n$ e $|A| = m$, pode ser representado como uma matriz de adjacência. A matriz de adjacência é uma matriz $M = [m_{i,j}]$ simétrica $n \times n$ que armazena o relacionamento entre os vértices do grafo. Cada entrada $m_{i,j}$ é igual a:

$$m_{i,j} = \begin{cases} 0, & \text{se } i \text{ não é adjacente a } j \\ m, & \text{se } i \text{ é adjacente a } j \text{ com } i \neq j \\ p, & \text{se } i = j, p \text{ é a quantidade de laços incidentes ao vértice } i \end{cases}$$

Figura 10 – Exemplo de um Grafo para a representação em Matrizes



(b)

Fonte: (PRESTES, 2016)

A Figura 11 representa uma matriz de adjacência para o Grafo exemplificado na Figura 10.

Figura 11 – Matriz de adjacência para o Grafo da Figura 10

	1	2	3	4
1	0	2	1	0
2	2	1	1	0
3	1	1	0	3
4	0	0	3	0

Fonte: (PRESTES, 2016)

2.7.1.1 Algoritmo de Menor Caminho

O Algoritmo de Menor Caminho consiste em calcular, para cada vértice V, a menor distância de S para V, ou seja, a menor soma possível de pesos de arestas que formem um caminho de S para V, onde S e V são vértices dentro de um grafo G. Os Algoritmos de *Dijkstra* e *Bellman-Ford* são os mais conhecidos para realizar o cálculo da menor distância entre dois vértices (EVEN; EVEN, 2011).

No caso do presente trabalho, foi utilizado o Algoritmo de Dijkstra para se calcular o menor caminho entre as relações de requisitos funcionais e não funcionais para

que se tenha o Produto Mínimo Viável, na sua versão preliminar. Foi escolhido esse algoritmo por não haver pesos negativos na modelagem do grafo do relacionamento entre os requisitos.

2.8 Resumo do Capítulo

Neste capítulo, foram apresentados os principais conceitos que servem como base para o entendimento do contexto do trabalho desenvolvido, no caso, o *iFlow*: uma ferramenta que oferece apoio ao processo de Engenharia de Requisitos, objetivando a construção de um MVP.

Para a apresentação desses conceitos, fez-se necessária uma breve explicação sobre cada etapa do processo de Engenharia de Requisitos, fazendo correlação com a ferramenta desenvolvida. O grande intuito é contextualizar o processo e os artefatos gerados em cada etapa.

Em seguida, tem-se uma explanação concreta da Engenharia de Requisitos e sua importância no processo de construção de um produto de *software*, bem como diversas etapas essenciais neste processo, de modo a oferecer um panorama geral a respeito da Engenharia de Requisitos. Em seguida, foram apresentadas as principais etapas no processo de Engenharia de Requisitos, sendo elas: Pré-rastreabilidade; Pós-rastreabilidade; Elicitação; Modelagem; Verificação; e Priorização.

Por fim foram explorados conceitos importantes no contexto do desenvolvimento deste trabalho que foram o *House of Quality* e os Grafos.

3 Referencial Tecnológico

Este capítulo apresenta as principais tecnologias que conferem apoio ao desenvolvimento deste trabalho na totalidade. As escolhas tecnológicas fundamentaram-se no uso de recursos gratuitos ou de código aberto; na afinidade dos autores, e no conhecimento prévio, considerando as recomendações da comunidade de *software* diante das necessidades de projeto apresentadas. A intenção foi minimizar os riscos quanto ao desenvolvimento da ferramenta desenvolvida. O capítulo está organizado em três partes, sendo: Tecnologias de Apoio à Aplicação (Seção 3.1), Tecnologias de Apoio à Pesquisa e à Comunicação (Seção 3.2) e o Resumo do Capítulo (Seção 3.3).

3.1 Tecnologias de Apoio à Aplicação

3.1.1 TypeScript

O *TypeScript* (TS) é uma linguagem dinâmica, interpretada e multiparadigma, que confere recursos aos paradigmas orientados a objetos, imperativos e declarativos (MICROSOFT, 2022b), além de ser fortemente tipada. A linguagem foi utilizada a partir do emprego do *NodeJS* e do *ReactJS* como ferramentas de apoio.

3.1.2 ReactJS

O *ReactJS* é uma biblioteca *JavaScript* visando criar *interfaces* de usuário (FACEBOOK, 2022). O *ReactJS* foi escolhido devido ao seu bom desempenho com virtual DOM (*Document Object Model*), e pela facilidade de criação de componentes que ele fornece. Além disso, o *ReactJS* utiliza visualizações declarativas, que torna o código mais previsível e fácil de depurar.

3.1.3 NodeJS

O *NodeJS* faz uma plataforma projetada para o desenvolvimento de aplicações escaláveis de rede, sendo um ambiente de execução de código *JavaScript* (JS, 2022). O NodeJS foi utilizado para a implementação da *API* que é a comunicação entre as camadas da ferramenta.

3.1.4 GitHub

O *GitHub* é uma plataforma que auxilia no ciclo de desenvolvimento de uma aplicação, hospedando o código-fonte (MICROSOFT, 2022a). A plataforma foi utilizada para armazenar e centralizar o código-fonte da ferramenta.

3.1.5 Git

O *Git* é uma ferramenta de controle de versão gratuita e de código aberto (TORVALDS; HAMANO, 2022). Foi utilizado para realizar o versionamento do código-fonte da ferramenta.

3.1.6 Docker

O *Docker* é uma plataforma para a criação, execução e publicação de contêineres (DOCKER, 2022b). Essa plataforma foi utilizada para a gerência de configuração da ferramenta, objetivando facilitar e otimizar as atividades de desenvolvimento e de implantação com a utilização dos contêineres.

3.1.7 Docker Compose

O *Docker Compose* é uma ferramenta para definir e executar contêineres. Com o Compose, faz-se uso de um arquivo *YAML* para configurar os serviços da aplicação (DOCKER, 2022a). O *Docker Compose* foi utilizado para a orquestração dos contêineres dos serviços utilizados pela ferramenta.

3.1.8 Trello

O Trello é uma ferramenta de gerenciamento de projetos que utiliza um esquema de listas, cartões e quadros para organizar atividades dentro de um projeto (ATLASSIAN, 2022). A ferramenta foi utilizada para a criação de um quadro *Kanban*, permitindo que haja o acompanhamento das atividades inerentes ao projeto.

3.1.9 Figma

Figma é um editor de gráficos vetoriais e, ao mesmo tempo, uma ferramenta que pode ser utilizada para prototipagem (SHARMA; TIWARI, 2021). Optou-se por essa

ferramenta, uma vez que, a mesma viabilizou a modelagem de um protótipo de alta fidelidade do produto, oriundo deste trabalho.

3.2 Tecnologias de Apoio à Pesquisa e à Comunicação

3.2.1 Google Drive

O *Google Drive* é uma plataforma criada pelo *Google*, e consiste em armazenar, compartilhar e colaborar arquivos e pastas (GOOGLE, 2022). O *Google Drive* foi utilizado para a elaboração de planilhas e documentos colaborativos referentes ao projeto.

3.2.2 LaTeX3

O *LaTeX3* é um sistema de preparação de documentos para composição tipográfica de alta qualidade (LAMPORT, 2022). O *LaTeX3* foi utilizado pelo fato de auxiliar na formatação do documento.

3.2.3 Overleaf Community Edition

O *Overleaf* é uma ferramenta *online* de escrita em *LaTeX* e publicação colaborativa para a elaboração de documentos técnicos e científicos (MOULTON, 2022). O *Overleaf* foi utilizado para a produção e a edição do texto científico.

3.2.4 Telegram

O *Telegram* é um aplicativo, disponível nas versões *mobile* e *web*, para o compartilhamento de mensagens. Além de ter suporte ao envio de mensagens de texto, voz, mídia e documentos, os usuários podem fazer chamadas de vídeo e ligações (DUROV; DUROV, 2022). O aplicativo foi utilizado para o compartilhamento de arquivos, *links*, reuniões, alertas e discussões imediatas.

3.2.5 Discord

O *Discord* é uma plataforma de comunicação que permite a efetuação de chamadas e a realização de videochamadas (DISCORD, 2022). A plataforma foi muito utilizada para a realização de reuniões remotas entre os desenvolvedores do projeto.

3.2.6 Notion

O *Notion* é uma plataforma utilizada para criar sistemas de gerenciamento de conhecimento, gerenciamento de dados, anotações, gerenciamento de projetos, entre outros (LABS, 2022). A ferramenta foi utilizada para haver organização e separação das atividades, além de salvar os artigos utilizados como base para o desenvolvimento do projeto.

3.3 Resumo do Capítulo

Neste capítulo, foram apresentadas as principais tecnologias que auxiliaram no desenvolvimento da ferramenta de apoio ao processo de Engenharia de Requisitos, *iFlow*, bem como as ferramentas que foram utilizadas para dar apoio à pesquisa e à comunicação.

As ferramentas de apoio ao desenvolvimento da ferramenta foram dispostas de modo a descrever como foram utilizadas para contribuir desde o gerenciamento do código-fonte, incluindo gerência de configuração e evolução, até o desenvolvimento do sistema, envolvendo as camadas de processamento de dados (*back-end*) e de *interface* com o usuário (*front-end*).

Além disso, as ferramentas de apoio à pesquisa e à comunicação foram descritas, de modo a indicar como foram utilizadas para o processo de pesquisa, referenciação e escrita. Adicionalmente, têm-se as ferramentas para facilitar a comunicação entre as partes desenvolvedoras do projeto, completando, assim, a apresentação do suporte tecnológico necessário na elaboração deste trabalho.

4 Metodologia

Neste capítulo, serão apresentados detalhes da metodologia aplicada no desenvolvimento teórico e prático do trabalho. Em primeiro momento, as atividades de pesquisa são classificadas conforme à abordagem, natureza, objetivos e procedimentos (Seção 4.1). Além disso, é definida a sequência de atividades realizadas neste trabalho de conclusão de curso (Seção 4.2). Logo em seguida, tem-se uma seção caracterizando os aspectos metodológicos adotados no levantamento bibliográfico (Seção 4.3). Posteriormente, tem-se o panorama do processo de desenvolvimento de *software* (Seção 4.4). Adicionalmente, foi apresentada a metodologia de análise dos resultados obtidos (Seção 4.5). Na sequência, constam os cronogramas de atividades (Seção 4.6), tanto para o escopo de atividades do TCC1, quanto para o escopo de atividades do TCC2. Por fim, tem-se o Resumo do Capítulo (4.7).

4.1 Metodologia de Pesquisa

De acordo com Gerhardt e Silveira (2009), a Metodologia pode ser classificada como o estudo da organização, dos caminhos a serem trilhados, para se realizar uma pesquisa ou um estudo, ou para produzir ciência. Já a pesquisa científica, refere-se ao resultado de "um inquérito ou exame minucioso, realizado para resolver um problema, recorrendo a procedimentos científicos" (GERHARDT; SILVEIRA, 2009).

A partir das definições de Gerhardt e Silveira (2009), esta seção visa categorizar a pesquisa deste trabalho quanto à abordagem, à natureza, aos objetivos e aos procedimentos.

4.1.1 Abordagem da Pesquisa

A pesquisa utilizada neste trabalho é classificada como Qualitativa. A pesquisa qualitativa visa obter uma compreensão mais profunda de um grupo social, de uma organização, entre outros. Portanto, a pesquisa qualitativa, preocupa-se com aspectos da realidade que não podem ser quantificados, concentrando-se em compreender e explicar a dinâmica das relações sociais. (GERHARDT; SILVEIRA, 2009).

Nessa abordagem, os passos adotados para validação do produto ocorram a partir

da coleta de métricas qualitativas, associadas à satisfação do usuário quanto ao uso da ferramenta *iFlow*.

4.1.2 Natureza da Pesquisa

A pesquisa deste trabalho pode ser classificada como Aplicada. A pesquisa aplicada visa produzir conhecimento para aplicação prática, de modo a resolver problemas específicos. Por outro lado, a pesquisa básica visa gerar conhecimentos novos, sem aplicação prática prevista (GERHARDT; SILVEIRA, 2009).

O viés aplicado do trabalho é bastante sensível, dado que a ideia foi prover uma ferramenta, para um público alvo técnico, visando mitigar problemas bem específicos do domínio de uma área de interesse da Engenharia de *Software*, no caso, a Engenharia de Requisitos. Adicionalmente, e ainda corroborando com a definição de uma pesquisa aplicada, o *iFlow* automatiza algumas atividades, além de gerar um Produto Mínimo Viável, em sua versão preliminar.

4.1.3 Objetivos da Pesquisa

Como objetivo, a presente pesquisa classifica-se como Exploratória. A pesquisa exploratória consiste em proporcionar maior familiaridade com o problema, visando torná-lo mais explícito ou formular hipóteses (GIL et al., 2002). Nesse sentido, a pesquisa exploratória se encaixa bem no presente trabalho, visando compreender, a partir da identificação de preocupações inerentes à área de Engenharia de Requisitos, explorar soluções que mitiguem, ao menos, tais preocupações, procurando contribuir de alguma forma com o sucesso do *software* produzido, e considerando as boas práticas acordadas na literatura especializada.

4.1.4 Procedimentos

A pesquisa deste presente trabalho pode ser classificada como Bibliográfica e Pesquisa-Ação. A pesquisa bibliográfica é realizada a partir de um acervo de referências teóricas analisadas e publicadas por meios escritos e tecnológicos, como livros, artigos científicos, *sites*, entre outros. A pesquisa-ação pressupõe a participação planejada do pesquisador na situação problemática a ser investigada, visando diagnosticar um determinado problema sobre uma dada situação, para chegar em um resultado prático (GIL et al., 2002).

Neste trabalho, buscou-se realizar um levantamento bibliográfico, apresentado na Seção 4.3, visando acordar boas práticas e preocupações da área de Engenharia de Requisitos, ambos abordados na ferramenta *iFlow*. Além disso, no intuito de levantar dados que corroborem ou não com a usabilidade da ferramenta, em simultâneo, foi utilizada a Pesquisa-Ação, que foi mais bem detalhada no Capítulo 6, visando coletar perspectivas dos usuários e gerar insumos de melhoria para trabalhos futuros. Os detalhes estão na Seção 4.5.

4.1.5 Classificação da Pesquisa

A classificação da pesquisa quanto à abordagem, à natureza, aos objetivos e aos procedimentos está descrita na Tabela 3

Quadro 3: Classificação da Pesquisa

Abordagem	Qualitativa
Natureza	Aplicada
Objetivo	Exploratória
Procedimentos	Levantamento Bibliográfico e Pesquisa-Ação

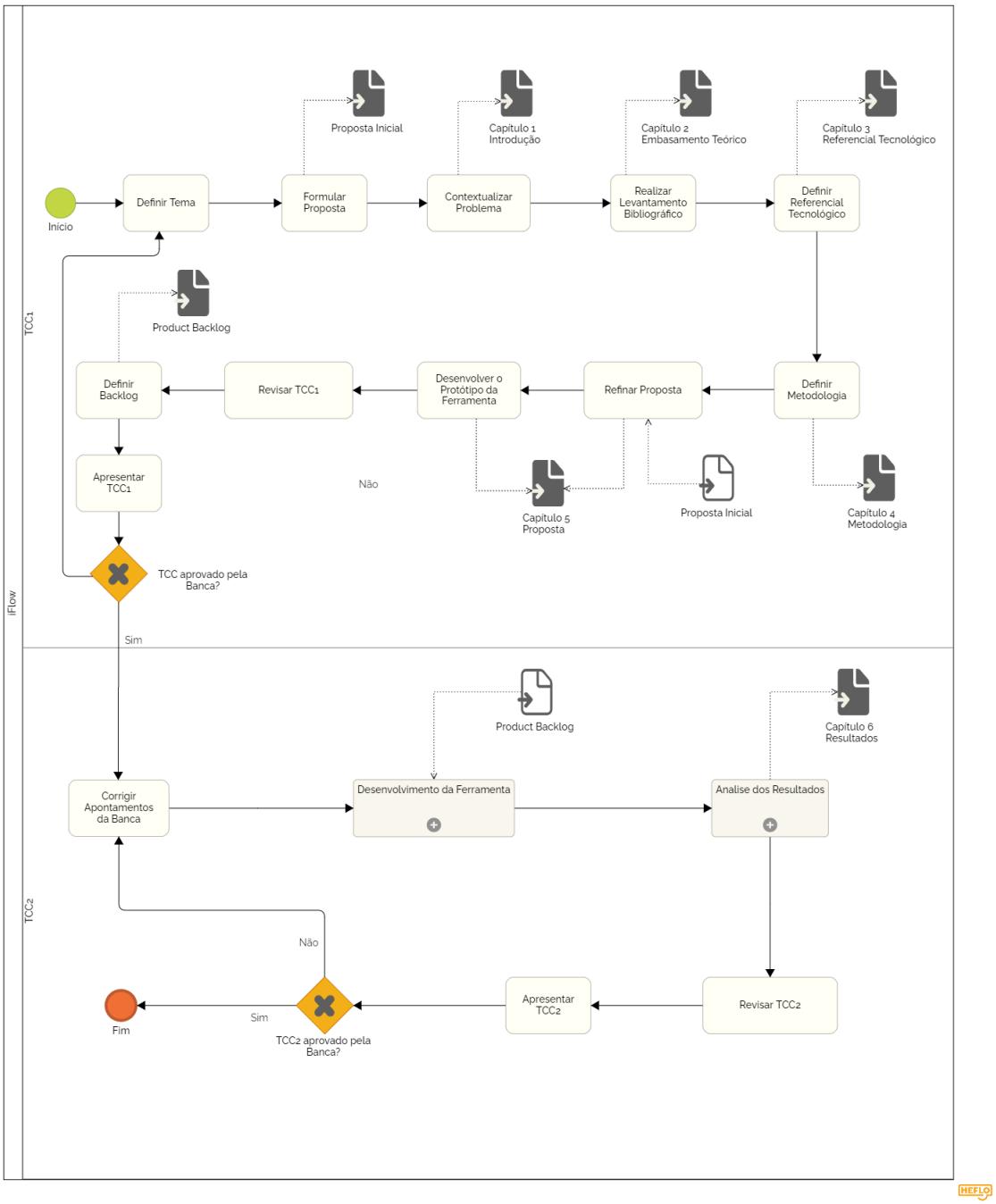
Fonte: Autores, 2022.

4.2 Fluxo de Atividades

Para adequada condução deste trabalho, foram seguidas algumas etapas. Com o intuito de melhorar a visualização e evidenciar essas etapas, foi elaborado o modelo, Figura 12, com base na notação *BPMN*[1].

[1]Notação gráfica criada pelo grupo conhecido como *BPMI* no ano de 2002 para representação de processos de negócio.

Figura 12 – Fluxo de Atividades do TCC



Fonte: Autores, 2022

1. **Definir Tema**: atividade destinada à escolha do tema para o desenvolvimento do trabalho. Foram apresentados diversos temas de interesse aos orientadores, e após várias discussões, foi escolhido o tema com maior perspectiva de desenvolvimento

e interesse entre as partes: a ferramenta *iFlow*;
Status: Concluída;

2. **Formular Proposta**: atividade que visa o fechamento de escopo, bem como contextualizar; identificar questões de pesquisa; estabelecer os objetivos, e conferir justificativas;
   Status: Concluída;
   Resultado: Capítulo Introdução (1);

3. **Contextualizar Problema**: atividade que consistiu em obter uma visão mais ampla da área de interesse deste trabalho, no caso, a Engenharia de Requisitos;
   Status: Concluída;
   Resultado: Capítulo Introdução (1);

4. **Realizar Levantamento Bibliográfico**: atividade relevante, e que permitiu consultar bases científicas, e investigar a literatura especializada sobre a área de Engenharia de Requisitos a afins. Outros detalhes, inerentes a essa atividade, constam na Seção 4.3;
   Status: Concluída;
   Resultado: Capítulo Embasamento Teórico (2);

5. **Definir Referencial Tecnológico**: atividade que compreende a definição dos apoios tecnológicos, no intuito de realizar o trabalho como um todo;
   Status: Concluída;
   Resultado: Capítulo Referencial Tecnológico (3);

6. **Definir Metodologia**: atividade, de suma importância, em que os autores se dedicaram ao desenho de um modelo que viabilizasse a execução do trabalho na totalidade, procurando definir passos a serem cumpridos, e esquematizando-os em atividades;
   Status: Concluída;
   Resultado: Fluxo de Atividades apresentado nesta seção (4.2);

7. **Refinar Proposta**: atividade que visou uma maior compreensão da proposta, procurando refiná-la e detalhá-la;
   Status: Concluída;
   Resultado: Capítulo Proposta, sendo esse um capítulo apresentado na monografia que compreende a primeira etapa do TCC. Alguns detalhes, inerentes à proposta, constam no Capítulo *iFlow* (5);

8. **Desenvolver o Protótipo da Ferramenta**: atividade destinada à elaboração de um protótipo de alta fidelidade, cujo objetivo é demonstrar, de forma mais clara, a proposta da ferramenta *iFlow*. O protótipo pode ser entendido ainda como uma prova de conceito, em relação à proposta, visto que acorda vários detalhamentos em termos de funcionalidades e fluxos pretendidos na ferramenta;
   Status: Concluída;
   Resultado: Capítulo *iFlow* (5);

9. **Revisar TCC1**: atividade que procura revisar cada aspecto do TCC, incluindo escrita da monografia, e elaboração de artefatos inerentes ao escopo de trabalho do TCC1. A revisão usa iterações, tanto entre os autores, quanto dos autores com os orientadores;
   Status: Concluída;
   Resultados: própria Monografia e artefatos em geral;

10. **Definir *Backlog***: atividade relevante para iniciar o desenvolvimento da ferramenta. A ideia é elencar de forma clara e objetiva, os principais requisitos da ferramenta;
    Status: Concluída;
    Resultado Esperado: *Backlog* do Produto;

11. **Apresentar TCC1**: atividade destinada à aprovação da proposta, junto à Banca Avaliadora;
    Status: Concluída;

12. **Corrigir Apontamentos da Banca**: atividade que procura, com base nos *feedbacks* conferidos pela Banca Avaliadora, revisar tanto a escrita da monografia, quanto a elaboração de artefatos;
    Status: Concluída;
    Resultados Esperados: própria Monografia e artefatos em geral evoluídos;

13. **Desenvolvimento da Ferramenta**: subprocesso que consiste no desenvolvimento da ferramenta. O detalhamento deste subprocesso encontra-se na Figura 13. As subatividades serão explicadas na Seção 4.4.1, na qual é definida a Metodologia de Desenvolvimento;
    Status: Concluída;
    Resultado Esperado: Ferramenta *iFlow*;

14. **Análise dos Resultados**: subprocesso que consiste em aplicar Pesquisa-Ação, visando coletar métricas qualitativas, e definindo um plano de ação, com base nessas métricas. Nesse processo, validou-se a ferramenta com apoio de testes e preenchimento de questionários junto ao público alvo. O detalhamento deste subprocesso encontra-se na Figura 14. As subatividades serão explicadas na Seção 4.5, onde é definida a Metodologia de Análise de Resultados;
Status: Concluída;
Resultados Esperados: Pesquisa-Ação e Ferramenta *iFlow* Evoluída;

15. **Revisar TCC2**: atividade que visa revisar cada aspecto do TCC, incluindo escrita da monografia, e elaboração de artefatos inerentes ao escopo de trabalho do TCC2. Foi utilizado o mesmo processo iterativo, conduzido para o escopo do TCC1, e
Status: Concluída;
Resultados Esperados: Monografia e artefatos em geral, em suas versões finais;

16. **Apresentar TCC2**: atividade destinada à aprovação do trabalho na totalidade, junto à Banca Avaliadora;
Status: Concluída;

4.3 Levantamento Bibliográfico

O Levantamento Bibliográfico foi realizado de modo a buscar os problemas inerentes à Engenharia de Requisitos. Não foi desenvolvida uma *string* de busca, mas o objetivo foi encontrar fontes confiáveis para o embasamento deste Trabalho de Conclusão de Curso.

Os principais autores e artigos usados tiveram como base referências usadas em disciplinas anteriores, cursadas durante a graduação. As palavras-chave utilizadas para guiar as pesquisas foram as mesmas usadas para as definições contidas no Embasamento Teórico (Capítulo 2), tais como: Engenharia de Requisitos, Elicitação, Modelagem, *House of Quality*, entre outros.

Vale ressaltar que os termos correspondentes foram utilizados na língua inglesa para haver uma maior quantidade de referências no contexto proposto. Obteve-se um total de 27 referências, dentre artigos, livros e sítios que podem ser mais bem visualizados no Documento de Referências (MOREIRA; CAIXETA, 2022c) usado para a centralização e manipulação da bibliografia do trabalho. Destaca-se, também, que essas foram as

referências levantadas em um primeiro momento, mas que a quantidade e as citações completas podem ser visualizadas nas Referências ao final do trabalho.

Para a seleção das referências, para a pesquisa bibliográfica, os seguintes itens foram utilizados como critérios de exclusão/inclusão:

- Excluem-se artigos em idiomas diferentes de inglês e português;
- Excluem-se artigos que não falam a respeito dos processos de Engenharia de Requisitos;
- Excluem-se artigos que não se aplicam ao contexto de *software*;
- Incluem-se artigos que possuem as definições necessárias;
- Incluem-se artigos a falar sobre a Engenharia de Requisitos e seus processos;
- Incluem-se artigos a falar sobre Elicitação, Modelagem, Análise, e Rastreabilidade;
- Incluem-se artigos a falar sobre *House of Quality*, e
- Incluem-se artigos que falem sobre as Estruturas de Dados.

4.4 Metodologia de Desenvolvimento

No que tange a metodologia voltada ao desenvolvimento prático da ferramenta, escolheu-se, por afinidade da dupla, a combinação de duas metodologias, *Scrum* & *Kanban*, as quais foram adaptadas em função do perfil do trabalho.

O *Scrum* é uma metodologia ágil para desenvolvimento de produtos complexos que mudam os requisitos rapidamente. O seu desenvolvimento é dado por uma série de iterações chamadas *Sprint*, que no caso desse projeto, foram semanais. A cada *Sprint* é realizada uma reunião de planejamento (*Sprint Planning*) e de revisão do trabalho (*Sprint Review & Restrospective*), com foco na melhoria contínua do processo e iteração entre as pessoas. Para promover a transparência e o alinhamento do trabalho, foram utilizados dois artefatos, o *Backlog* do Produto (*Product Backlog*) e o *Backlog* da *Sprint* (*Sprint Backlog*), nos quais são descritos, respectivamente, o escopo do produto e o escopo da *Sprint* (CAROLI, 2021).

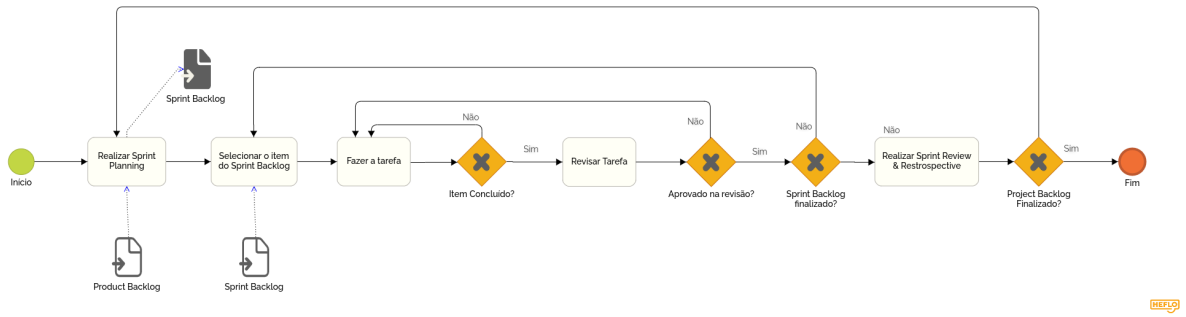
O *Kanban* é uma metodologia que visa um alinhamento mais claro de uma equipe, além de dar visibilidade ao que está sendo elaborado, proporcionando um ambiente

favorável à comunicação, com o menor ruído possível. Além disso, garante uma carga de trabalho adequada, considerando a capacidade produtiva da equipe. Um ponto essencial dessa metodologia é o quadro que deixa as tarefas e seus *status* visíveis para todo o time, pontuando em qual parte do fluxo de trabalho cada tarefa se encontra (ANDERSON; CARMICHAEL, 2016).

4.4.1 Processo de Desenvolvimento

O processo de desenvolvimento escolhido pode ser visualizado na Figura 13, elaborado com base na notação *BPMN*, a partir do detalhamento do subprocesso **Desenvolvimento da Ferramenta**, apresentado na Figura 12, consistindo das seguintes etapas:

Figura 13 – Fluxo de Atividades do Subprocesso Desenvolvimento da Ferramenta *iFlow*



Fonte: Autores, 2022.

1. **Realizar *Sprint Planning***: a partir do *Product Backlog*, definido conforme item 10 da Seção 4.2, as atividades são selecionadas consoante a sua prioridade e pontos definidos para execução da mesma. As tarefas selecionadas consideram a capacidade produtiva dos autores na *Sprint* e a gestão de riscos para a entrega do produto. Ao final, têm-se o *Sprint Backlog*;

2. **Selecionar o item do *Sprint Backlog***: a partir do *Sprint Backlog*, as tarefas são selecionadas para serem executadas pelos autores;

3. **Fazer a tarefa**: consiste em desenvolver a tarefa selecionada;

4. **Revisar tarefa**: consiste no processo de revisão por parte, dentre os autores, do membro que não atuou no desenvolvimento diretamente, para verificar se os

padrões de desenvolvimento da comunidade foram seguidos, e se atende ao que foi definido no *Product Backlog*;

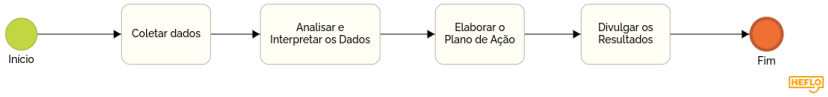
5. **Realizar *Sprint Review & Restrospective***: com a conclusão da *Sprint*, essa atividade elenca os pontos positivos e negativos da *sprint*, além de revisar as atividades entregues.

4.5 Metodologia de Análise de Resultados

Como definido na Seção 4.1, o método orientado para o processo de análise dos resultados do trabalho é a Pesquisa-Ação. De acordo com Gil et al. (2002), a pesquisa-ação ocorre com muitas oscilações entre as fases, determinada pela dinâmica do grupo de pesquisadores em seu relacionamento com a situação pesquisada. Além disso, a pesquisa-ação apresenta uma série de ações desordenadas no tempo, considerando os seguintes passos: a) fase exploratória; b) formulação do problema; c) construção de hipóteses; d) realização do seminário; e) seleção da amostra; f) coleta de dados; g) análise e interpretação dos dados; h) elaboração do plano de ação; e i) divulgação dos resultados.

Já conferindo uma versão adaptada do protocolo, que pode ser mais bem visualizada na Figura 14, a pesquisa-ação utilizada neste trabalho foi conduzida com base nas seguintes etapas:

Figura 14 – Fluxo de Atividades do Subprocesso Análise dos Resultados da Ferramenta *iFlow*



Fonte: Autores, 2022.

- **Coletar dados**: nesta etapa, realiza-se a coleta de dados qualitativos através da interação dos usuários com a ferramenta;
- **Analisar e interpretar dos dados**: nesta etapa, busca-se examinar os dados coletados e, em seguida, realizar a explanação dos resultados adquiridos;
- **Elaborar plano de ação**: pretende-se elaborar um plano de ação que visa mitigar os problemas encontrados a partir dos dados analisados, e

- **Divulgar resultados**: para validação, os resultados obtidos e o plano de ação definido foram documentados e validados com a banca, bem como junto ao público alvo, no intuito de compreender sobre as contribuições da ferramenta no escopo de atuação da mesma.

O uso dessa metodologia deu-se por coleta de dados com os *stakeholders* que avaliaram a experiência do usuário na ferramenta, para validar se a ferramenta cumpre com os **objetivos específicos** 3 e 4. Realizou-se ainda testes que buscaram validar se os requisitos propostos foram cumpridos e, principalmente, se o **objetivo específico** 2 foi satisfeito.

4.6 Cronogramas

As Tabelas 1 e 2 apresentam os cronogramas do TCC1 e do TCC2, respectivamente. Nas tabelas, as atividades são descritas de acordo com suas datas de implementação, baseando-se no fluxo de atividades descrito na Seção 4.2, e considerando uma temporalidade contínua e não baseada apenas em período letivo.

Tabela 1 – Cronograma de Atividades do TCC1

Atividade	Jan/2022	Fev/2022	Mar/2022	Abr/2022	Mai/2022
Definir Tema	X				
Formular Proposta		X			
Contextualizar Problema		X			
Realizar Levantamento Bibliográfico			X		
Definir Referencial Tecnológico			X		
Definir Metodologia			X	X	
Refinar Proposta			X	X	
Desenvolver o Protótipo da Ferramenta				X	
Revisar TCC1				X	
Definir *Backlog*				X	
Apresentar TCC1					X

Fonte: Autores, 2022.

Tabela 2 – Cronograma de Atividades do TCC2

Atividade/Subprocesso	Mai/2022	Jun/2022	Jul/2022	Ago/2022	Set/2022
Corrigir Apontamentos da Banca	X				
Desenvolvimento da Ferramenta	X	X	X	X	X
Coleta de Dados				X	X
Análise dos Resultados				X	X
Revisar TCC2					X
Apresentar TCC2					X

Fonte: Autores, 2022.

4.7 Resumo do Capítulo

Neste capítulo, foram apresentadas as escolhas metodológicas definidas para o desenvolvimento deste Trabalho de Conclusão de Curso. Desta forma, foram evidenciadas as metodologias da fase de pesquisa, de desenvolvimento, e de análise dos resultados, a fim de identificar e categorizar os pontos mais importantes em cada uma delas. A seguir, visando retratar as etapas de desenvolvimento do trabalho, foram apresentados os fluxos de atividades que orientaram o desenvolvimento do trabalho com as respectivas descrições correspondentes das atividades ilustradas nesses fluxos. Por fim, os cronogramas das duas fases de realização do trabalho (TCC1 e TCC2) foram apresentados.

5 *iFlow*

Este capítulo visa detalhar o *iFlow*, o qual é uma ferramenta que se baseia no processo proposto pela Engenharia de Requisitos para gerenciar e dar suporte àqueles que estão inseridos nesse contexto. Primeiramente, tem-se uma seção apresentando a contextualização do *iFlow* (Seção 5.1). Em seguida, apresenta-se a ferramenta *iFlow* (Seção 5.2). Adicionalmente, descreve-se a arquitetura de *software* (Seção 5.3), utilizada para o desenvolvimento da ferramenta, além das especificidades dos módulos do sistema, seguindo para a apresentação da identidade visual (Seção 5.4). Logo depois, apresentam-se os principais diagramas da ferramenta (Seção 5.5). Posteriormente, tem-se a apresentação das telas desenvolvidas do *iFlow* e do *backlog* da ferramenta (seções 5.6 e 5.7). Por fim, têm-se as telas desenvolvidas do *iFlow* e o resumo do capítulo (Seção 5.8 e 5.9).

5.1 Contextualização

A principal justificativa para o desenvolvimento da ferramenta *iFlow* diz respeito a buscar minimizar e evidenciar os problemas associados aos processos da Engenharia de Requisitos. Vale ressaltar, que não existem, até o momento da publicação deste Trabalho de Conclusão de Curso, trabalhos correlatos que tenham o escopo de atuação e busquem solucionar os problemas da área da forma proposta.

Reproduzindo um papel essencial no desenvolvimento de produtos de *software*, a Engenharia de Requisitos fornece uma maneira de entender e descrever problemas que precisam de uma solução de *software*. Além disso, é um ramo que está inteiramente ligado ao ciclo de desenvolvimento do *software* (ELLIOTT, 2012). Dessa forma, por se tratar de uma área negligenciada por muitos, foi realizada a construção de uma ferramenta que oferece apoio tecnológico, viabiliza e semi-automatiza as etapas da Engenharia de Requisitos, com o propósito de facilitar e melhorar estes processos.

A seguir, tem-se uma descrição mais aprofundada da ferramenta desenvolvida neste Trabalho de Conclusão de Curso.

5.2 Sobre a Ferramenta *iFlow*

A ferramenta *iFlow* tem o objetivo de viabilizar e semi-automatizar o processo da Engenharia de Requisitos, de modo a embasar a construção de um produto de *software* e evitar que retrabalhos e desperdícios de tempo ocorram devido ao mau planejamento. Neste sentido, algumas automatizações foram realizadas, de modo a estabelecer e se apoiar no conceito de um *Minimum Viable Product* (MVP). Além disso, a ferramenta confere ênfase às etapas da Engenharia de Requisitos. Dessa forma, a ferramenta é composta por algumas etapas, dentre elas: a) pré-rastreabilidade; b) elicitação; c) modelagem; d) *house of quality*; e e) pós-rastreabilidade.

5.2.1 Pré-rastreabilidade

Nesta primeira etapa, o foco da ferramenta foi em aplicar os conceitos propostos em pré-rastreabilidade (Seção 2.2.1), e iniciar o processo da Engenharia de Requisitos, descrevendo as primeiras definições do produto de *software* em questão, iniciando o levantamento dos requisitos.

Com esse objetivo, os artefatos usados para esta etapa são o *5W2H* (Seção 2.2.1.1), que sintetiza a ideia inicial do produto, e o *Rich Picture* (Seção 2.2.1.2), que transcreve visualmente os atores e o fluxo de informações envolvidos no processo.

5.2.2 Elicitação

Estabelecida a visão geral do produto de *software*, a ferramenta parte para a segunda etapa do processo, que levanta o maior número possível de requisitos. A intenção é que se colete diferentes pontos de vistas em diferentes contextos e com opiniões distintas, visando a aquisição de definições do produto mais ricas que agregue mais valor para os *stakeholders*.

Dessa forma, na ferramenta, existem diversas propostas de métodos que proporcionam modelos que auxiliam na elicitação dessas funcionalidades. Vale salientar que tais métodos não são obrigatórios para que o usuário siga para a próxima etapa, mas evidencia-se a importância desta etapa, além de uma confirmação de que as funcionalidades levantadas são suficientes para a construção do produto de *software*.

No contexto de cada ferramental, o usuário tem a possibilidade de adicionar diferentes fontes de dados para compor o artefato, sendo: imagens, áudios e documentos

de texto. O intuito é que, a partir dessas informações, novas funcionalidades sejam adquiridas.

Neste sentido, os artefatos disponibilizados para viabilizar a etapa da elicitação são:

- *Brainstorming* (Seção 2.3.1), que coleta o maior número de ideias acerca do produto de *software*;
- Questionário (Seção 2.3.2), que tem o objetivo de coletar o maior número de informações a respeito do tema em questão;
- Protótipo de Baixa Fidelidade (2.3.3), que ajuda, visualmente, na elicitação dos requisitos;
- *Storytelling* (Seção 2.3.4), que extrai as *features* do sistema em um determinado contexto;
- Entrevista (Seção 2.3.5), que, de certa forma, traz uma visão mais concreta acerca de um tema pelo fato de se ter uma comunicação mais natural entre os envolvidos, e
- Análise de Protocolo (Seção 2.3.6), que sintetiza os pensamentos de um usuário e, a partir disso, tem-se a extração de possíveis requisitos do sistema.

5.2.3 Modelagem

Com as funcionalidades levantadas, a ferramenta segue para a próxima etapa do processo, que visa colocar em modelos estruturados todas as ideias obtidas na etapa anterior (Seção 5.2.2). Diferentemente da elicitação, que tem uma visão muito aberta e livre para proporcionar um espaço saudável para que a criatividade seja o principal foco, este nível visa chegar em um ponto mais concreto e consistente para a construção de uma aplicação. O propósito, é gerar modelos capazes de representar características ou comportamentos do produto de *software*, que possam guiar todo o processo de desenvolvimento.

Vale destacar a importância deste processo, pois concluída esta etapa, o usuário deve viabilizar uma forma para a construção do produto de *software*. Neste ponto, deve-se ter uma visão mais concreta e definitiva da aplicação que se almeja desenvolver.

Esta etapa é imprescindível, pois aqui são gerados artefatos essenciais para a aplicação que será desenvolvida. Sendo assim, os artefatos usados para esta etapa são o *Backlog* (Seção 2.4.1), que concentra a lista de defeitos, melhorias solicitadas pelo cliente, funcionalidades, entre outros itens que expressam os requisitos do produto; o NFR *Framework* (Seção 2.4.2), que sintetiza os requisitos não funcionais do sistema no formato de um grafo, e os Léxicos (2.4.3), que transcrevem todos os vocábulos utilizados no domínio de uma aplicação.

5.2.4 Priorização

Essa etapa é de suma importância dentro do contexto de Engenharia de Requisitos, pois nela que é definido o Produto Mínimo Viável (Seção 2.6) a partir da *House of Quality* (Seção 2.7), onde se tem a correlação dos requisitos funcionais e não funcionais e seus *softgoals*. Aqui, determina-se quais são as principais funcionalidades do produto de *software*, que devem ser validadas e efetivadas pelo usuário final. Dessa forma, para a geração do Produto Mínimo Viável, faz-se o uso de Grafos (Seção 2.7.1), utilizando-se de um algoritmo de Menor Caminho (Dijkstra).

5.3 Arquitetura da Ferramenta *iFlow*

Conforme é observado na Figura 15, a arquitetura de *software* do *iFlow* inclui a integração de dois módulos: o *front-end* e o *back-end*.

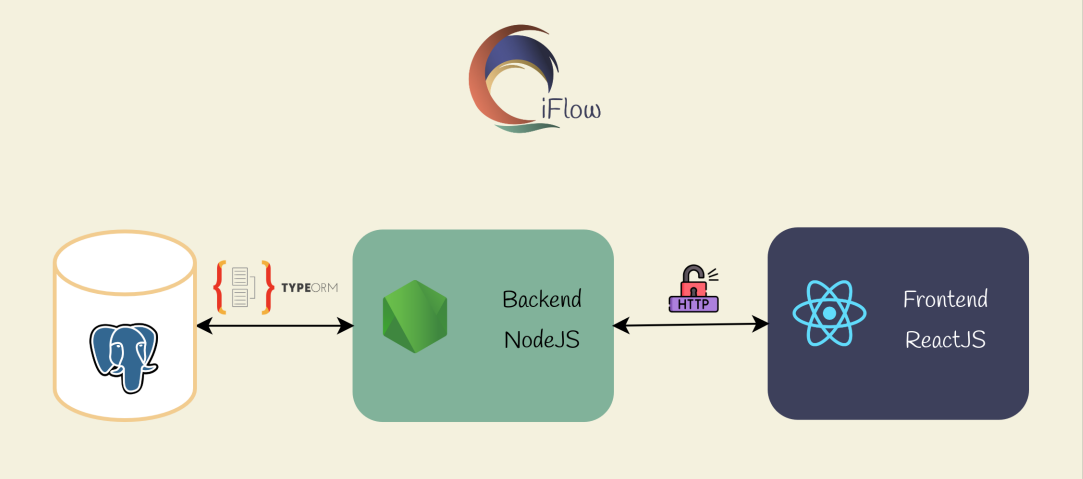
5.3.1 *Front-end*

O *Front-end* tem a função de realizar as interações com o usuário e com o *back-end*. Além disso, é responsável pela *interface* da ferramenta utilizando recursos *HTML* e *CSS* com o *framework React*. Sendo assim, essas tecnologias foram utilizadas devido ao grande suporte que elas oferecem para a construção de *interfaces* de usuário interativas, descritas com mais detalhes no Capítulo 3.

5.3.2 *Back-end*

O *Back-end* tem a função de realizar a conexão com o banco de dados; armazenar os dados que são consumidos ou manipulados pelo *software*, e por realizar o tratamento dos dados. Dessa forma, foi utilizada a tecnologia *Node.js* visando realizar a comunicação

Figura 15 – Arquitetura do *Software* da ferramenta *iFlow*



Fonte: Autores, 2022.

entre as camadas de banco de dados e de *front-end*. A tecnologia é descrita com mais detalhes no Capítulo 3.

5.4 Identidade Visual da Ferramenta *iFlow*

A Identidade Visual é um conjunto de elementos gráficos e visuais, com o intuito de mostrar os principais elementos do produto e seus valores. Com o intuito de dar uma identidade visual para a ferramenta *iFlow*, foi elaborado um manual que elenca os logotipos, as cores, a tipografia e os símbolos definidos para o projeto. Esse manual está descrito com maiores detalhes no Apêndice B.

5.4.1 Logotipo

O logotipo da ferramenta é a representação de ondas buscando transmitir a noção do processo proveniente da Engenharia de Requisitos, conforme pode-se observar na Figura 16.

Figura 16 – Logotipo da Ferramenta *iFlow*

Fonte: Autores, 2022.

5.5 Modelagem da Ferramenta *iFlow*

5.5.1 Diagramas de Banco de Dados

Os diagramas de Banco de Dados visam modelar como os dados serão armazenados na Ferramenta *iFlow*. O DER (Figura 17) é a representação gráfica e a principal ferramenta de visualização dos relacionamentos entre as entidades do sistema. Já o DL (Figura 18) descreve como os dados serão armazenados no banco, além dos seus relacionamentos.

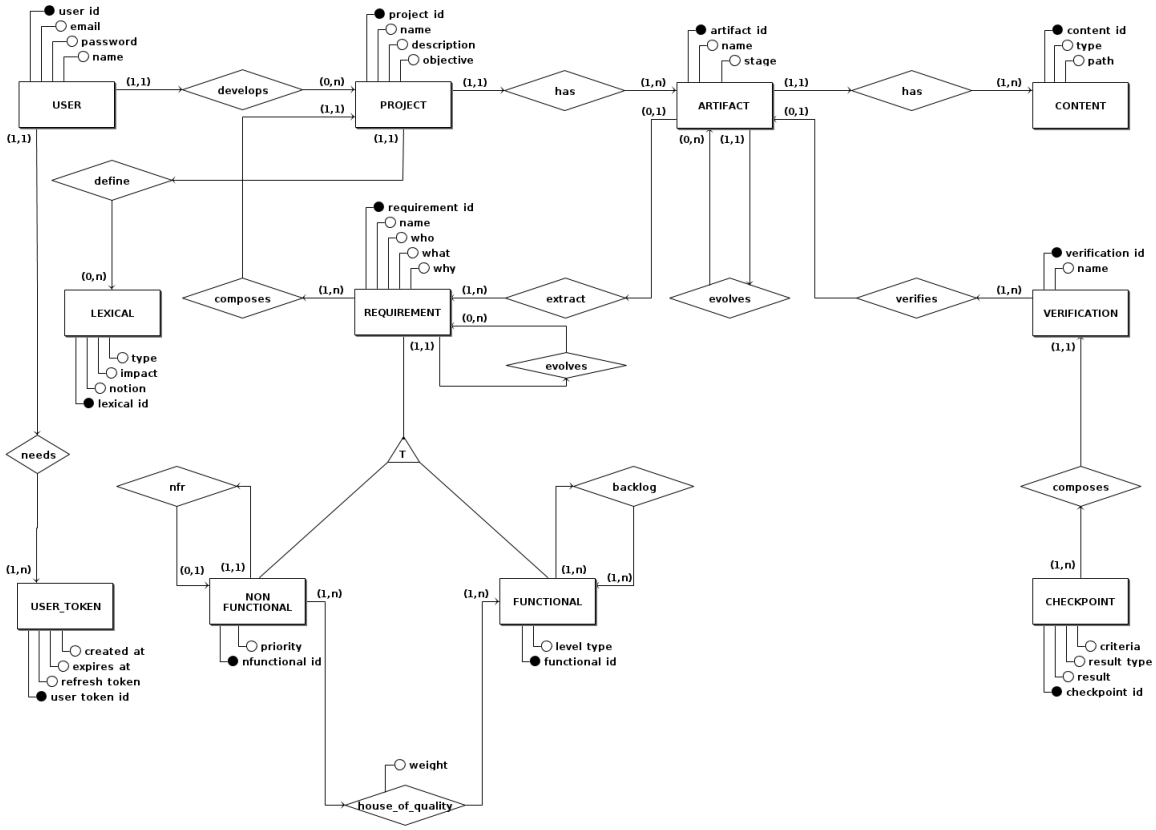
O código que cria a base de dados da Ferramenta *iFlow* está disponível no Apêndice C.

5.5.2 Diagrama de Pacotes

O Diagrama de Pacotes é representado como um esquema estático, possibilitando a visualização e a organização do projeto de maneira mais adequada e simplificada, proporcionando uma visão em módulos, facilitando o entendimento do projeto durante o desenvolvimento e em possíveis manutenções. Foram desenvolvidos a partir das melhores práticas e padrões adotadas por ambas as comunidades, *ReactJS* e *NodeJS*. Dessa forma, os diagramas das Figuras 19 e 21 procuram detalhar a organização interna dos pacotes, quanto aos diferentes módulos da Ferramenta *iFlow*, *Front-end* e *Back-end*.

Uma visualização ampliada foi adicionada para o detalhamento da pasta do diagrama de pacotes do *Back-end*, na Figura 20, para melhor visualização.

Figura 17 – DER do Banco de Dados do *iFlow*



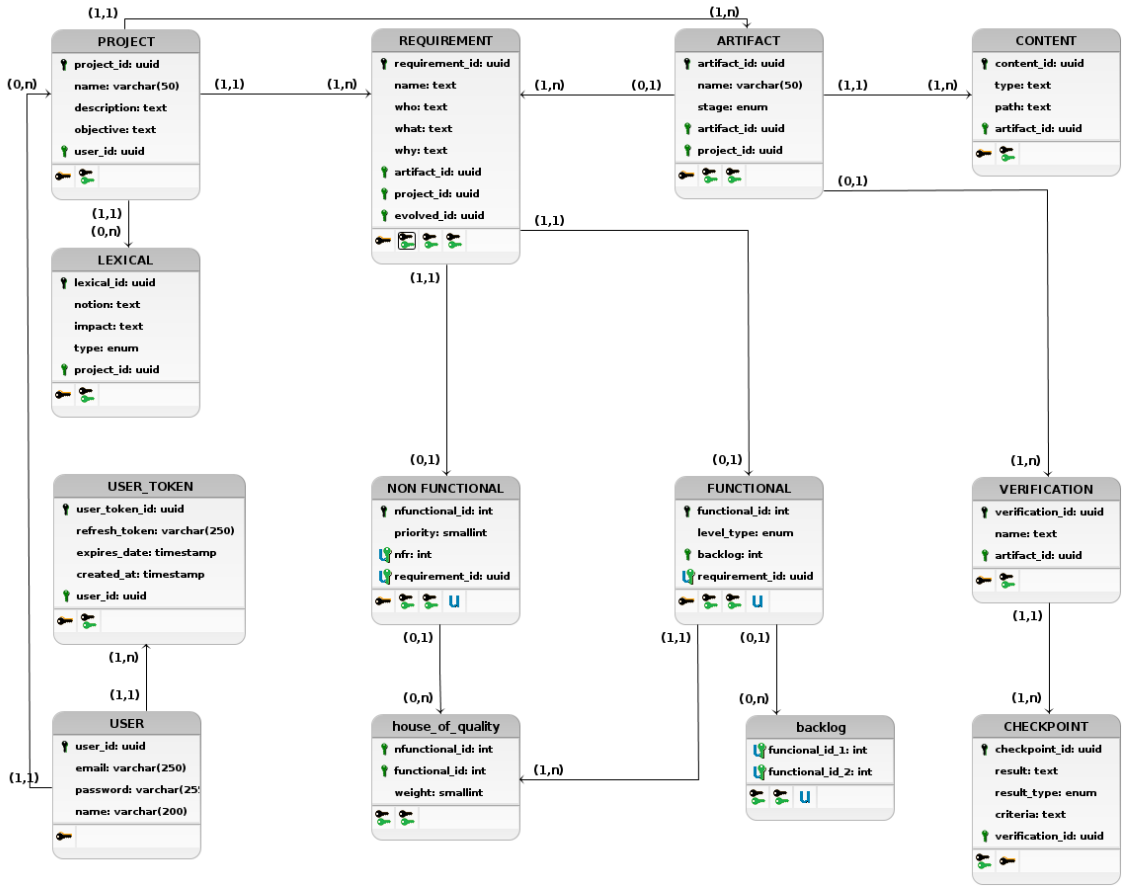
Fonte: Autores, 2022.

5.6 Protótipo de Alta Fidelidade

Um protótipo de alta fidelidade deve se aproximar ao máximo dos aspectos visuais e funcionais do produto final, incluindo o conteúdo, fluxo de navegação e interações. São muito utilizados para testes e validação com usuários, ou para apresentar uma ideia. Objetivamente, a ideia do protótipo é saber como a ferramenta funcionará (WALKER et al., 2002).

Com este intuito, foi elaborado um Protótipo de Alta Fidelidade navegável, na plataforma *Figma* (SHARMA; TIWARI, 2021). Como, a quantidade de telas geradas foi demasiadamente grande, foram disponibilizadas apenas algumas das muitas telas desenvolvidas, para evidenciar, ainda que em suma, qual o perfil do produto. Entretanto, para visualizar todas as telas e poder navegar e interagir, optou-se por disponibilizar o enlace para ser acessado *on-line*: Moreira e Caixeta (2022d).

Figura 18 – DL do Banco de Dados do *iFlow*



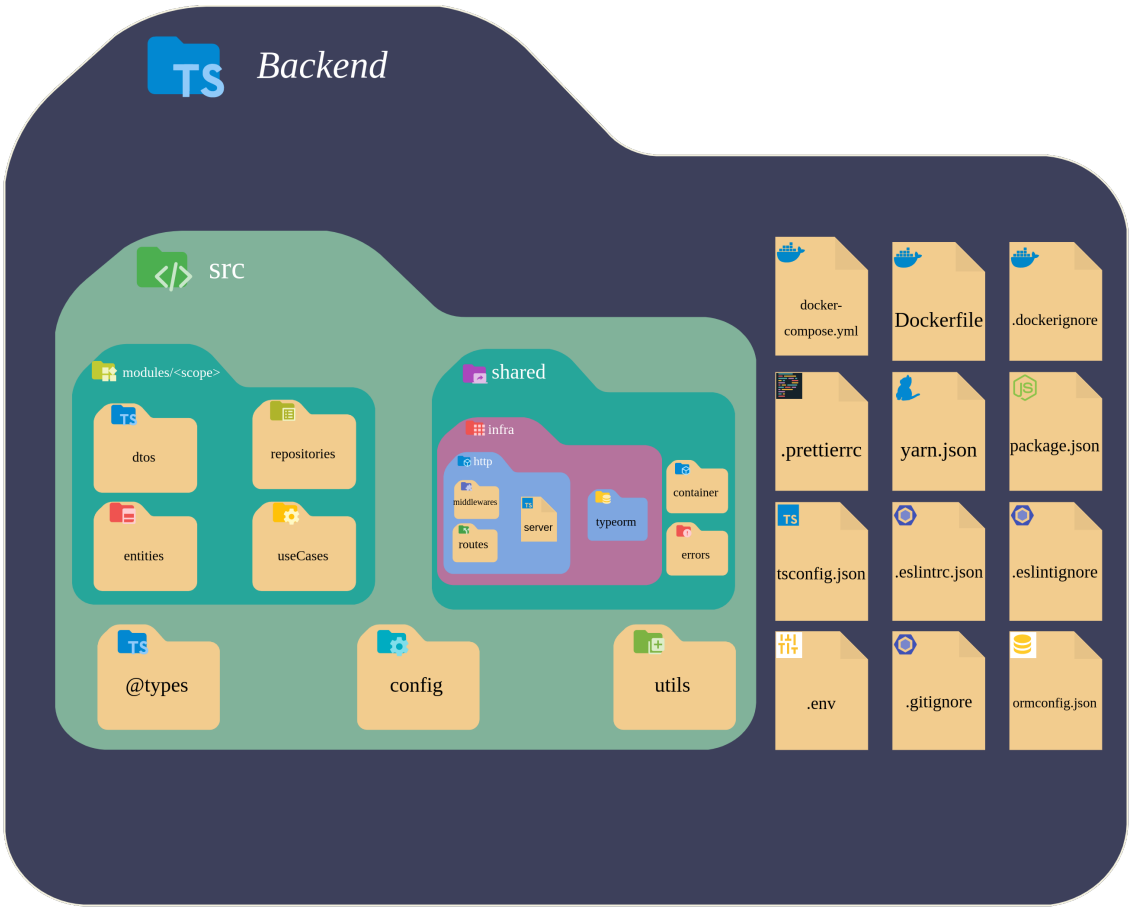
Fonte: Autores, 2022.

Estão elencadas a seguir as principais telas desenhadas para o *iFlow*:

- A Figura 22 trata da página onde estão elencadas todas as etapas necessárias para o desenvolvimento da Engenharia de Requisitos, proposto para a ferramenta *iFlow*;

- A Figura 23 trata da tela onde são listados os artefatos que já foram concluídos, que, neste caso, se trata dos gerados na elicitação do projeto em questão;
- A Figura 24 representa as telas de criação do *Backlog* do Produto e da *House of Quality*, que as etapas de suma importância dentro deste processo, e
- Por fim, tem-se a Figura 25 com a tela de renderização do Produto Mínimo Viável gerado, em sua versão preliminar.

Figura 19 – Diagrama de Pacotes do *Backend* do *iFlow*



Fonte: Autores, 2022.

5.7 *Backlog* do Produto

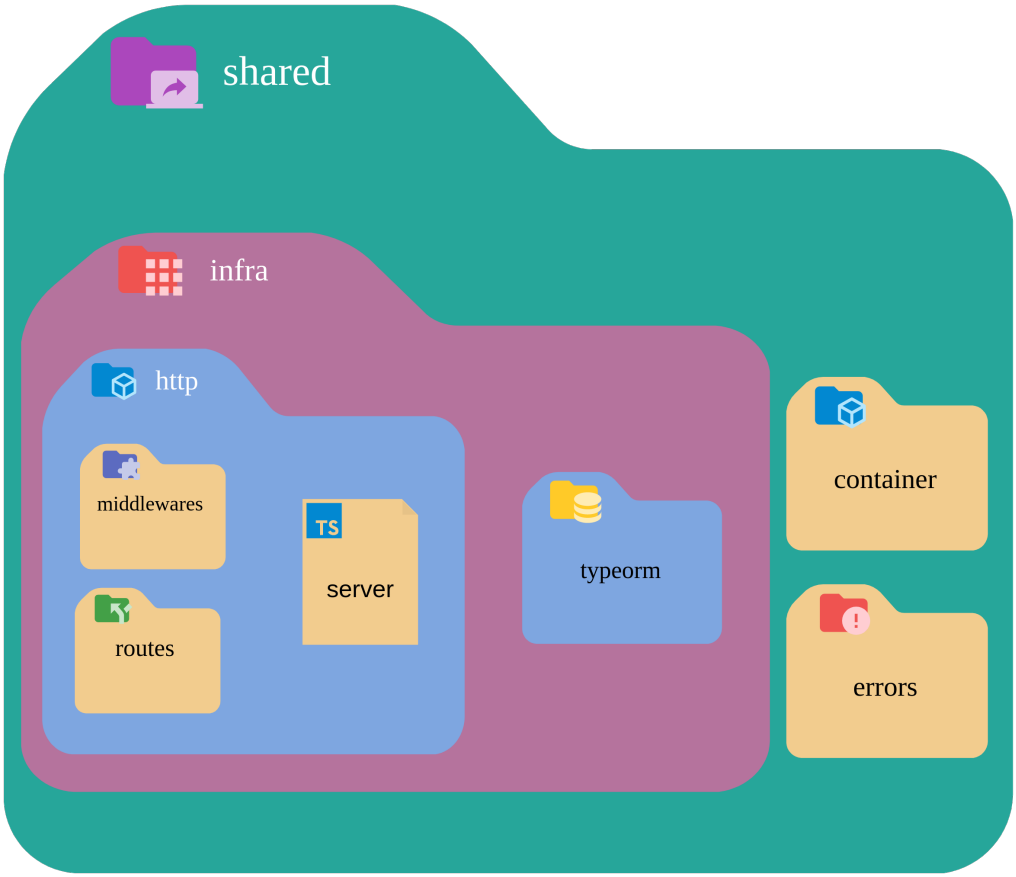
Com o intuito de guiar o processo de desenvolvimento da ferramenta *iFlow*, foi elaborado um *backlog* com os requisitos necessários para compor a aplicação. As definições, e correspondente tabela, foram colocadas no Apêndice A.

5.8 Versão Final — *iFlow*

5.8.1 Telas desenvolvidas

A versão final das telas não se difere muito em relação ao protótipo de alta fidelidade (Seção 5.6). Todavia, algumas telas tiveram que ser simplificadas, para que se pudesse desenvolver todas as telas indispensáveis ao projeto, ou foram melhoradas em

Figura 20 – Diagrama de Pacotes do *Backend* do *iFlow*: detalhamento da pasta *shared*



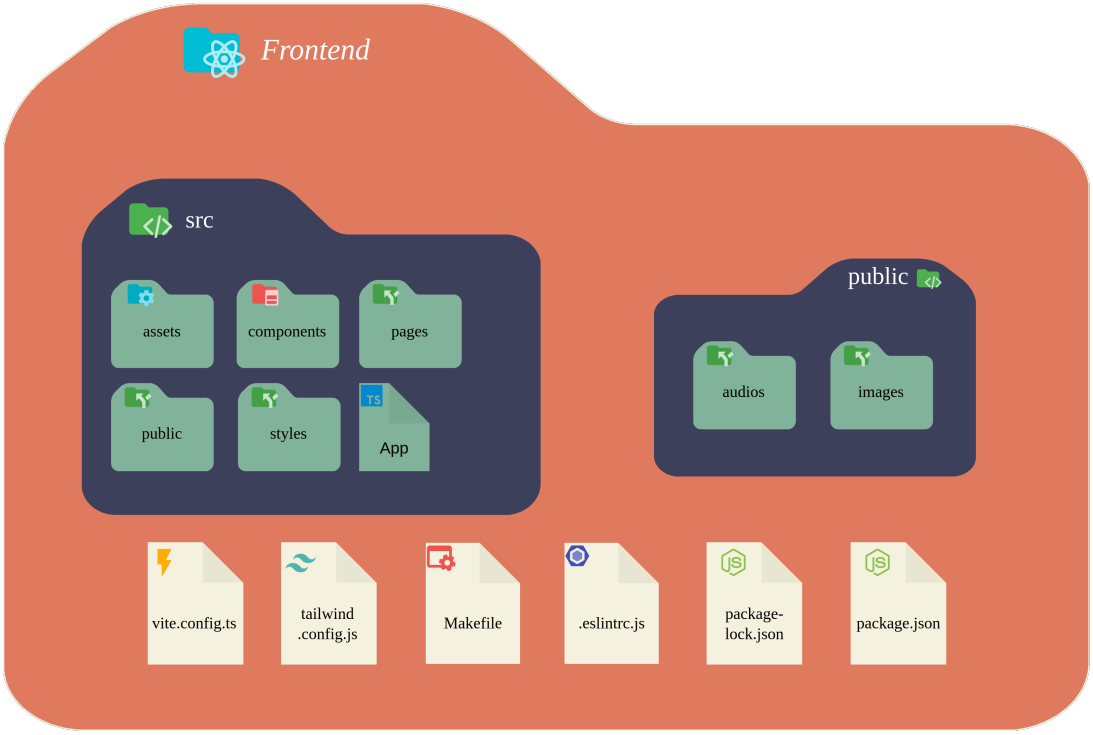
Fonte: Autores, 2022.

algum aspecto visual que não ficou tão satisfatório no protótipo de alta fidelidade.

Como foram desenvolvidas muitas telas, a seguir, foram elencadas as telas mais relevantes da ferramenta, correspondentes às apresentadas no protótipo de alta fidelidade (Seção 5.6), sendo elas:

- A Figura 26 retrata a tela das Etapas da Engenharia de Requisitos em sua versão final, mostrando semelhança em relação ao protótipo desenvolvido;
- A Figura 27 exibe os artefatos finalizados na etapa de elicitação, mostrando semelhança com o protótipo desenvolvido;
- A Figura 28 revela a criação de um novo artefato, nesse caso, um novo Questionário. Pode-se observar que essa tela possui algumas divergências em relação ao protótipo,

Figura 21 – Diagrama de Pacotes do *Frontend* do *iFlow*



Fonte: Autores, 2022.

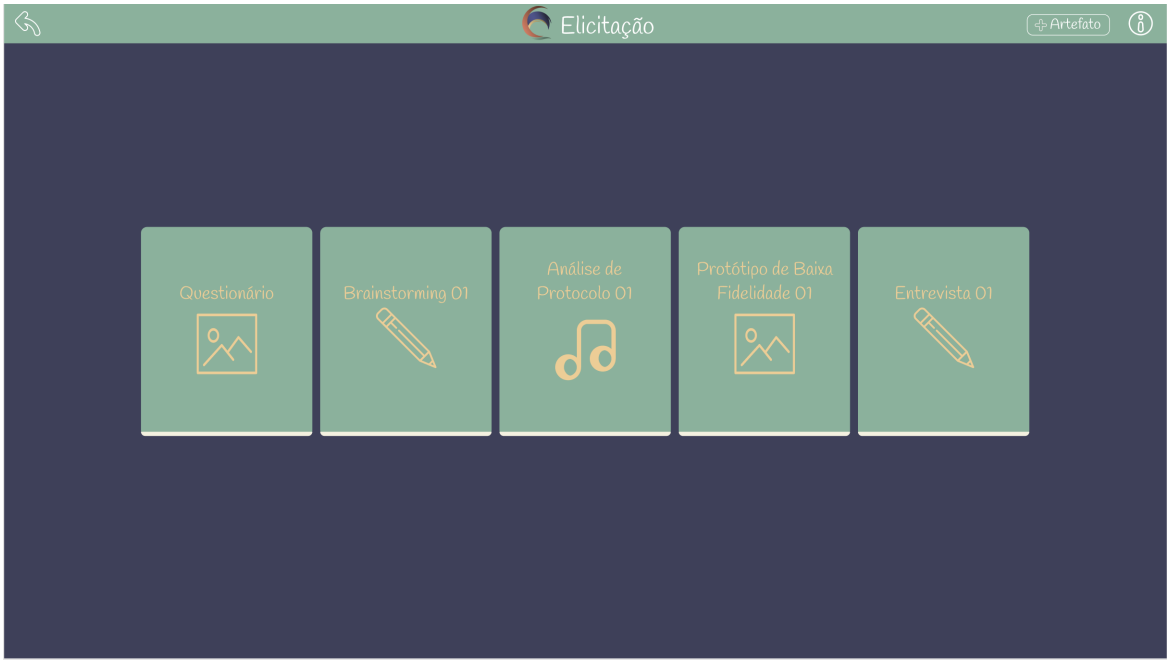
onde os requisitos são listados abaixo do conteúdo do artefato, possuindo uma visualização menos adequada. Tendo isso em vista, a melhora de visualização e interface do *iFlow* ficariam como trabalhos futuros;

- A Figura 29 representa a criação do *Backlog* do Produto em sua versão final. É uma tela de suma importância no contexto de Engenharia de Requisitos, sendo bem semelhante à desenvolvida no protótipo, além de ter a funcionalidade de arrastar o requisito funcional para a tabela do *Backlog*, e
- Por fim, tem-se a Figura 30, a qual representa a tela de criação da *House of Quality* em sua versão final. Pode-se reparar que esta tela se assemelha bastante com a desenvolvida no protótipo, possuindo a funcionalidade de arrastar os requisitos funcionais para as respectivas relações com os requisitos não funcionais advindos da etapa do NFR, definindo essa relação como fraca, média ou forte.

Figura 22 – Tela das Etapas da Engenharia de Requisitos do Protótipo

Fonte: Autores, 2022.

Figura 23 – Tela dos Artefatos Elaborados



Fonte: Autores, 2022.

Figura 24 – Telas de Criação do *Backlog* do Produto e da *House of Quality*, respectivamente

Fonte: Autores, 2022.

Figura 25 – Tela com o Produto Mínimo Viável da Aplicação em sua Versão Preliminar

Fonte: Autores, 2022.

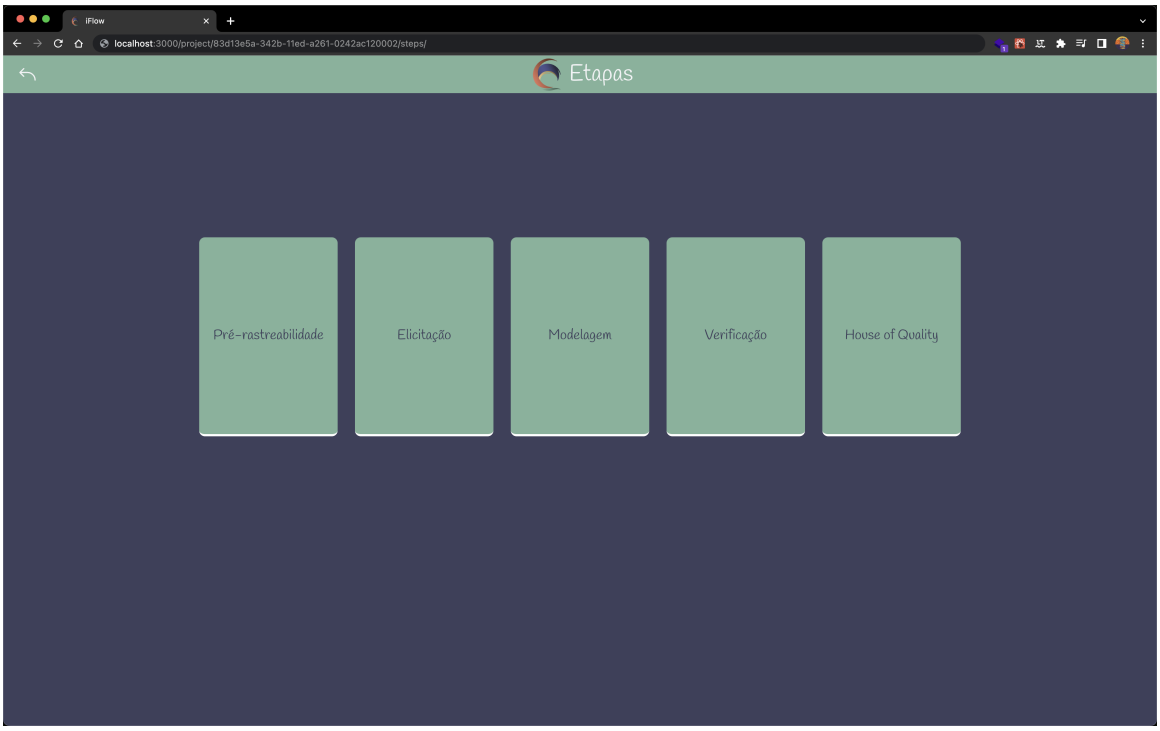
5.8.2 Priorização das funcionalidades do *iFlow*

A Engenharia de Requisitos trata de uma parte vital do processo de desenvolvimento de um *software*. Portanto, representa uma etapa minuciosa e meticulosa que visa concretizar desejos, pensamentos, demandas e requisitos concretos que possam ser codificados. Entretanto, representa também uma etapa que demanda tempo considerável para a sua correta execução.

Sabendo disso, para viabilizar as etapas do processo, e ainda obter um Produto Mínimo Viável em sua versão preliminar, o principal diferencial da ferramenta, foram priorizadas as funcionalidades mínimas para gerar insumos para tal feito, sendo elas:

- Toda a parte de geração de artefatos, tanto na pré-rastreabilidade quanto na elicitação, para que possibilitasse a geração de requisitos mais concisos e completos;
- Todas as funcionalidades relacionadas à geração do *Backlog* e do *NFR*, para que os requisitos fossem modelados e amadurecidos para serem usados posteriormente;
- A funcionalidade da *House of Quality* para poder viabilizar a modelagem dos requisitos como um grafo, conferindo insumos para a geração do Produto Mínimo

Figura 26 – Tela das Etapas em sua Versão Final



Fonte: Autores, 2022.

Viável em sua versão preliminar;

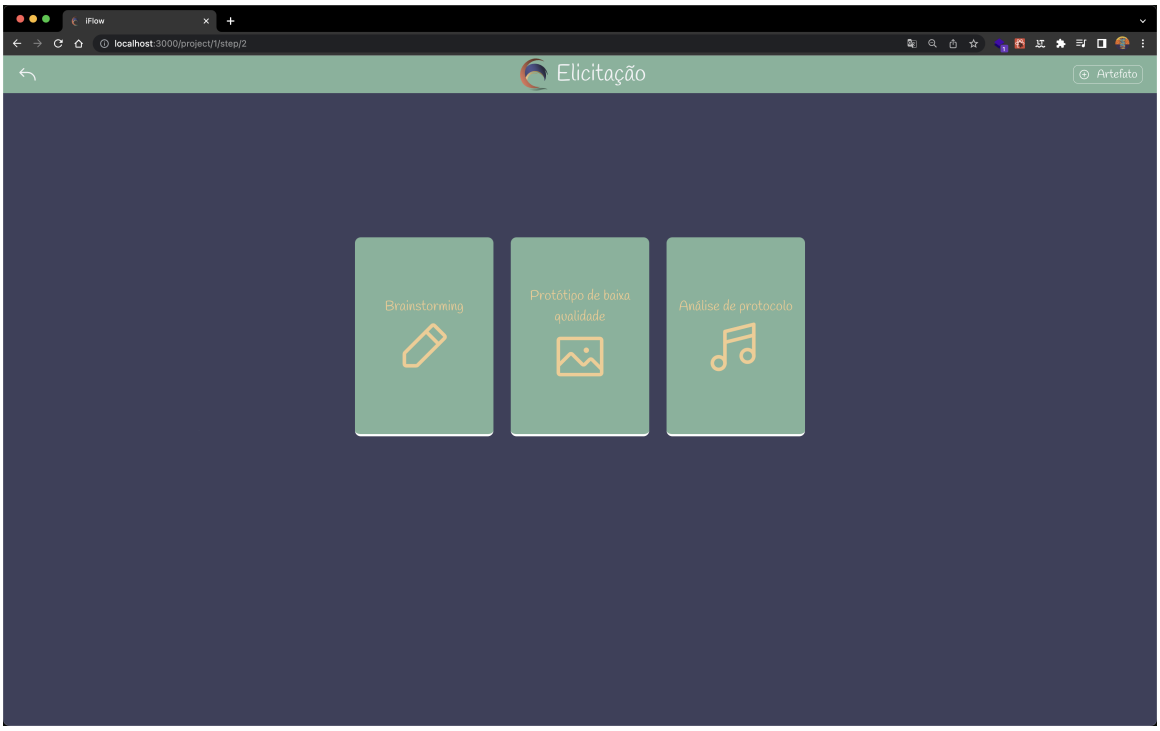
- O algoritmo de menor caminho necessário para viabilizar a geração do Produto Mínimo Viável em sua versão preliminar, e
- A parte do gerenciamento de usuário que viabilizasse controle, organização e segurança de tudo que está sendo desenvolvido na ferramenta.

Vale ressaltar que, outro ponto que guiou o desenvolvimento da ferramenta, foi o uso das melhores práticas de padrões e desenho de projeto da Engenharia de *Software*, para que fossem gerados códigos de qualidade, não apenas com o intuito de fazer a aplicação, mas de gerar algo que possa ser aperfeiçoado em trabalhos futuros, servindo de base de código para outros trabalhos.

5.8.3 Ajustes de Escopo

A partir do exposto na Seção 5.8.2, e conferindo ajustes no escopo do desenvolvimento da ferramenta, algumas *features* e *user stories* não foram concluídas. Isso era

Figura 27 – Tela dos Artefatos Elaborados em sua Versão Final

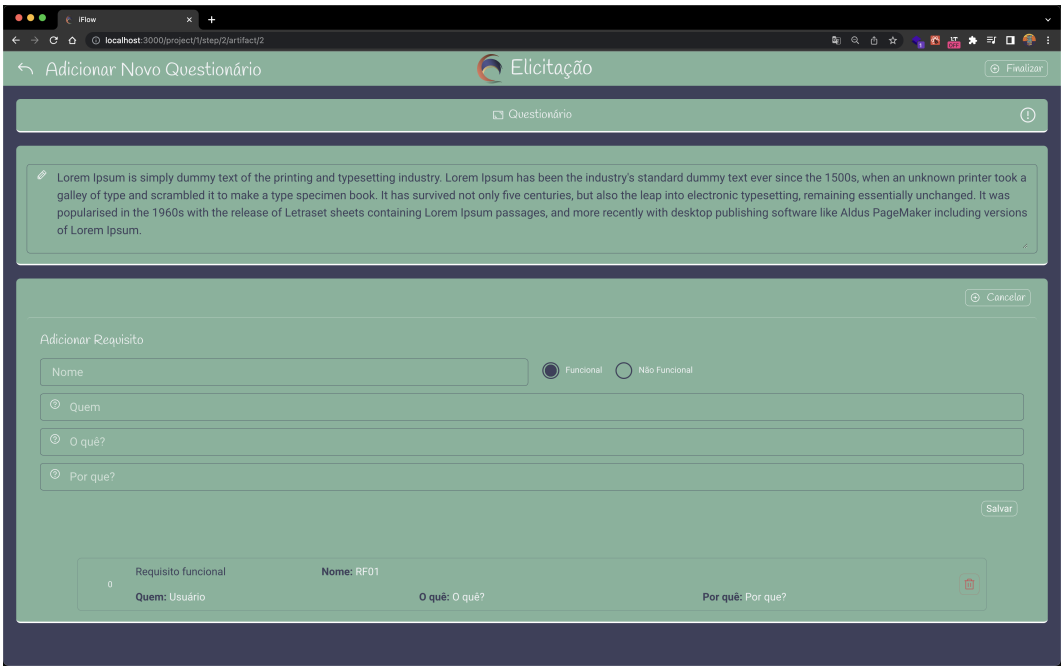


Fonte: Autores, 2022.

esperado, dado que o escopo do projeto era abrangente, e precisava ser ajustado para os prazos de um Trabalho de Conclusão de Curso. Entretanto, optou-se por especificar, desde o começo, algo mais refinado e completo, visando facilitar evoluções futuras da ferramenta *iFlow*, e procurando incorporar muitos dos aspectos, estudados na literatura especializada, e que conferem uma visão bastante adequada às atividades da Engenharia de Requisitos. Seguem as *features* e *user stories* não concluídas:

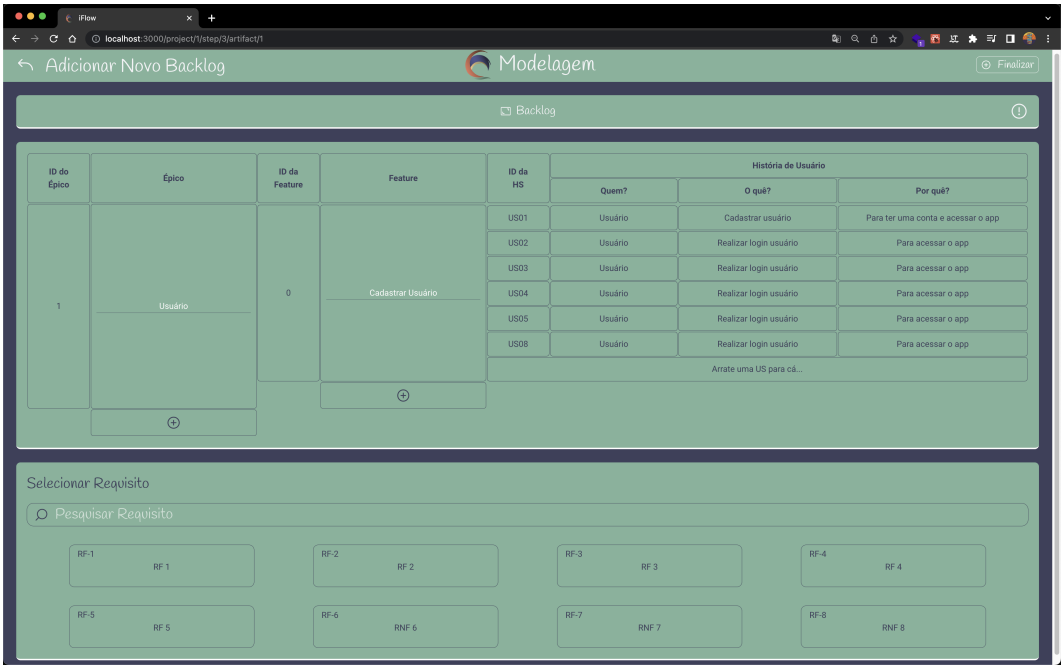
1. **US012**: funcionalidade de *hiperlinkagem* dos léxicos em textos e definições presentes na ferramenta;
2. **FE004**: *feature* relacionada à etapa de verificação;
3. **FE008**: *feature* relacionado ao escopo de edição das informações da conta de usuário, e
4. **US030**: funcionalidade de geração de relatórios com base nos artefatos de cada etapa.

Figura 28 – Tela de Criação de um Novo Artefato



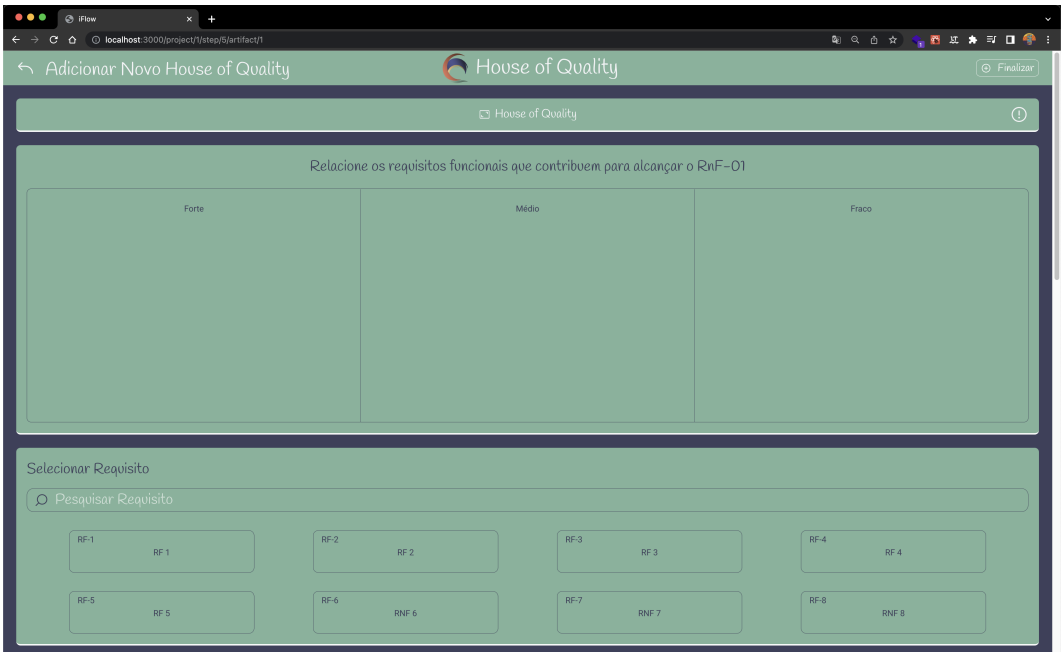
Fonte: Autores, 2022.

Figura 29 – Tela de Criação do *Backlog* do Produto em sua Versão Final



Fonte: Autores, 2022.

Figura 30 – Tela de criação da *House of Quality* em sua Versão Final



Fonte: Autores, 2022.

Uma visualização mais precisa e descritiva dessas funcionalidades pode ser conferida na Tabela 5.

5.8.4 Base de código da Ferramenta *iFlow*

A base de código da ferramenta *iFlow* pode ser encontrada na plataforma GitHub, na organização do projeto, por meio do link: Moreira e Caixeta (2022a).

Há muitos aspectos interessantes, em termos de implementação da Ferramenta *iFlow*. Entretanto, cabe um olhar mais aprofundado em dois pontos:

1. Implementação de telas complexas, tal como ocorre para o caso da tela do *Backlog* do Produto, pelo fato de existir uma funcionalidade de *Drag and Drop* para os requisitos em tela, conferindo maior usabilidade para o usuário na hora de construir este artefato.

Código 1 – *Drag and Drop* na tela do *Backlog* do Produto

```
1 const onDropUserStory = (featureIndex: number, epicIndex:
    number) => {
```

```
    const newEpics = [...epics]
    const userStoryCard = props.userStoryCard
    const featureUserStories =
      newEpics[epicIndex].features[featureIndex].
          userStories

    newEpics[epicIndex].features[featureIndex].userStories.
       pop()
    newEpics[epicIndex].features[featureIndex].userStories
       = [
      ...featureUserStories,
      {
        id: userStoryCard.id,
        name: userStoryCard.name,
        functional: userStoryCard.functional,
        who: userStoryCard.who,
        what: userStoryCard.what,
        why: userStoryCard.why,
      },
    ]
    newEpics[epicIndex].features[featureIndex].userStories.
       push({
      id: 0,
      name: '',
      functional: '',
      who: '',
      what: '',
      why: '',
    })

    const element = document.getElementById('user-story-' +
        userStoryCard.id)
    element?.classList.add('dropped-user-story')
    element?.setAttribute('draggable', 'false')

```

```
32      setEpics(newEpics)
33    }
```

Um código similar também foi utilizado na tela da *House of Quality*, já que ambas as telas possuem a funcionalidade de *Drag and Drop*. O código completo encontra-se no seguinte link: Moreira e Caixeta (2022b). Esse *script* tem o objetivo de deixar os *cards* referentes aos requisitos funcionais como arrastáveis, para facilitar a construção do *Backlog* do Produto.

2. Geração do Produto Mínimo Viável, em sua forma preliminar, por meio da *House of Quality* (Seção 2.7) com uso de *grafos* (Seção 2.7.1) e do algoritmo de menor caminho. A modelagem do grafo deu-se da seguinte forma:

   a) Cada um dos requisitos, amadurecidos na etapa de *modelagem* (Seção 5.2.3), funcionais, categorizados como *user story*, e não funcionais, do último nível do NFR, representa um vértice do grafo;

   b) As arestas são bidirecionais, definidas na etapa de desenvolvimento da *House of Quality*, de forma que o usuário indica o tipo de relação: Forte, considerado peso 2; Média, considerando peso 5; e Fraca, considerando peso 9;

   c) Todas as relações indicadas pelo usuário são salvas no banco de dados por meio do relacionamento gerado na tabela de **House of Quality**, como pode ser visto na Figura 18, e

   d) O algoritmo de menor caminho de **Dijkstra**, sendo uma biblioteca disponibilizada pelo *NPM*, é rodado de forma que o começo e o fim do algoritmo são definidos a partir da combinação sem repetição entre os 3 requisitos não funcionais obtidos no NFR.

O código responsável pelo cálculo do menor caminho, implementado no *backend*, pode ser visto na seção a seguir. Vale ressaltar que o mesmo foi encapsulado com um *Provider*, ou seja, ele possui uma interface comum que pode ser usada para implementar outro tipo de algoritmo de menor caminho, como, por exemplo, Floyd-Warshall. Este pode ser facilmente plugado na rota responsável por retornar esse resultado para o *frontend*.

Código 2 – Algoritmo do Menor Caminho (Dijkstra)

```
1 import Graph from "node-dijkstra";
2
```

```
import { HouseOfQuality } from "../../../../../modules/requirements/entities/HouseOfQuality";
import { ISingleShortestPathProvider } from "../ISingleShortestPathProvider";

interface IPathResult {
  path: string[];
  cost: number;
}

class DijkstraProvider implements ISingleShortestPathProvider {
  run(data: HouseOfQuality[]): number[] {
    const graph = new Graph();

    const nfunctionalIds = data.map((relation) => {
      const { nfunctional_id, functional_id, weight } = relation;
      graph.addNode(nfunctional_id.toString(), {
        [functional_id.toString()]: weight,
      });
      graph.addNode(functional_id.toString(), {
        [nfunctional_id.toString()]: weight,
      });

      return relation.nfunctional_id;
    });

    const uniqueNfunctionalIds = [...new Set(nfunctionalIds)];

    const paths = uniqueNfunctionalIds.flatMap((S, i) =>
      uniqueNfunctionalIds.slice(i + 1).map(
        (V) =>
          graph.path(S.toString(), V.toString(), {
```

```
33              cost: true,
34            }) as IPathResult
35        )
36      );
37
38      const pathsCosts = paths.map((path) => path.cost);
39      const greatestPathCostIndex = pathsCosts.indexOf(Math.
          max(...pathsCosts));
40
41      const pathCasted = paths[greatestPathCostIndex].path.
          map((functional_id) =>
42        parseInt(functional_id, 10)
43      );
44
45      return pathCasted;
46    }
47  }
48
49  export { DijkstraProvider };
```

5.9 Resumo do Capítulo

Neste capítulo, foi apresentado o *iFlow*, que consiste em uma ferramenta que possibilita a semiautomatização, conferindo apoio nos processos da Engenharia de Requisitos. Para isso, foi apresentada a contextualização do *iFlow*; uma seção abordando a ferramenta *iFlow* e seus principais pontos; a arquitetura da solução, com detalhamento dos componentes presentes na mesma; a identidade visual da ferramenta e os diagramas do *iFlow*, sendo o Diagrama de Banco de Dados e o Diagrama de Pacotes.

Além disso, foram mostradas as telas mais relevantes do Protótipo de Alta Fidelidade, Backlog do Produto, e as telas desenvolvidas em sua versão final. Por fim, há uma breve ponderação sobre o escopo da ferramenta.

6 Análise de Resultados

Este capítulo tem por objetivo apresentar os resultados deste trabalho, bem como descrever as atividades de análise aplicadas sobre a ferramenta *iFlow*, Capítulo 5, e acordar sobre as ações de melhoria identificadas. A Análise dos Resultados foi conduzida com base na Metodologia de Análise de Resultados (Seção 4.5), já apresentada anteriormente. Essa estabelece um protocolo de Pesquisa-Ação, o qual compreende as fases de Coleta de Dados (Seção 6.2), Análise e Interpretação dos Dados (Seção 6.3), Elaboração do Plano de Ação (Seção 6.4), e Divulgação dos Resultados (Seção 6.5), conforme apresentado na sequência. Por fim, é apresentado o resumo do capítulo (Seção 6.6).

6.1 Fases da Pesquisa-Ação

Conforme descrito na Metodologia de Análise de Resultados (Seção 4.5), este trabalho segue as fases da Pesquisa-Ação. Em um primeiro momento, consta a fase de Coleta de Dados, sendo essa breve. Logo em seguida, tem-se a fase de Análise e Interpretação dos Dados, onde os dados são coletados e analisados de forma, predominantemente, qualitativa. Seguindo para a Elaboração do Plano de Ação, é realizado um planejamento para solucionar/mitigar erros e imprecisões apontadas pela fase anterior. Por fim, tem-se a fase de Divulgação de Resultados, concluindo o protocolo de Pesquisa-Ação.

6.2 Coleta de Dados

Nesta fase, inicia-se o processo da validação da ferramenta *iFlow* que foi desenvolvida ao longo deste Trabalho de Conclusão de Curso, com a identificação do problema e a descrição do contexto em que esse se insere. Problema: ***Como podemos desenvolver uma ferramenta que consiga apoiar o Engenheiro de Requisitos na sintetização de um Produto Mínimo Viável de software em sua versão preliminar?*** Contexto: um dos principais aspectos levantados nos objetivos específicos foi a preocupação da qualidade da interação do usuário com a ferramenta (Seção 1.4.2), evitando retrabalhos e desperdício de tempo.

Para isso, a coleta de dados foi feita visando coletar, do nosso público alvo, dados com relação à interação e a satisfação do uso da ferramenta *iFlow*. Para tanto, a aplicação

dos testes deu-se sem treinamento prévio, justamente para se ter dados mais precisos na intuitividade da aplicação proposta.

6.3 Análise e Interpretação dos Dados

Nesta fase, foi realizada a coleta de *feedbacks*, visando validar a ferramenta *iFlow* no processo de Engenharia de Requisitos. Os participantes são atuantes na área de *software* e requisitos. Esse público possui experiência prévia dos conceitos inerentes ao escopo deste trabalho, podendo validar o uso da ferramenta *iFlow* com mais tranquilidade.

O formulário foi criado pela plataforma do *Google Forms*, ficou disponível para respostas durante 5 dias corridos, e obteve um total de 40 respostas. Vale ressaltar que todos os usuários que enviaram a pesquisa, concordaram com a divulgação dos dados no contexto deste Trabalho de Conclusão de Curso de forma anônima, com termo de consentimento de conhecimento de todos.

6.3.1 Questionário

Para validar o *iFlow*, foi criado um questionário com algumas perguntas, obrigatórias ou não, além de um teste de usabilidade, de modo a entender a experiência dos usuários. Essa seção apresenta as perguntas inseridas no questionário e, em seguida, as respostas obtidas, por meio de Figuras com os gráficos das respostas. As respostas de escolha múltipla usaram a escala de 1(um) a 5(cinco), validando dentro de uma escala qualitativa indo de péssimo(1) a ótimo(5).

O Questionário foi dividido em três seções, sendo elas: *House of Quality*, NFR Framework (*Non-Functional Requirements*) e *Backlog* do Produto, em que o objetivo foi de validar a usabilidade dos 3 módulos considerados primordiais para o funcionamento da ferramenta *iFlow*.

6.3.1.1 *House of Quality*

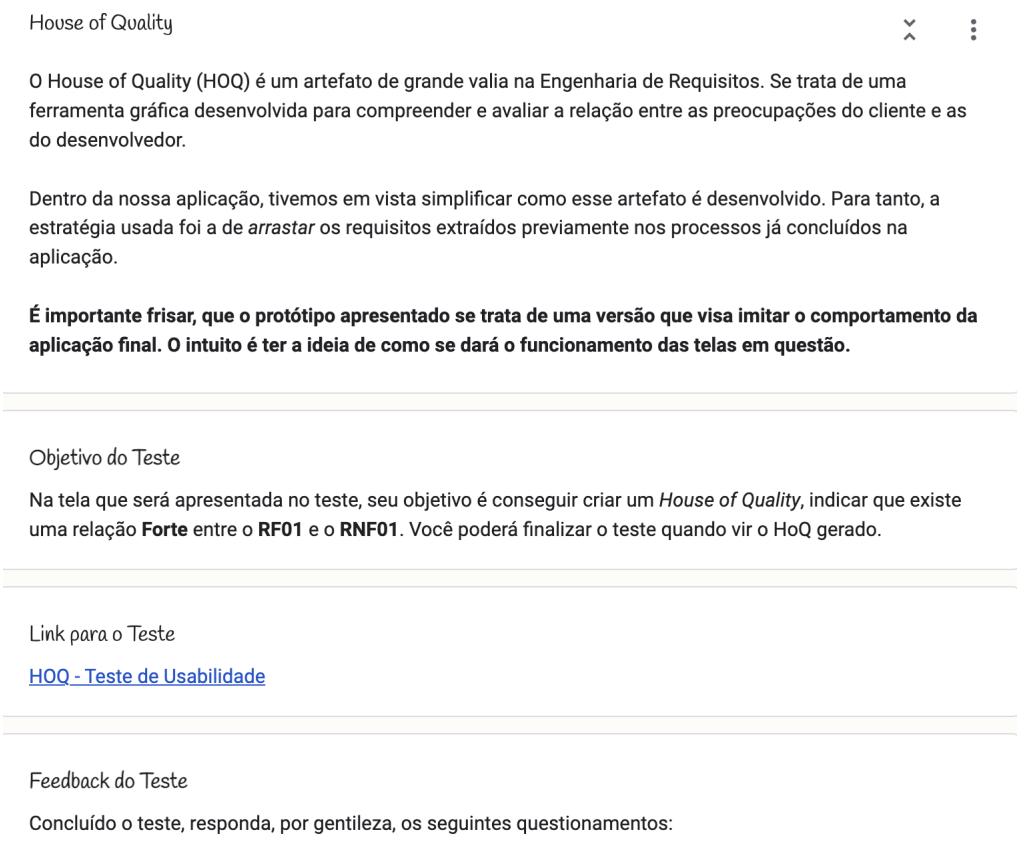
A seção do *House of Quality*, no questionário, dividiu-se da seguinte forma:

1. Uma breve seção explicando como a *House of Quality* se aplica no *iFlow*,
2. Uma pequena descrição e um *hyperlink* do teste de usabilidade que foi realizado pelo usuário, e

3. As perguntas referentes ao teste de usabilidade.

A Figura 31 ilustra a primeira seção do questionário, onde é retratada a *House of Quality*.

Figura 31 – Seção do *House of Quality* no Questionário desenvolvido

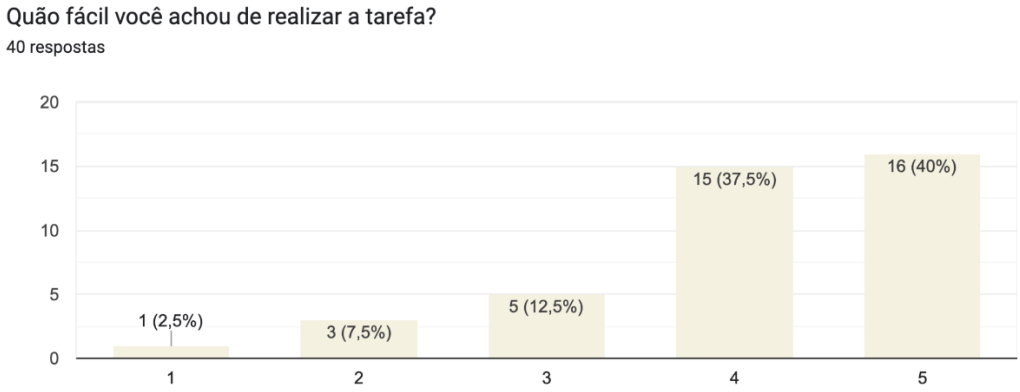


Fonte: Autores, 2022.

Dessa forma, após a realização do teste por parte do usuário, foram indagadas as seguintes perguntas:

- Quão fácil você achou de realizar a tarefa?
- Quão importante foi o processo de arrastar os requisitos funcionais para dentro dos não funcionais?
- O que você acredita que poderia ser melhorado nesse processo?

Figura 32 – Resultados da Pergunta 1: Quão fácil você achou de realizar a tarefa?



Fonte: Autores, 2022.

A Figura 32 representa as respostas acerca da facilidade em se realizar a tarefa na etapa da *House of Quality*. Conclui-se que grande parte dos usuários conseguiram realizar a tarefa de modo fácil, tendo **77,5%** das respostas finais nas notas 4 e 5.

A Figura 33 representa as respostas acerca do processo de arrastar os requisitos funcionais para dentro dos não funcionais, considerando a House of Quality, trazendo uma maior visão sobre os aspectos de usabilidade da ferramenta *iFlow*. Conclui-se que **85%** dos usuários afirmaram que este processo simplifica a criação da *House of Quality*, acarretando um *feedback* positivo.

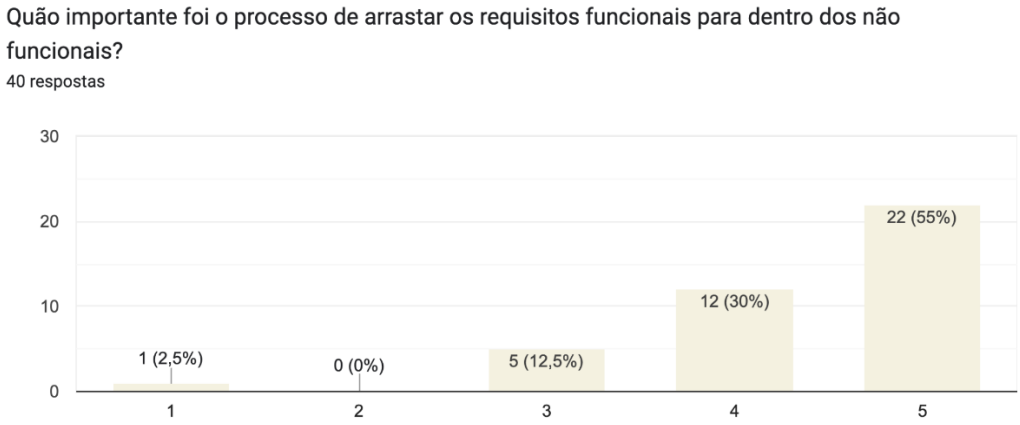
Por fim, foi perguntado aos usuários o que poderia ser melhorado neste processo da *House of Quality* e, fazendo um apanhado geral das respostas, tem-se que:

1. Melhorar o contraste das cores e dar um destaque maior para os requisitos funcionais, além de diferenciar os componentes em tela;
2. Deixar mais explícito que os requisitos podem ser arrastados e indicar quais elementos são clicáveis, e
3. Adicionar um *tooltip* para exibir os detalhes sobre cada requisito funcional.

6.3.1.2 *NFR*

A seção do *NFR*, no questionário, dividiu-se da seguinte forma:

Figura 33 – Resultados da Pergunta 2: Quão importante foi o processo de arrastar os requisitos funcionais para dentro dos não funcionais?



Fonte: Autores, 2022.

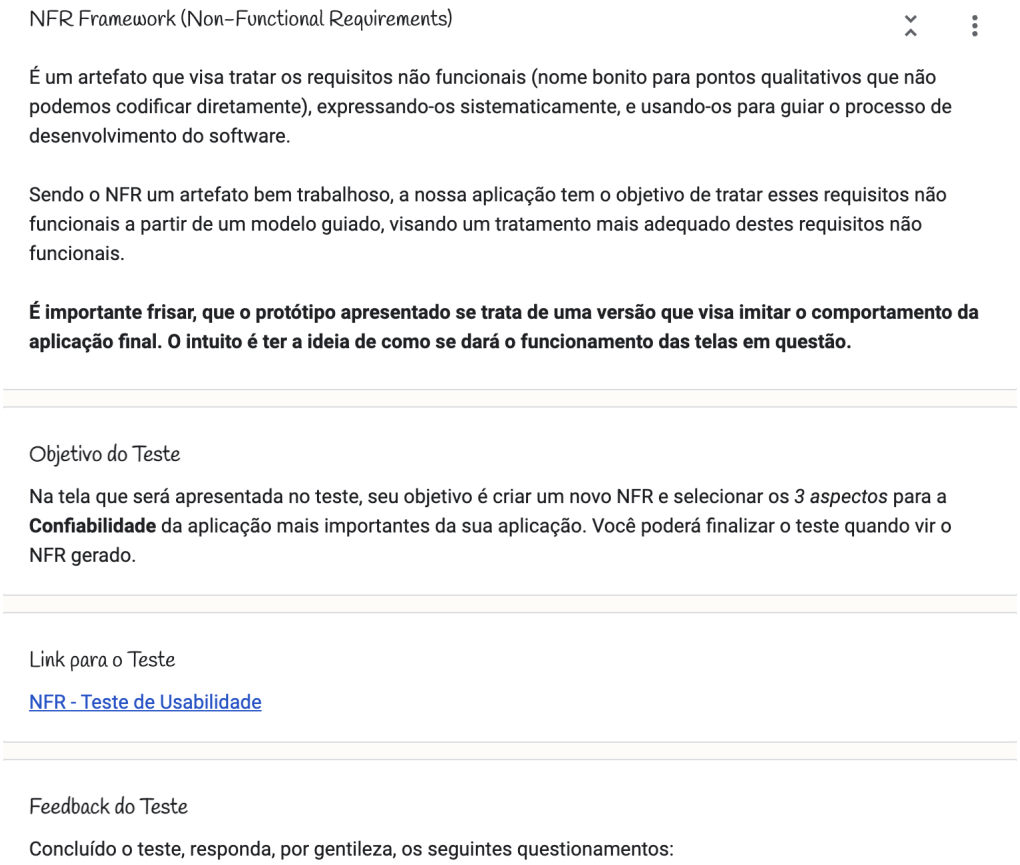
1. Uma breve seção explicando como o NFR se aplica no *iFlow*;
2. Uma pequena descrição e um *hyperlink* do teste de usabilidade que foi realizado pelo usuário, e
3. As perguntas referentes ao teste de usabilidade.

A Figura 34 retrata a segunda seção do questionário, onde é retratado o NFR.

Dessa forma, após a realização do teste por parte de cada usuário, foram indagadas as seguintes perguntas:

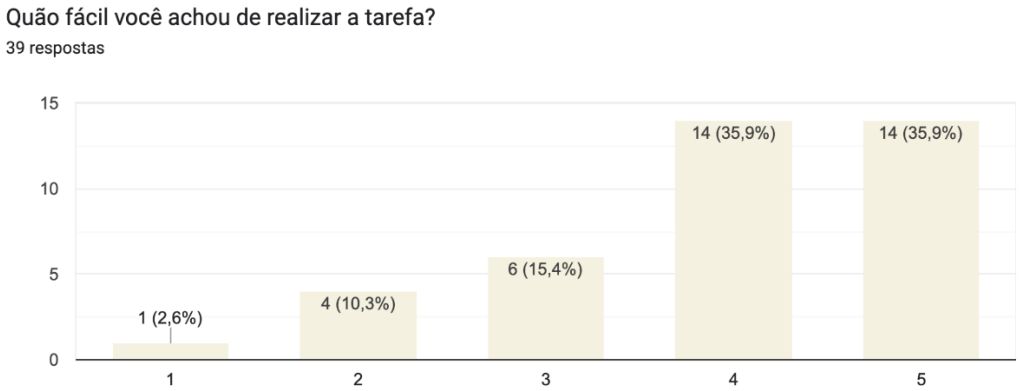
- Quão fácil você achou de realizar a tarefa?
- É mais fácil determinar o NFR a partir de um modelo guiado?
- Você julga que essas telas são intuitivas?
- O que você acredita que poderia ser melhorado nesse processo?

Figura 34 – Seção do NFR no Questionário desenvolvido



Fonte: Autores, 2022.

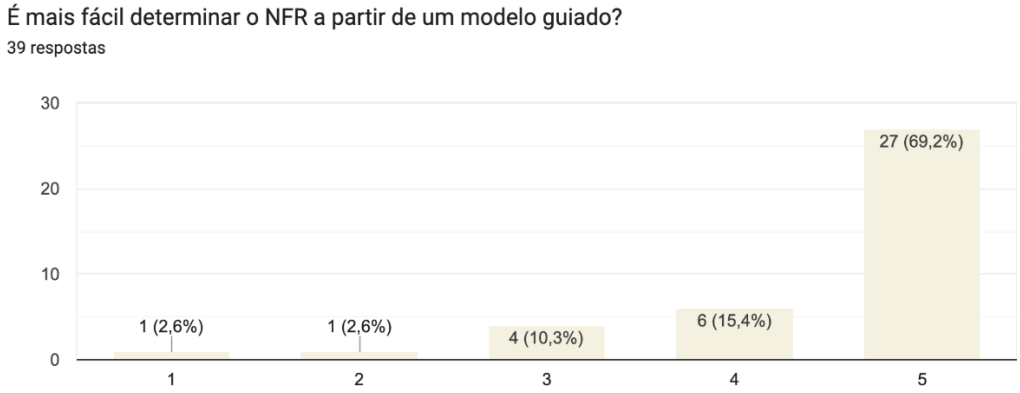
Figura 35 – Gráfico retratando as respostas da pergunta 1: Quão fácil você achou de realizar a tarefa?



Fonte: Autores, 2022.

A Figura 35 representa as respostas acerca da facilidade em se realizar a tarefa na etapa do NFR. Conclui-se que grande parte dos usuários conseguiram realizar a tarefa de modo fácil, tendo **71,8%** das respostas finais nas notas 4 e 5.

Figura 36 – Gráfico retratando as respostas da pergunta 2: É mais fácil determinar o NFR a partir de um modelo guiado?



Fonte: Autores, 2022.

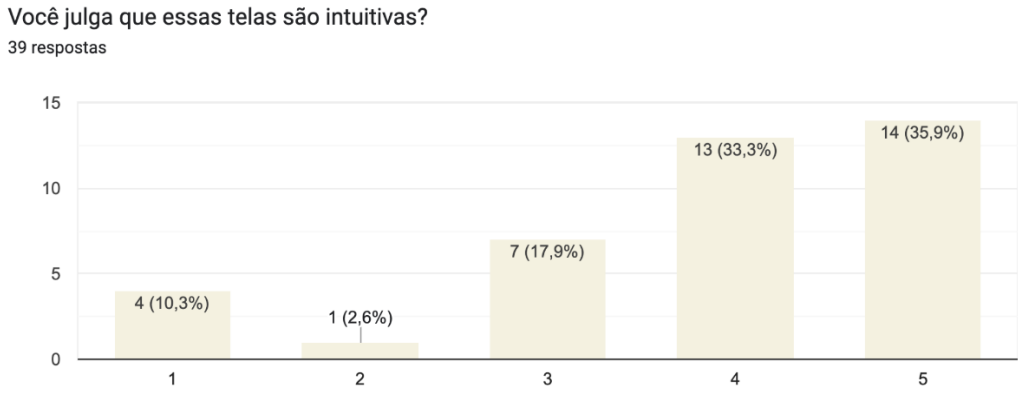
A Figura 36 representa as respostas acerca da facilidade em se realizar o NFR a partir de um modelo guiado, já que é uma representação bem complexa no método tradicional. Conclui-se que **84,6%** dos usuários afirmaram ser de grande utilidade e de grande facilidade realizar o NFR orientando-se por um modelo guiado, trazendo um *feedback* positivo em relação ao método tradicional. Vale ressaltar que essa é uma versão minimizada do NFR, já que o *iFlow* usa apenas três aspectos qualitativos aprofundados em apenas dois níveis.

A Figura 37 demonstra se as telas referentes ao NFR são intuitivas. Conclui-se que **69,2%** dos usuários afirmaram que as telas são de fácil entendimento. Entretanto, **30,8%** dos usuários afirmaram que o nível de entendimento das telas está numa categoria de regular para ruim, o que mostra que essas telas precisam ser melhoradas para uma versão futura.

Por fim, foi perguntado aos usuários o que poderia ser melhorado neste processo do NFR e, fazendo um conglomerado das respostas, tem-se que:

1. Deixar mais claro sobre como relacionar os requisitos não funcionais a cada aspecto;

Figura 37 – Gráfico retratando as respostas da pergunta 3: Você julga que essas telas são intuitivas?



Fonte: Autores, 2022.

2. Acrescentar destaque aos aspectos que estão sendo considerados;
3. Utilizar mais cores para diferenciar o que cada componente faz, e
4. Deixar mais claro quais são os botões clicáveis.

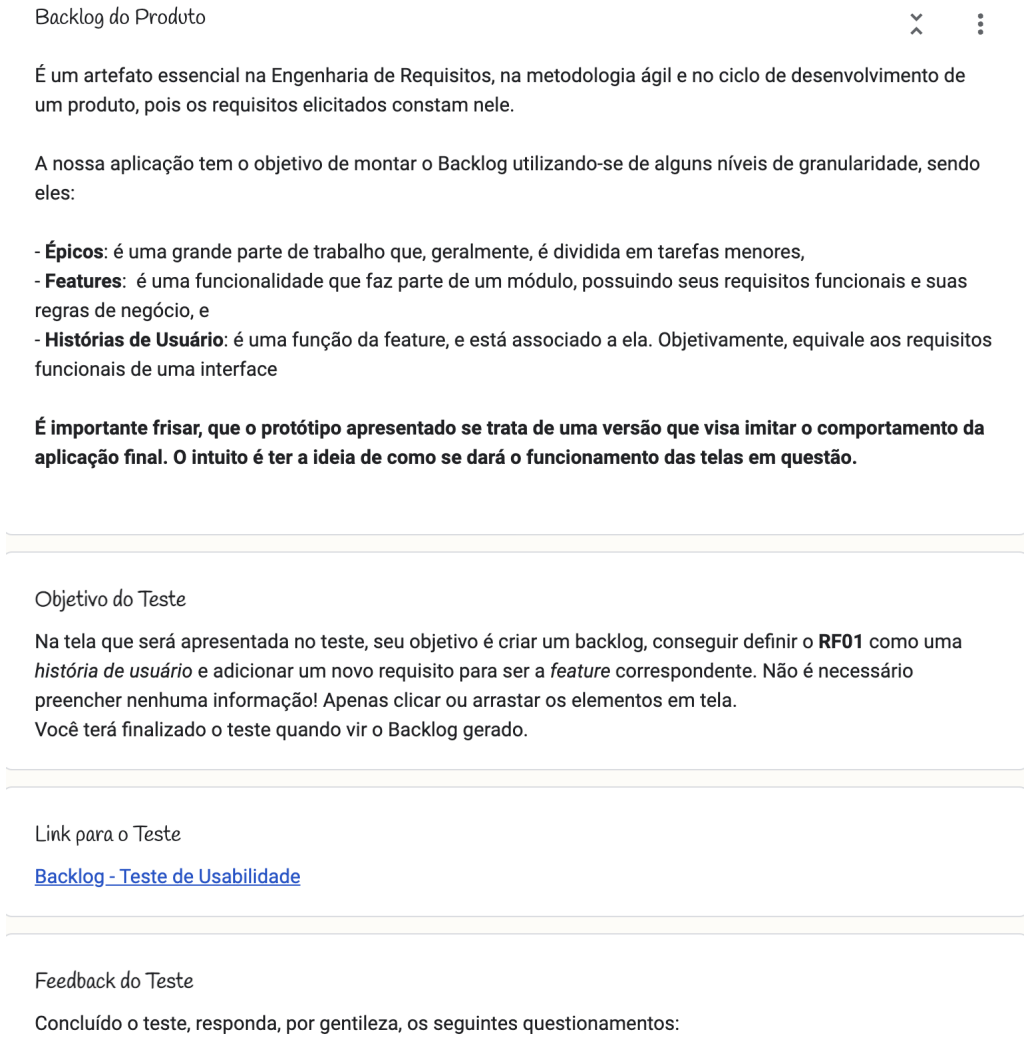
6.3.1.3 *Backlog* do Produto

A seção do *Backlog* do Produto, no questionário, dividiu-se da seguinte forma:

1. Uma breve seção explicando como o *Backlog* do Produto se aplica no *iFlow*;
2. Uma pequena descrição e um *hyperlink* do teste de usabilidade que foi realizado pelo usuário, e
3. As perguntas referentes ao teste de usabilidade.

A Figura 38 ilustra a terceira seção do questionário, onde é retratado o *Backlog* do Produto.

Figura 38 – Seção do Backlog no Questionário desenvolvido

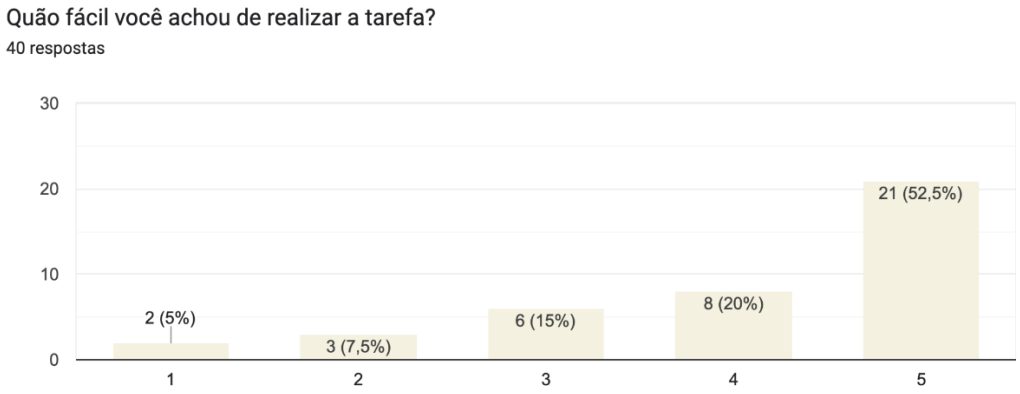


Fonte: Autores, 2022.

Dessa forma, após a realização do teste por parte do usuário, foram indagadas as seguintes perguntas:

- Quão fácil você achou de realizar a tarefa?
- Quão importante você julga que foi arrastar os requisitos em tela?
- O que você acredita que poderia ser melhorado nesse processo?

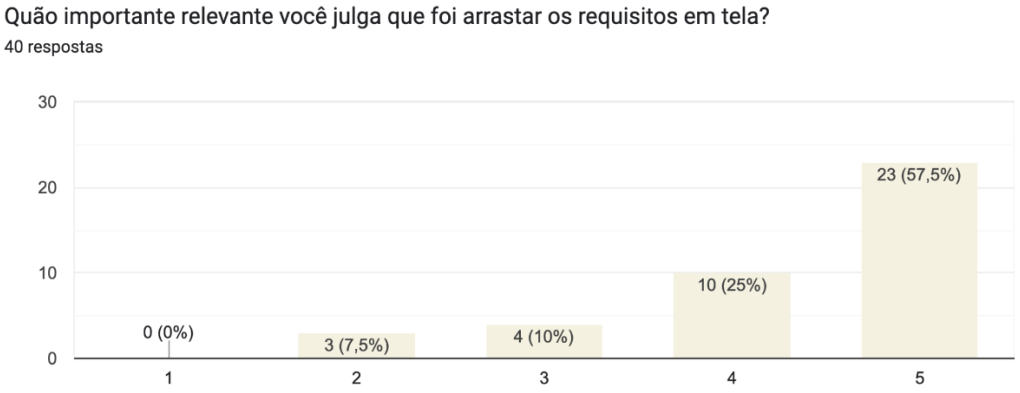
Figura 39 – Gráfico retratando as respostas da pergunta 1: Quão fácil você achou de realizar a tarefa?



Fonte: Autores, 2022.

A Figura 39 representa as respostas acerca da facilidade em se realizar a tarefa na etapa do *Backlog* do Produto. Conclui-se que grande parte dos usuários conseguiram realizar a tarefa de modo fácil, tendo **72,5%** das respostas finais nas notas 4 e 5. Entretanto, **15%** afirmaram que a tarefa teve uma dificuldade regular.

Figura 40 – Gráfico retratando as respostas da pergunta 2: Quão relevante você julga que foi arrastar os requisitos em tela?



Fonte: Autores, 2022.

A Figura 40 demonstra a relevância na funcionalidade de arrastar os requisitos em

tela, acarretando em uma maior visão sobre os aspectos de usabilidade da ferramenta *iFlow*. Conclui-se que **82,5%** dos usuários tiveram um olhar otimista e positivo em relação a essa funcionalidade.

Por fim, foi perguntado aos usuários o que poderia ser melhorado neste processo do *Backlog* do Produto e, fazendo um apanhado geral das respostas, tem-se que:

1. Diferenciar os componentes em tela, pois são muitos e isso pode confundir o usuário;
2. Usar cores para destacar o que já foi realizado, pois isso atrapalha o usuário, e
3. Mais informações a respeito do que se deve fazer, por exemplo, indicando que os requisitos podem ser arrastados.

6.3.2 Teste de Usabilidade

Nesta fase, também, foi realizada a coleta de dados, visando validar a usabilidade da ferramenta *iFlow* no processo de Engenharia de Requisitos. O teste foi realizado na plataforma *Usability Hub*, criando três testes para as telas mais relevantes do *iFlow* (*House of Quality*, *NFR* e *Backlog* do Produto).

Dessa forma, para se ter uma melhor interpretação dos dados, foram utilizados *Heatmaps* para se ter uma melhor visualização, ou seja, pistas visuais óbvias sobre como o fenômeno está agrupado. O Heatmap, nesse contexto, tem o objetivo de mostrar quais as áreas que foram mais clicadas, colocando uma variação de cor, do mais quente (maior quantidade de cliques), para o mais frio (menor quantidade de cliques).

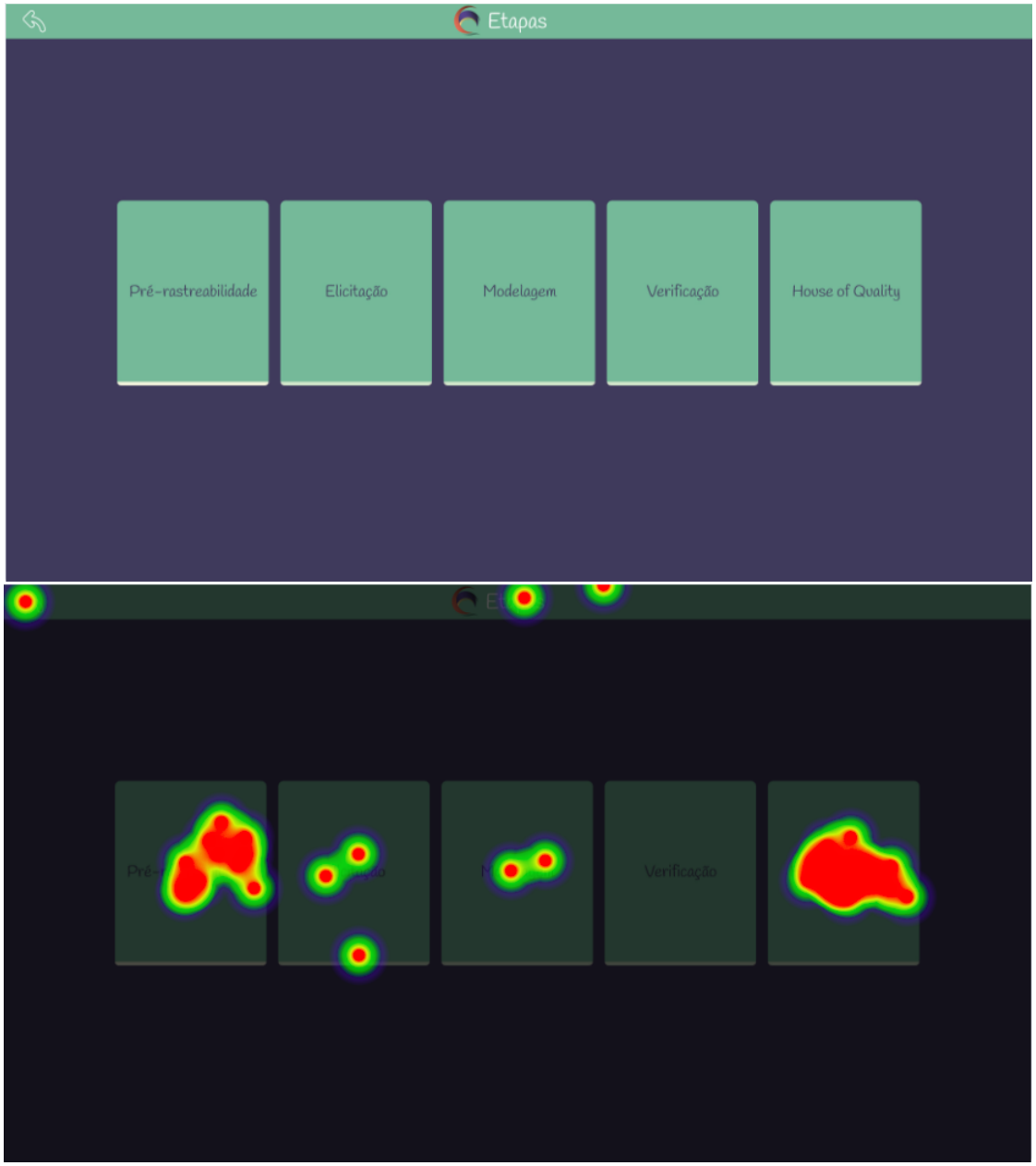
Os participantes são atuantes na área de *software* e requisitos. Esse público possui experiência prévia dos conceitos inerentes ao escopo deste trabalho, podendo validar a usabilidade da ferramenta *iFlow* com mais propriedade.

6.3.2.1 *House of Quality*

Este cenário de usabilidade consistiu em uma simulação do usuário tentando fazer o relacionamento de um requisito funcional com um não funcional na *interface* do *House of Quality* da ferramenta *iFlow*. As Figuras 41 e 42 revelam quais foram os passos do usuário, de forma que se pode observar os pontos mais clicados e menos clicados durante este teste.

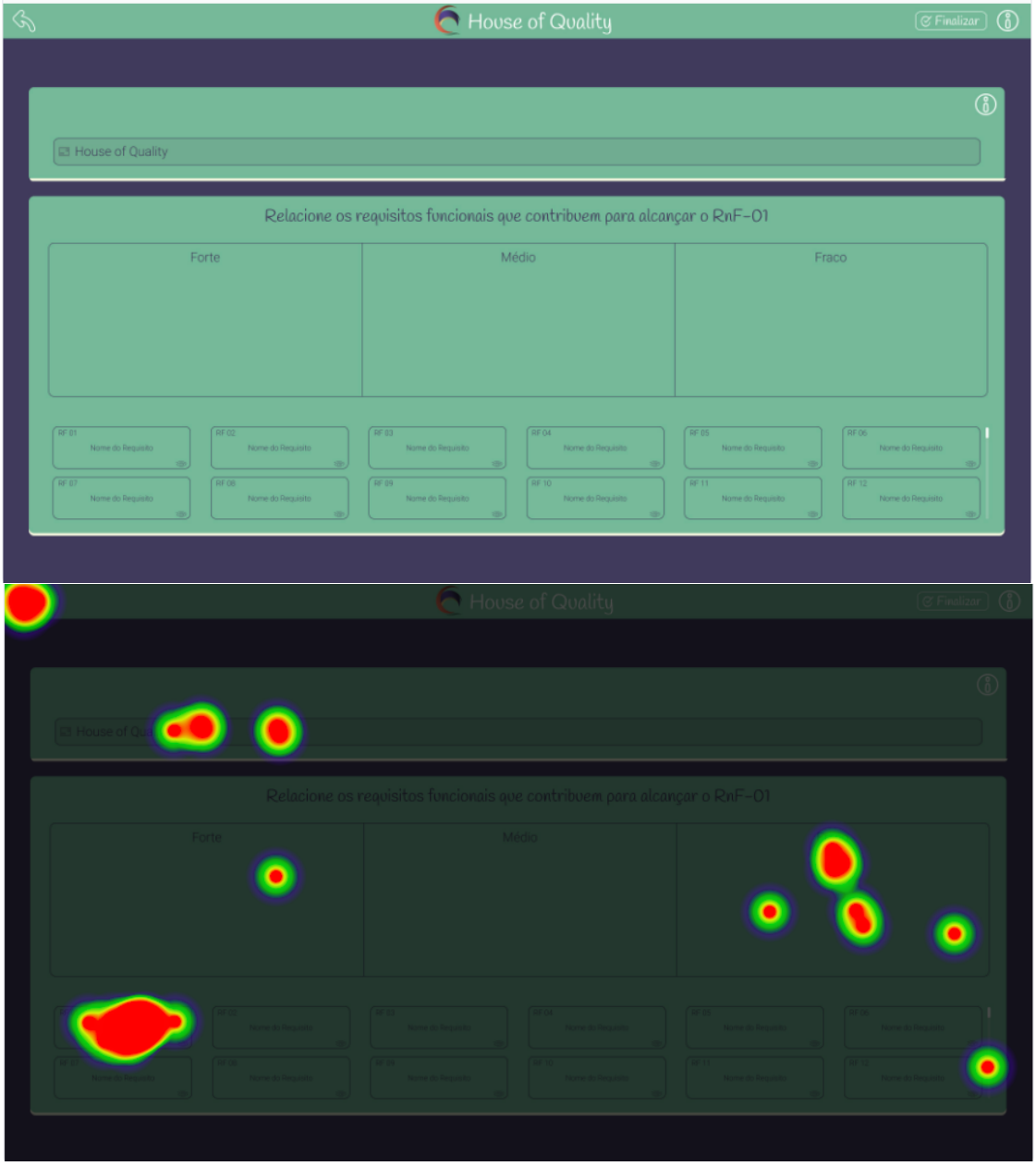
Com base nos dados expostos, podemos tirar as seguintes conclusões:

Figura 41 – *Heatmap* da Tela das Etapas da Engenharia de Requisitos



Fonte: Autores, 2022.

Figura 42 – *Heatmap* da Tela de Criação do *House of Quality*



Fonte: Autores, 2022.

1. O primeiro ponto a ser observado nesse teste se dá pelo fato de os usuários não conhecerem a plataforma e, por isso, foram clicando de forma aleatória até se familiarizar com a ferramenta, e

2. Ainda que de forma um tanto quanto rústica, por não ser possível ainda realizar o movimento de arrastar nas navegações do protótipo, pode-se observar que os usuários conseguiram chegar ao requisito funcional determinado pelo cenário (Figura 31).

6.3.2.2 NFR

O intuito deste cenário de usabilidade foi o de simular o usuário seguindo os passos determinados para gerar o *NFR* na ferramenta *iFlow*. As Figuras 43, 44, 45, 46 e 47 revelam quais foram os passos do usuário, de forma que se pode observar os pontos mais clicados e menos clicados durante este teste.

Com base nos dados expostos, podemos tirar as seguintes conclusões:

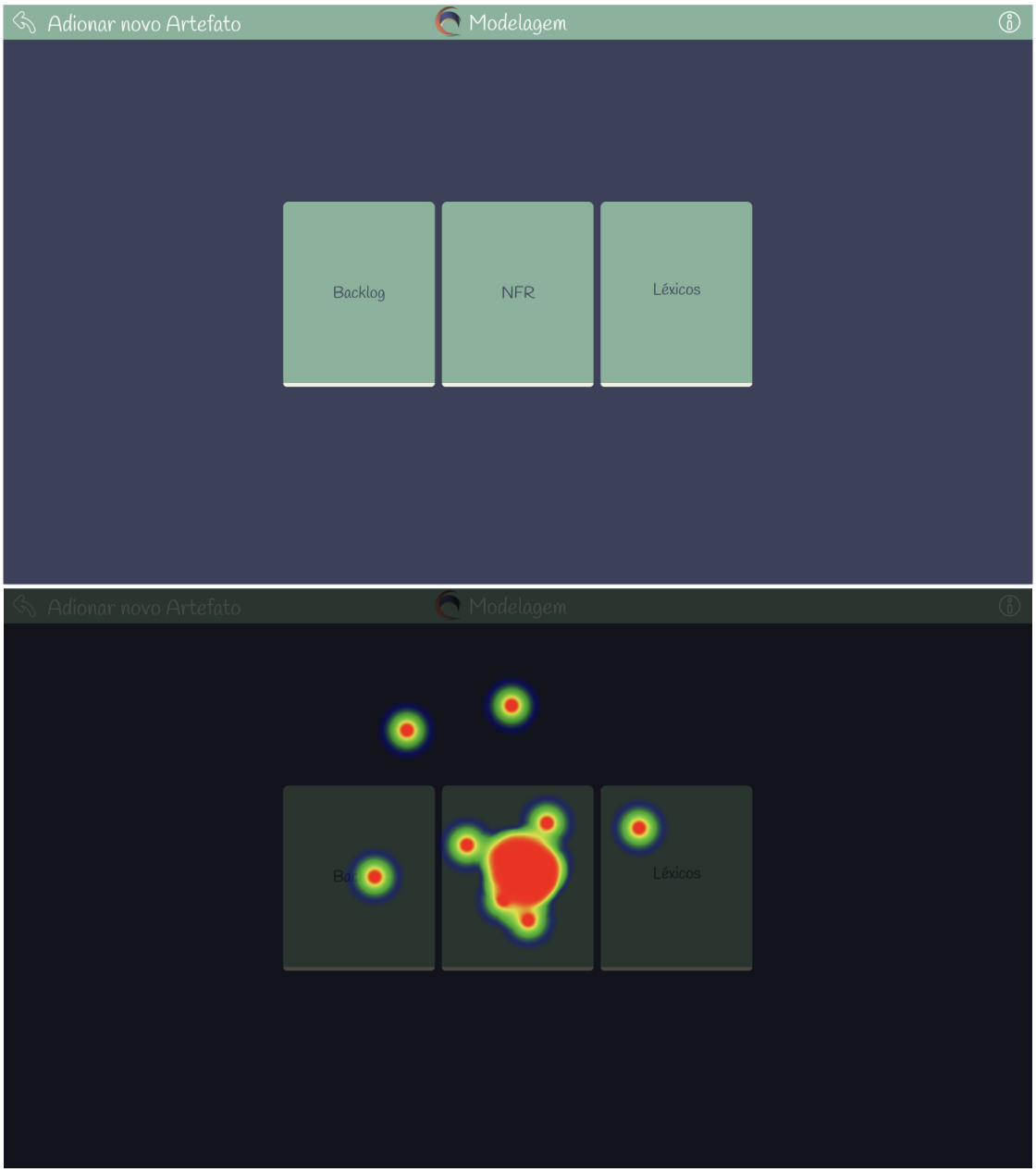
1. No escopo da seleção/criação do primeiro requisito não-funcional, pode-se notar certa dificuldade dos usuários em saber aonde clicar. Isso pode ter sido dado pela diferenciação fraca entre componentes clicáveis e não clicáveis, e

2. Realizada a primeira definição do requisito não-funcional, o usuário conseguiu seguir com a criação do NFR como o esperado.

6.3.2.3 *Backlog* do Produto

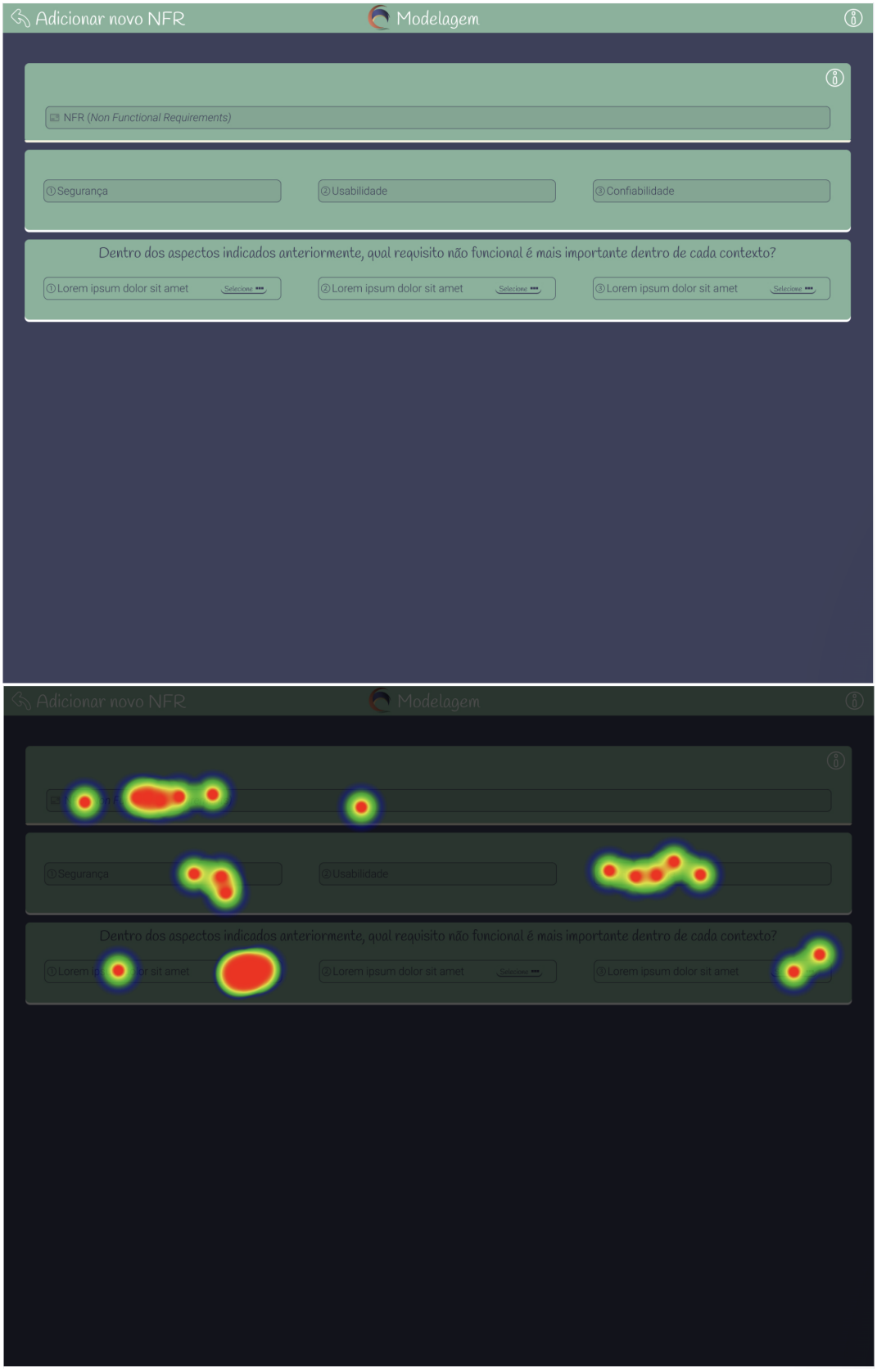
A finalidade deste cenário de usabilidade foi o de reproduzir a criação de um *Backlog* de produto na ferramenta *iFlow*. As Figuras 48, 49, 50, 51 e 52 demonstram quais foram os passos do usuário, de forma que se pode observar os pontos mais clicados e menos clicados durante este teste.

Figura 43 – *Heatmap* da Tela de Criação dos Artefatos da Etapa de Modelagem



Fonte: Autores, 2022.

Figura 44 – *Heatmap* da Tela de Criação do NFR no seu Primeiro Nível



Fonte: Autores, 2022.

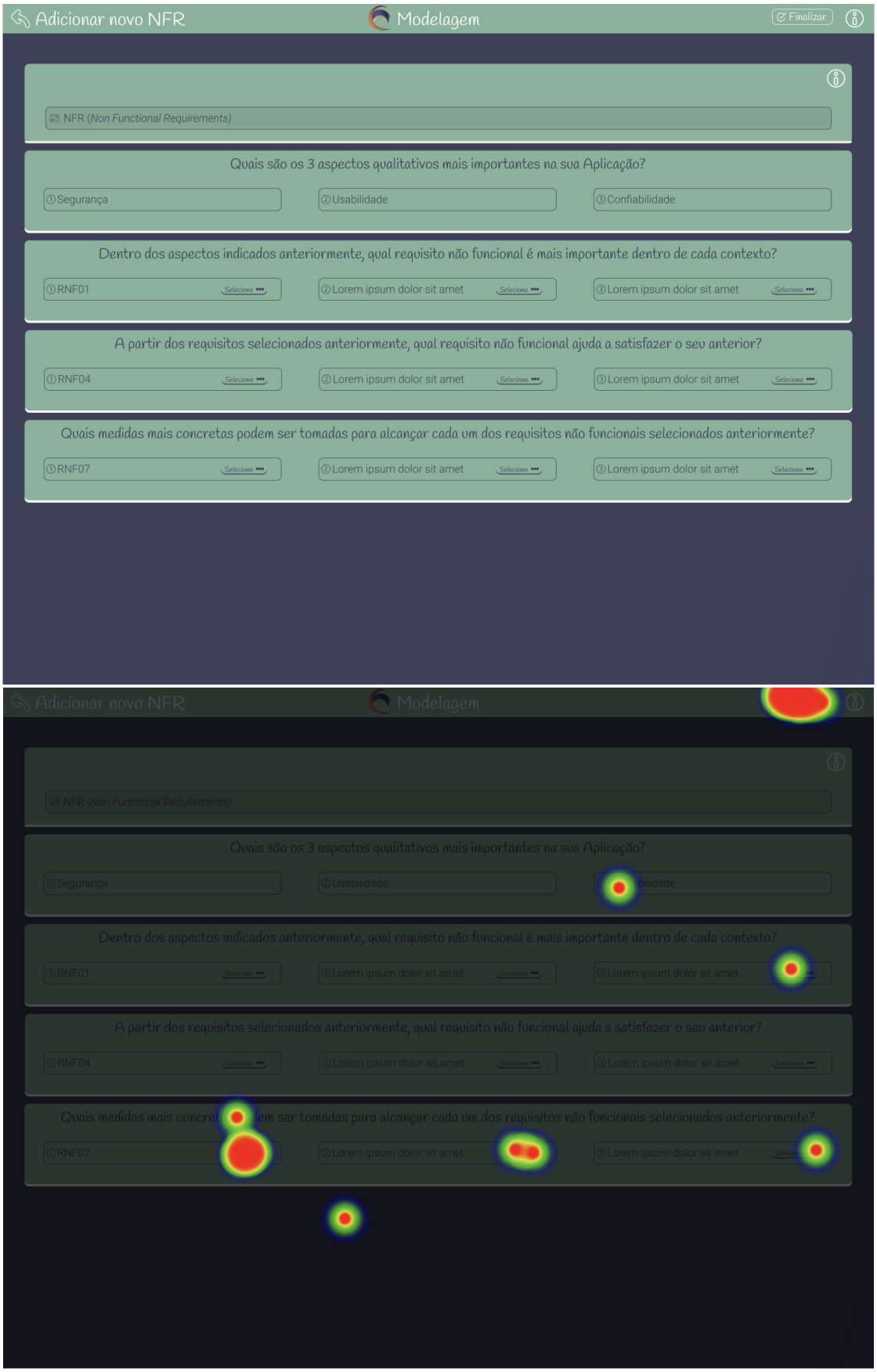
Figura 45 – *Heatmap* da Tela de Criação do NFR no seu Segundo Nível

Fonte: Autores, 2022.

Figura 46 – *Heatmap* da da Tela de Criação do NFR no seu Terceiro Nível

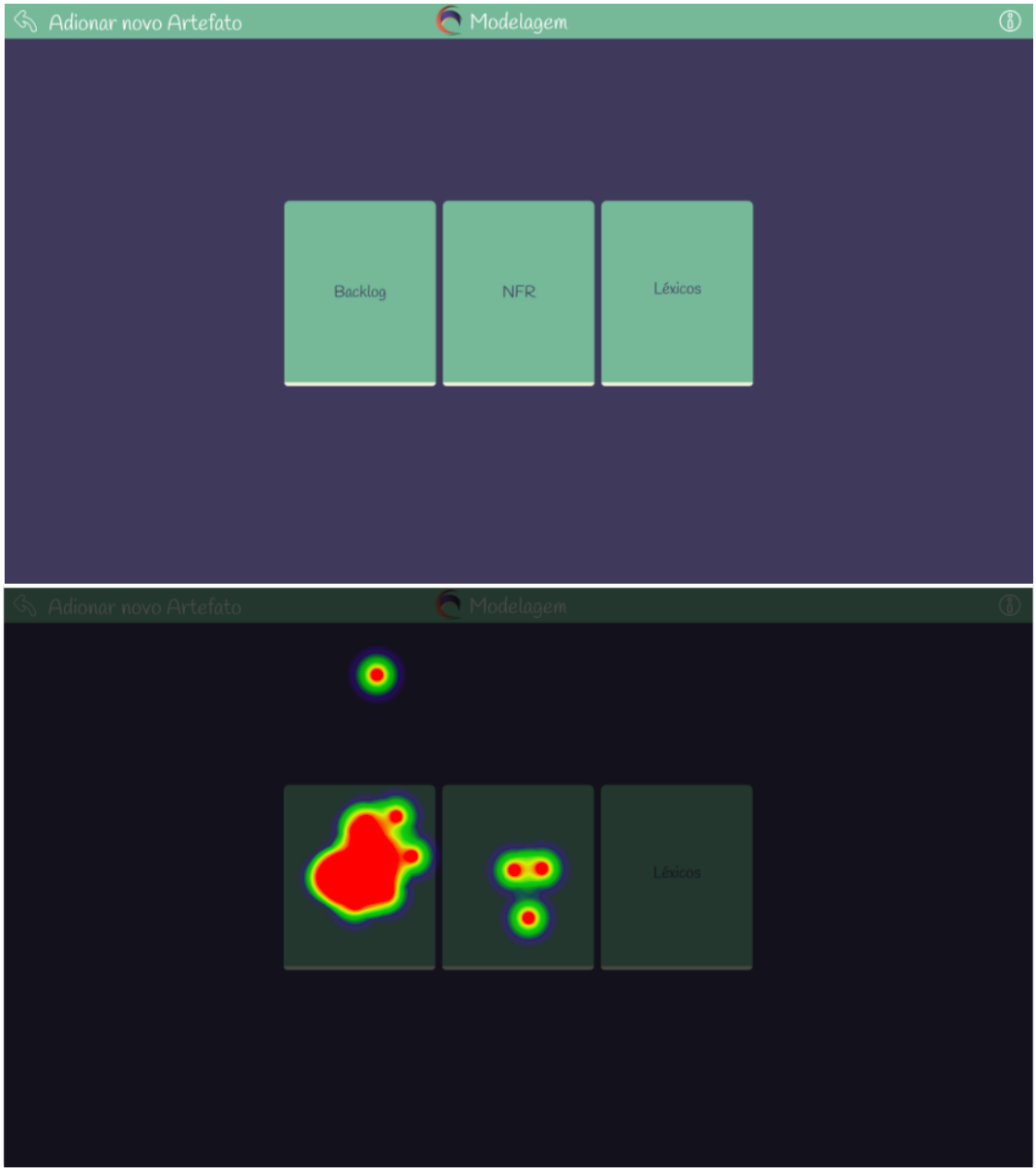
Fonte: Autores, 2022.

Figura 47 – *Heatmap* da Tela de Criação do NFR com Todos os Níveis Preenchidos



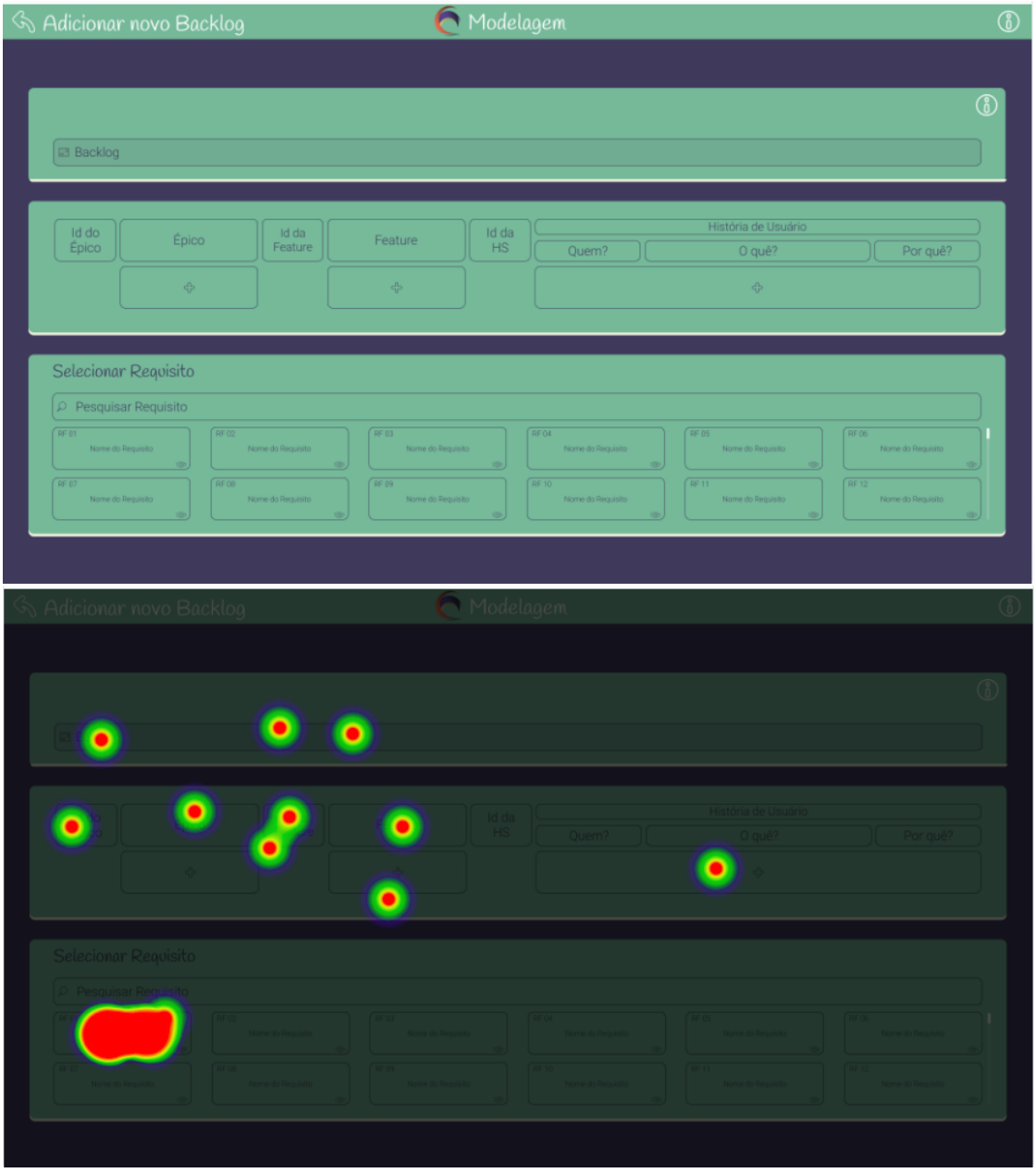
Fonte: Autores, 2022.

Figura 48 – *Heatmap* da Tela de Criação dos Artefatos da Etapa de Modelagem



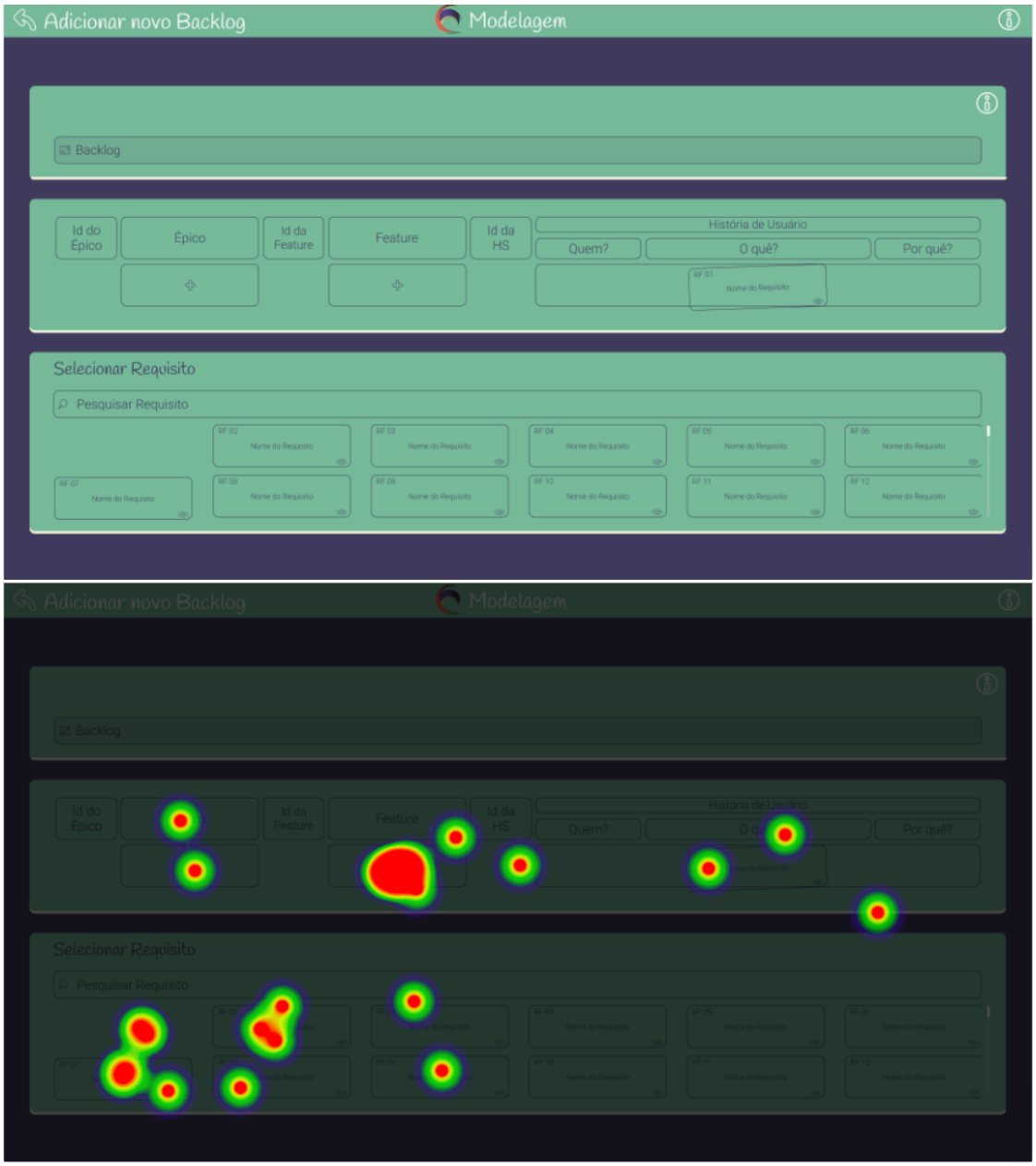
Fonte: Autores, 2022.

Figura 49 – *Heatmap* da Tela de Criação do *Backlog*



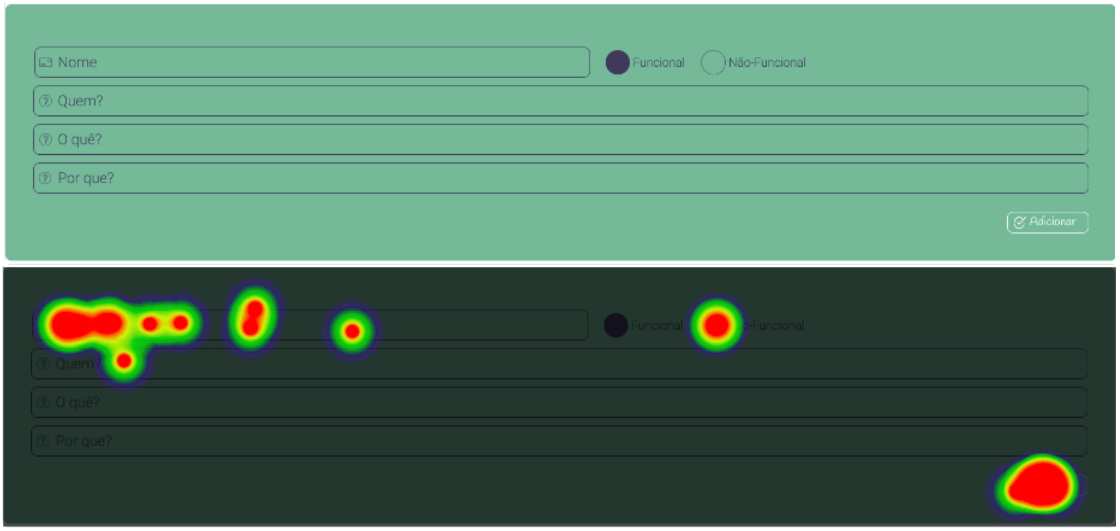
Fonte: Autores, 2022.

Figura 50 – *Heatmap* da Tela de Criação do *Backlog* com o Requisito sendo Arrastado



Fonte: Autores, 2022.

Figura 51 – *Heatmap* da Tela de Criação de um novo Requisito Funcional ou Requisito não Funcional



Fonte: Autores, 2022.

Com base nos dados expostos, podemos tirar as seguintes conclusões:

1. Este processo pode ser considerado um dos mais trabalhosos dentre os passos necessários da Engenharia de Requisitos, e a interface proposta mostrou-se com uma grande taxa de sucesso, uma vez que ainda que tenham alguns cliques fora do componente esperado, os cliques estão mais concentrados no componente esperado;

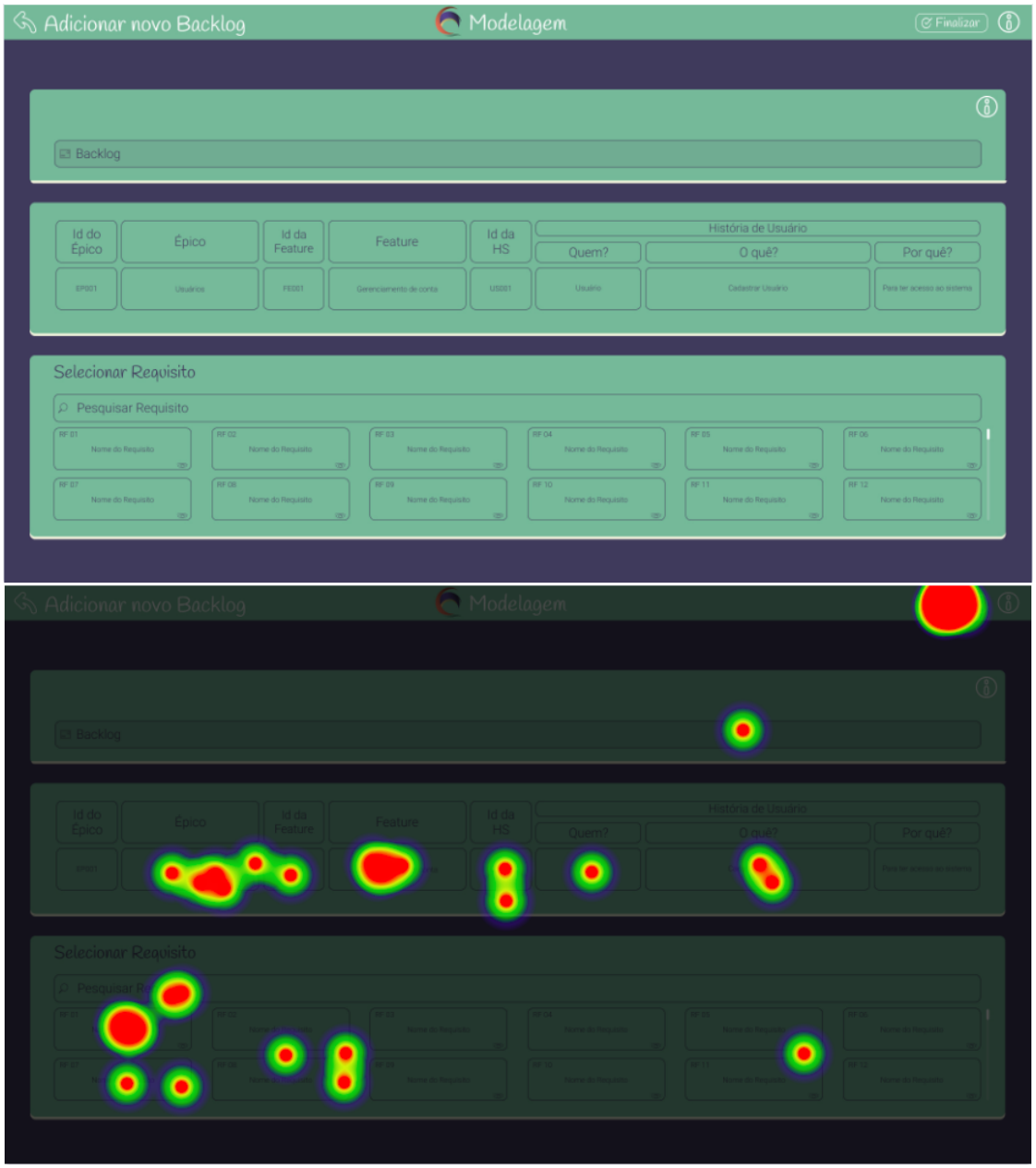
2. Na Figura 51, pode-se observar que a confusão de cliques ocorreu pelos usuários quererem preencher as informações solicitadas no formulário, o que se mostra positivo, pois pode evidenciar uma interface simples e direta, e

3. A confusão de cliques da Figura 52 pode ter se dado pela falta de evidência de que o requisito não estava mais disponível para ser usado.

6.4 Elaboração do Plano de Ação

A partir dos *feedbacks* coletados com o questionário e com o teste de usabilidade, e avaliando as análises descritas nas seções anteriores, foi possível levantar alguns pontos para melhorar o *iFlow*, resultando em um Plano de Ação. Dentre os pontos de melhorias, compreendidos no Plano de Ação, destacam-se:

Figura 52 – *Heatmap* da Tela do *Backlog* preenchido com Épicos, *Features* e História de Usuário



Fonte: Autores, 2022.

1. Melhoria de contraste no uso das cores presentes nas telas e nos componentes;
2. Adição de um componente, na tela de *House of Quality*, para poder visualizar as informações dos requisitos funcionais;
3. Adicionar mais cores para poder destacar e evidenciar cada componente e facilitar o desenvolvimento das tarefas correspondentes;
4. Deixar mais evidente quais componentes são clicáveis;
5. Melhorar, na tela do *NFR*, como se relacionam os aspectos qualitativos propostos com os requisitos que estão sendo destrinchados;
6. Adição de mensagens intuitivas que evidenciem que os componentes em tela são arrastáveis, e guiar para aonde devem ser arrastados, e
7. Adicionar evidências para quando um componente já não estiver mais disponível para ser arrastado, na tela de *Backlog*.

6.5 Divulgação de Resultados

Diante do abrangente escopo de atuação que foi definido para a ferramenta *iFlow*, sendo, de fato, automatizada boa parte do processo da Engenharia de Requisitos com um enfoque em gerar um Produto Mínimo Viável, em sua versão preliminar, a presente Pequisa-Ação teve o intuito de levantar pontos de melhoria da ferramenta e definir um conjunto de próximos passos para solucioná-los. As implementações dessas melhorias compreendem excelentes oportunidades para trabalhos futuros, permitindo refinar ainda mais a ferramenta *iFlow*, e conferindo mais fidedignidade e contribuições às atividades da Engenharia de Requisitos.

6.6 Considerações Finais

Neste capítulo, foram apresentados os resultados obtidos ao longo das fases de Pesquisa-Ação. A fase de Coleta de Dados identifica o problema e o contexto em que este trabalho está inserido, visando coletar dados para as fases posteriores. Em seguida, a fase de Análise e Interpretação de Dados aborda o estudo feito para a Coleta de Dados, provida consultando os usuários participantes da pesquisa. Logo em seguida, tem-se a Elaboração de um Plano de Ação que, a partir dos dados coletados, foram planejadas

ações futuras para atender às sugestões dos usuários participantes do questionário e do teste de usabilidade aplicados. Por fim, tem-se a divulgação dos resultados provindos do Plano de Ação, em que estes ficam como oportunidade para trabalhos futuros.

7 Conclusão

Este capítulo apresenta as considerações finais das atividades relacionadas ao Trabalho de Conclusão de Curso, bem como os resultados alcançados durante a condução, a elaboração e o desenvolvimento. Dessa forma, em um primeiro momento, é apresentado o andamento geral do trabalho (Seção 7.1), desde a Introdução até a Apresentação para a banca (TCC2). Na sequência, constam as respostas para as questões de pesquisa levantadas (Seção 7.2). Há ainda um detalhamento sobre o cumprimento dos objetivos propostos (Seção 7.3). Por fim, são elencadas as principais contribuições e fragilidades da Ferramenta *iFlow*, dando um panorama de melhorias para trabalhos futuros.

7.1 *Status* do Trabalho

Na primeira etapa deste trabalho, focou-se no desenvolvimento das atividades que fundamentavam e sistematizavam a proposta de projeto em relação à criação de uma ferramenta, capaz de conferir apoio e semiautomatizações às etapas relacionadas ao processo de Engenharia de Requisitos. Nesse sentido, a Tabela 3 mostra o andamento das atividades da fase inicial do trabalho, desde a definição do tema até à apresentação aos membros da banca (TCC1), sendo todas tarefas já concluídas.

Tabela 3 – Andamento das atividades do TCC1

Atividade	Andamento
Definir Tema	Concluída
Formular Proposta	Concluída
Contextualizar Problema	Concluída
Realizar Levantamento Bibliográfico	Concluída
Definir Referencial Tecnológico	Concluída
Definir Metodologia	Concluída
Refinar Proposta	Concluída
Desenvolver o Protótipo da Ferramenta	Concluída
Revisar TCC1	Concluída
Definir *Backlog*	Concluída
Apresentar TCC1	Concluída

Fonte: Autores, 2022.

Em um segundo momento, a condução do trabalho deu-se de forma mais prática,

para desenvolver a ferramenta *iFlow* (Capítulo 5); levantar dados para avaliar a usabilidade da ferramenta, e divulgar os resultados, documentando-os nesta monografia. Nesse sentido, a Tabela 4 mostra o *status* de todas as atividades conduzidas nesse contexto, sendo todas já concluídas (exceto a apresentação à banca, que ocorrerá em breve).

Tabela 4 – Andamento das atividades do TCC2

Atividade	Andamento
Corrigir Apontamentos da Banca	Concluída
Desenvolvimento da Ferramenta	Concluída
Coleta de Dados	Concluída
Análise dos Resultados	Concluída
Revisar TCC2	Concluída
Apresentar TCC2	Concluída

Fonte: Autores, 2022.

7.2 *Status* das Questões de Pesquisa

Para a condução deste estudo, foram levantadas algumas indagações que permearam os passos e as decisões tomadas. A partir de todos os expostos e evidências presentes nesse Trabalho de Conclusão de Curso, conclui-se que:

1. **Como podemos desenvolver uma ferramenta que apoie o Engenheiro de Requisitos na sintetização de um produto de *software* abstrato para um que seja concreto?**: diante dos resultados apresentados nos Capítulos *iFlow* (Capítulo 5) e Análise e Interpretação dos Dados (Capítulo 6), e das respostas obtidas pelo Questionário e pelo Teste de Usabilidade, pode-se concluir que o *iFlow* aponta em direção da solução. Uma vez que, ainda que a aplicação tenha sido testada em módulos, concomitantemente foi testada toda a integração do *iFlow*, dado que os módulos são fortemente dependentes um do outro. Entretanto, há necessidade de melhorias, sendo essas oportunidades para trabalhos futuros. Nesse sentido, sabe-se *iFlow* é orientado pelos processos da Engenharia de Requisitos de forma simplificada e intuitiva, além de se apoiar na literatura especializada para que se tenha um MVP, na sua versão preliminar, como resultado. Além disso, o *iFlow* foi pensado para que os Engenheiros de Requisitos façam todas as etapas do processo para evitar retrabalhos e desperdícios de tempo.

2. **Como podemos facilitar o desenvolvimento criativo do Engenheiro de Requisitos, de tal forma que a rastreabilidade das fontes dos requisitos**

sejam mantidas?: para prover um desenvolvimento criativo e manter a rastreabilidade dos requisitos, o *iFlow* foi desenvolvido de tal forma que o usuário consiga criar seus próprios artefatos e, posteriormente, extrair os requisitos para cada um, mantendo tudo salvo. Dessa forma, o usuário consegue acessar e visualizar os rastros das fontes destes requisitos sem dificuldades, provendo uma aplicação que, ainda que seja forçado o uso de uma metodologia, busca promover funcionalidades que não limitem a criatividade, como a possibilidade de refatoração de artefatos e integração com plataformas externas por meio do *upload* dos artefatos já desenvolvidos.

3. **Tendo posse dos requisitos amadurecidos e sua importância para o produto de *software*, como podemos correlacionar esses dados para gerar um possível *Minimum Viable Product* (MVP)?**: a ferramenta *iFlow* foi desenvolvida com o objetivo de gerar um possível MVP, na sua versão preliminar, fazendo a correlação dos dados provenientes de todo o processo de Engenharia de Requisitos proposto, conforme descrito no capítulo de Embasamento Teórico (Capítulo 2). Dessa forma, o *iFlow* apoia o usuário na realização das etapas, passo a passo, para que, no final, com os requisitos detalhados e etapas finalizadas, ele consiga gerar um possível MVP para a sua aplicação.

7.3 Objetivos Concluídos

Retomando os objetivos específicos, apresentados no capítulo de Introdução (Capítulo 1), descritos como:

- **Automatizar a *hiperlinkagem* entre os artefatos de requisitos**:
  Status: Visão preliminar proposta, cabendo maior aprimoramento na renderização desses dados na interface.

- **Viabilizar a geração do MVP (versão preliminar) com base nos artefatos gerados**:
  Status: Visão preliminar proposta, cabendo ainda pequenos ajustes, com base no andamento do projeto, em termos evolutivos e trabalhos futuros.

- **Prover uma *interface* para a ferramenta que guie o usuário no processo de criação de alguns artefatos de requisitos pré-definidos**:

Status: Visão preliminar proposta, cabendo ainda maior aprimoramento em alguns aspectos mencionados na seção de Análise e Interpretação dos Dados (Seção 6.3), para que se tenha uma maior facilidade em se realizar o que foi proposto em cada etapa do processo.

- **Viabilizar a criação da *interface*, mencionada no item anterior, de modo a proporcionar uma experiência de usuário satisfatória, visando facilitar o processo da Engenharia de Requisitos**:
  Status: Visão preliminar proposta, cabendo ainda pequenos ajustes de interface para que se tenha uma maior usabilidade, com base no andamento do projeto, em termos evolutivos e com trabalhos futuros.

- **Realizar uma primeira análise dos resultados obtidos, usando como base uma amostra do público alvo**:
  Status: Visão preliminar proposta, podendo ser visualizada na seção Análise e Interpretação dos Dados (Seção 6.3).

Foi possível alcançar grande parte dos objetivos específicos, conforme descritos na seção Objetivos Específicos (Seção 1.4.2). Com base nesse embasamento teórico, foi possível propor uma ferramenta capaz de gerar um Produto Mínimo Viável, na sua versão preliminar, a partir das etapas mais relevantes dentro do processo de Engenharia de Requisitos.

Realizou-se uma Pesquisa-Ação para validação da usabilidade do *iFlow* e, para isso, foram elaborados um questionário e um teste de usabilidade na ferramenta *UsabilityHub* (MILOSAVLJEVIC; FIRTH-MCCOY; LASLETT, 2008), para avaliar o uso da ferramenta desenvolvida. As validações podem ser vistas no Capítulo Análise dos Resultados (Capítulo 6.3).

7.4 Considerações do *iFlow*

7.4.1 Contribuições

O *iFlow* é uma ferramenta que centraliza todo o processo de desenvolvimento da Engenharia de Requisitos, facilitando a *linkagem* entre artefatos e requisitos e a consolidação dos rastros entre ambos. Confere ao usuário a liberdade de adicionar artefatos produzidos externamente e oferece uma interface capaz de extrair os requisitos concomitantemente.

Além de ter o escopo de consolidar artefatos produzidos, tanto na pré-rastreabilidade (Seção 2.2.1) como elicitação (Seção 2.3), o *iFlow* conseguiu prover telas, de implementação não trivial, com interfaces simplificadas, capazes de facilitar a montagem de artefatos complexos, como o *Backlog* (Seção 2.4.1) e o NFR (Seção 2.4.2). Ainda que sejam necessários ajustes, como foi revelado no Capítulo de Análise dos Resultados (Seção 6), as telas tiveram alta taxa de acertos e *feedbacks* dos usuários.

Ressalta-se, sendo esse um ponto de significativa relevância, que a ferramenta provê uma forma de relacionar requisitos funcionais e não funcionais, conferindo um olhar mais qualitativo a produtos de *software*. A partir desses relacionamentos, o *iFlow* ainda sugere um Produto Mínimo Viável, em sua forma preliminar, que viabiliza uma versão do produto, agregando valor ao usuário.

7.4.2 Fragilidades

Dentre os pontos que necessitam de maior atenção na ferramenta, há necessidade de implementação das funcionalidades que não fizeram parte dessa primeira versão, como já mencionado na seção Ajustes de Escopo (Seção 5.8.3). Com isso, sendo possível uma automatização completa das etapas propostas da Engenharia de Requisitos.

Existe a demanda também de melhorar a correlação entre os requisitos funcionais e não funcionais, para poder ser considerado no momento da geração do Produto Mínimo Viável.

Por fim, vale ressaltar que, ainda que o Protótipo de Alta Fidelidade (Seção 5.6) tenha sido desenvolvido com bastante cuidado, buscando facilitar o trabalho do usuário, alguns pontos de melhoria foram levantados na Análise dos Resultados (Capítulo 6) e precisam de atenção.

7.4.3 Trabalhos Futuros

Diante do exposto, alguns pontos de melhoria são necessários para que a ferramenta atenda de forma mais adequada a Engenharia de Requisitos, sendo eles:

1. Melhorar os aspectos de usabilidade e de interface para que os processos na aplicação fiquem mais intuitivos;
2. Evoluir o NFR para aumentar o número de aspectos qualitativos;

3. Desenvolver as funcionalidades que foram desconsideradas, ao se ajustar o escopo (Seção 5.8.3);

4. Realizar e automatizar a *hyperlinkagem* dos léxicos no contexto da ferramenta, e

5. Aperfeiçoar o algoritmo de menor caminho para que ele possa considerar aspectos qualitativos na geração do MVP;

6. Trocar o algoritmo de **Djikstra** para outras possibilidades que considerem pesos negativos, para mudar a abordagem usada no cálculo do caminho máximo, como, por exemplo, o de **Bellman-Ford**. Buscar outras abordagens de programação que sejam mais flexíveis para computar o *MVP* correlacionando os requisitos funcionais e não-funcionais, como a programação dinâmica.

Referências

ADAMU, M. et al. London ambulance service software failure. In: . [S.l.: s.n.], 2010. Citado na página 21.

ANDERSON, D. J.; CARMICHAEL, A. *Kanban Essencial Condensado*. [S.l.]: Lean-Kanban University, 2016. ISBN 978-09845214-2-5. Citado na página 60.

ATLASSIAN. *Trello*. 2022. Disponível em: <https://trello.com/pt-BR>. Citado na página 49.

BATISTA, E. A.; CARVALHO, A. M. B. R. Uma taxonomia facetada para técnicas de elicitação de requisitos. In: *WER*. [S.l.: s.n.], 2003. p. 48–62. Citado 2 vezes nas páginas 35 e 37.

BELGAMO, A.; MARTINS, L. E. G. Estudo comparativo sobre as técnicas de elicitação de requisitos do software. In: *XX Congresso Brasileiro da Sociedade Brasileira de Computação (SBC), Curitiba–Paraná*. [S.l.: s.n.], 2000. Citado na página 38.

BOULILA, N.; HOFFMANN, A.; HERRMANN, A. Using storytelling to record requirements: Elements for an effective requirements elicitation approach. In: *2011 Fourth International Workshop on Multimedia and Enjoyable Requirements Engineering (MERE'11)*. [S.l.: s.n.], 2011. p. 9–16. Citado na página 37.

BROOKS, F. P. *The mythical man-month : essays on software engineering*. Anniversary edition. [S.l.]: Pearson Education, 1995. ISBN 0201835959. Citado na página 29.

CAIXETA, R. S. et al. *5W2H da aplicação iGado*. 2022. Disponível em: <https://pax-app.github.io/Wiki/#/>. Citado na página 34.

CAROLI, P. *Lean Inception: Como alinhar pessoas e construir o produto certo*. 1. ed. [S.l.]: Caroli, 2018. Citado na página 43.

CAROLI, P. *Product Backlog Building*. 1. ed. [S.l.]: Caroli, 2021. Citado 3 vezes nas páginas 38, 59 e 123.

CHUNG, L. et al. *Non-functional requirements in software engineering*. [S.l.]: Springer Science & Business Media, 2012. v. 5. Citado na página 39.

COUTO, A. D. A.; MARTINS, L. E. G. Uma proposta de validação de requisitos não funcionais utilizando o nfr-framework. 2008. Citado na página 39.

DISCORD, I. *Discord*. 2022. Disponível em: <https://discord.com/>. Citado na página 50.

DOCKER, I. *Docker Compose(versão 1.29.2)*. 2022. Disponível em: <https://docs.docker.com/compose/>. Citado na página 49.

DOCKER, I. *Docker (versão 4.5)*. 2022. Disponível em: <https://www.docker.com>. Citado na página 49.

DUROV, N.; DUROV, P. *Telegram*. 2022. Disponível em: <https://telegram.org/>. Citado na página 50.

ELLIOTT, R. A. Software requirements elicitation, verification, and documentation: an ontology based approach. 2012. Citado 3 vezes nas páginas 20, 35 e 64.

EVEN, S.; EVEN, G. *Graph Algorithms*. Cambridge University Press, 2011. ISBN 9781139504157. Disponível em: <https://books.google.com.br/books?id=m3QTSMYm5rkC>. Citado na página 46.

FACEBOOK, O. S. *ReactJS (versão 17.0.2)*. 2022. Disponível em: <https://reactjs.org/>. Citado na página 48.

FAGAN, M. E. Design and code inspections to reduce errors in program development. *IBM Systems Journal*, v. 15, n. 3, p. 182–211, 1976. Citado na página 42.

GERHARDT, T. E.; SILVEIRA, D. T. *Métodos de pesquisa*. [S.l.]: Plageder, 2009. Citado 2 vezes nas páginas 52 e 53.

GIL, A. C. et al. *Como elaborar projetos de pesquisa*. [S.l.]: Atlas São Paulo, 2002. v. 4. Citado 2 vezes nas páginas 53 e 61.

GOGUEN, J. A.; LINDE, C. Techniques for requirements elicitation. In: IEEE. *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*. [S.l.], 1993. p. 152–164. Citado na página 38.

GOOGLE. *Google Drive*. 2022. Disponível em: <https://www.google.com/intl/pt-PT/drive/>. Citado na página 50.

HOWARD, S. Using qfd to tame wicked information system's requirements. *Australasian Journal of Information Systems*, v. 7, n. 1, 1 1999. Disponível em: <https://journal.acs.org.au/index.php/ajis/article/view/285>. Citado na página 43.

JS, O. F. *NodeJS (versão 16.14.12)*. 2022. Disponível em: <https://nodejs.org>. Citado na página 48.

JUNIOR, P. R. D. O. B. Elicitação de requisitos de software através da utilização de questionários. 2005. Citado na página 36.

KALINOWSKI, M.; SPíNOLA, R. Introdução à inspeção de software - aumentando a qualidade através de verificações intermediárias. *Engenharia de Software Magazine*, I, p. 68–74, 01 2008. Citado na página 42.

KENDALL, K. E.; KENDALL, J. E. *Systems analysis and design*. [S.l.]: Prentice-Hall, Inc., 1992. Citado na página 36.

KING, T.; MARASCO, J. What is the cost of a requirement error. *StickyMinds www. stickyminds. com/sitewide. asp*, 2008. Citado na página 22.

KLEINBERG, J.; TARDOS, E. *Algorithm Design*. [S.l.]: Addison Wesley, 2006. Citado na página 45.

LABS, N. *Notion*. 2022. Disponível em: <https://www.notion.so>. Citado na página 51.

LAMPORT, L. *LaTeX*. 2022. Disponível em: <https://www.latex-project.org>. Citado na página 50.

LEITE, J. *Livro Vivo: Engenharia de Requisitos*. 2007. Citado na página 34.

LEITE, J. d. P.; FRANCO, A. P. M. A strategy for conceptual model acquisition. In: IEEE. *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*. [S.l.], 1993. p. 243–246. Citado na página 41.

MAZZOTTI, K.; BROEGA, A. C.; GOMES, L. A exploração da criatividade, através do uso da técnica de brainstorming, adaptada ao processo de criação em moda. In: *Anais do 1º Congresso Internacional de Moda e Design CIMODE, Guimarães (PT), Universidade do Minho*. [S.l.: s.n.], 2012. Citado na página 35.

MICROSOFT. *GitHub*. 2022. Disponível em: <https://github.com>. Citado na página 49.

MICROSOFT. *TypeScript*. 2022. Disponível em: <https://www.typescriptlang.org/>. Citado na página 48.

MILOSAVLJEVIC, M.; FIRTH-MCCOY, N.; LASLETT, S. *Usabilityhub*. 2008. Disponível em: <https://usabilityhub.com/>. Citado na página 115.

MONK, A.; HOWARD, S. Methods & tools: The rich picture: A tool for reasoning about work context. *Interactions*, Association for Computing Machinery, New York, NY, USA, v. 5, n. 2, p. 21–30, mar 1998. ISSN 1072-5520. Disponível em: <https://doi.org/10.1145/274430.274434>. Citado na página 33.

MOREIRA, L. F. C.; CAIXETA, R. S. *Base de código do iFlow*. 2022. Disponível em: <https://github.com/iflow-app>. Citado na página 81.

MOREIRA, L. F. C.; CAIXETA, R. S. *Código da tela de Backlog extraído do Frontend do iFlow*. 2022. Disponível em: <https://github.com/iflow-app/frontend/blob/main/src/pages/Artifacts/ArtifactInputs/Backlog/Backlog.tsx#L222>. Citado na página 83.

MOREIRA, L. F. C.; CAIXETA, R. S. *Documento de referências do iFlow*. 2022. Disponível em: <https://unb-rogerio.notion.site/Refer-ncias-c31bfce5dba543619a9d0913f4ba9a00>. Citado na página 58.

MOREIRA, L. F. C.; CAIXETA, R. S. *Protótipo navegável do iFlow*. 2022. Disponível em: <https://www.figma.com/proto/8itTrODQ9x6DeKR1nVluW4/TCC---iFlow?node-id=62:38&scaling=scale-down&page-id=0:1&starting-point-node-id=62:38>. Citado na página 70.

MOREIRA, L. F. C. et al. *5W2H da aplicação iGado*. 2022. Disponível em: <https://unbarqdsw.github.io/2020.1_G13_iGado/#/>. Citado 2 vezes nas páginas 33 e 35.

MOULTON, R. *Overleaf*. 2022. Disponível em: <https://www.overleaf.com/for/authors>. Citado na página 50.

NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: a roadmap. In: *Proceedings of the Conference on the Future of Software Engineering*. [S.l.: s.n.], 2000. p. 35–46. Citado na página 35.

PINHEIRO, F. A. Requirements traceability. In: *Perspectives on software requirements*. [S.l.]: Springer, 2004. p. 91–113. Citado 2 vezes nas páginas 32 e 35.

PRESTES, E. Introdução à teoria dos grafos. *Universidade Federal do Rio Grande do Sul, Instituto de Informática, Departamento de Informática Teórica, Tech. Rep*, 2016. Citado 2 vezes nas páginas 45 e 46.

RABUSKE, F. B. O uso das ferramentas brainstorming e 5w2h no planejamento de combate a incêndio em indústrias de tabaco. 2016. Citado na página 32.

SEGUNDO, A. N. et al. *Custom Survey: Uma Ferramenta para Criação e Distribuição de Questionários Customizados*. Dissertação (B.S. thesis) — Universidade Federal do Rio Grande do Norte, 2017. Citado na página 36.

SHARMA; TIWARI. *Figma*. 2021. Disponível em: <https://www.figma.com/>. Citado 2 vezes nas páginas 49 e 70.

SILVA, L. F.; CHAPETTA, W.; TRAVASSOS, G. Inspeções de requisitos de software utilizando pbr e apoio ferramental. In: *Anais do III Simpósio Brasileiro de Qualidade de Software*. Porto Alegre, RS, Brasil: SBC, 2004. p. 88–101. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/sbqs/article/view/16186>. Citado na página 42.

SOMMERVILLE, I. *Software Engineering*. 9. ed. Harlow, England: Addison-Wesley, 2010. ISBN 978-0-13-703515-1. Citado 3 vezes nas páginas 28, 37 e 38.

TORVALDS, L.; HAMANO, J. *Git (versão 2.35.1)*. 2022. Disponível em: <https://git-scm.com/>. Citado na página 49.

ULLAH, S.; IQBAL, M.; KHAN, A. M. A survey on issues in non-functional requirements elicitation. *International Conference on Computer Networks and Information Technology*, p. 333–340, 2011. Citado na página 40.

WALKER, M. et al. High-fidelity or low-fidelity, paper or computer choosing attributes when testing web prototypes. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, v. 46, 09 2002. Citado na página 70.

WESTFALL, L. Software requirements engineering: What, why, who, when, and how. p. 1–18, 01 2014. Citado 3 vezes nas páginas 27, 28 e 31.

ZANETTE, F. *MVP*. 2020. Disponível em: <https://resultadosdigitais.com.br/marketing/mvp-minimo-produto-viavel/>. Citado na página 43.

APÊNDICE A – *Backlog* da Ferramenta

O *Backlog* do Produto é um artefato essencial na Engenharia de Requisitos, na metodologia ágil e no ciclo de desenvolvimento de um produto, pois todos os requisitos constam nele. Em uma definição mais concreta, o *Backlog* é uma lista ordenada e emergente de tudo necessário no produto, ou seja, é a única fonte dos requisitos (CAROLI, 2021). Todas as funcionalidades são descritas no *Backlog* e é o que o cliente espera receber no fim do projeto. Além disso, algumas definições são importantes no *Backlog*, são elas:

- **Épico**: é uma grande parte de trabalho que, geralmente, é dividida em tarefas menores, ou seja, histórias de usuário;
- ***Feature***: é uma funcionalidade que faz parte de um módulo, possuindo seus requisitos funcionais e suas regras de negócio;
- **História de Usuário**: é uma função da *feature* e está associado a ela. Objetivamente, equivale aos requisitos funcionais de uma *interface*.

A Tabela 5 descreve as funcionalidades da ferramenta proposta, usando as definições descritas anteriormente, buscando proporcionar um melhor nível de detalhamento para ter requisitos mais consistentes e coerentes com as necessidades do produto.

Tabela 5 – *Backlog* do iFlow

ID	**Épico**	**ID**	**Feature**	**ID**	**História de Usuário**		
					Quem?	**O que?**	**Por quê?**
[EP001]	Artefatos	**[FE001]**	Pré-rastreabilidade	**[US001]**	Eu, como usuário,	desejo poder criar um 5W2H da minha aplicação,	para definir o escopo de atuação do meu projeto
				[US002]	Eu, como usuário,	desejo inserir o Rich Picture da minha aplicação, criado externamente,	para compor meu portfólio de requisitos
		[FE002]	Elicitação	**[US003]**	Eu, como usuário,	desejo documentar um *Brainstorming* realizado com meu time,	para poder extrair requisitos
				[US004]	Eu, como usuário,	desejo adicionar os dados obtidos de um questionário, realizado externamente,	para poder extrair requisitos
				[US005]	Eu, como usuário,	desejo adicionar um protótipo de baixa fidelidade, desenvolvido externamente,	para poder extrair requisitos

			[**US006**]	Eu, como usuário,	desejo criar um Storytelling,	para poder extrair requisitos
			[**US007**]	Eu, como usuário,	desejo criar um roteiro de entrevista,	para poder conduzir uma entrevista
			[**US008**]	Eu, como usuário,	desejo desenvolver uma Analise de Protocolo,	para poder extrair reuquisitos
	[**FE003**]	Modelagem	[**US009**]	Eu, como usuário,	desejo criar o *backlog* a partir das requisitos já extraídos,	para ter requisitos mais amadurecidos
			[**US010**]	Eu, como usuário,	desejo criar um NFR adaptado,	para ter requisitos não-funcionais amadurecidos
			[**US011**]	Eu, como usuário,	desejo criar Léxicos para minha aplicação,	para melhorar a definição dos termos usados na aplicação
			[**US012**]	Eu, como usuário,	desejo realizar a ligação entre os léxicos e as suas definições,	para melhorar a navegabilidade

	[**FE004**]	Verificação	[**US013**]	Eu, como usuário,	desejo fazer um *checklist*,	para corroborar a qualidade e o formato do artefato gerado
			[**US014**]	Eu, como usuário,	desejo ter pré-sugestões de comprovações baseadas na literatura,	para facilitar e usar as definições adequadas
			[**US015**]	Eu, como usuário,	desejo usufruir de uma interface adequada para relizar a verificação,	para realizar o procedimento corretamente
	[**FE005**]	Priorização	[**US016**]	Eu, como usuário,	desejo desenvolver o *House of Quality* a partir dos requisitos elicitados, correlacionando requisitos funcionais e não funcionais,	para identificar as necessidades do sistema

				[US017]	Eu, como usuário,	desejo gerar um *Minimum Viable Product* (MVP) a partir do do *House of Quality*,	para saber o que precisa ser desenvolvido com prioridade
[EP002]	Usuários	**[FE006]**	Gerenciamento de Contas	**[US018]**	Eu, como usuário,	desejo registrar uma conta de usuário,	para conseguir utilizar a ferramenta e criar um novo projeto
				[US019]	Eu, como usuário,	desejo deletar a minha conta de usuário,	para não utilizar mais a aplicação
				[US020]	Eu, como iFlow,	desejo validar o *e-mail* do usuário cadastrado,	para garantir a existência do *e-mail*
		[FE007]	Acesso	**[US021]**	Eu, como usuário,	desejo realizar o *login*,	para iniciar minha sessão
				[US022]	Eu, como usuário,	desejo realizar o *logout*,	para sair da minha sessão
				[US023]	Eu, como iFlow,	desejo controlar o acesso por meio de *token JSON Web Token* (JWT),	para garantir a segurança da ferramenta

		[**FE008**]	Manutenção de Contas	[**US024**]	Eu, como usuário,	desejo editar minha senha,	para manter a segurança dos meus dados
				[**US025**]	Eu, como usuário,	desejo editar meu perfil,	para alterar meus dados
[**EP003**]	Performance	[**FE009**]	Requisitos	[**US026**]	Eu, como iFlow,	desejo que os requisitos de todas as etapas sejam modelados como um grafo,	para viabilizar o cálculo do *Minimum Viable Product* (MVP)
				[**US027**]	Eu, como usuário,	desejo que os requisitos possam ser versionados e evoluídos,	para manter o rastro de um requisito desde a sua origem
		[**FE010**]	Utilidades	[**US028**]	Eu, como usuário,	desejo navegar entre as etapas já disponíveis,	para consertar ou adicionar algo que foi esquecido no desenvolvimento do processo
				[**US029**]	Eu, como usuário,	desejo visualizar, de forma agrupada, os artefatos gerados,	para apresentar para os *stakeholders* do projeto

				[**US030**]	Eu, como usuário,	desejo gerar um relatório dos artefatos gerados em cada processo,	para apresentar externamente ou usar como documentação
				[**US031**]	Eu, como usuário,	desejo ter uma fácil visualização das etapas da Engenharia de Requisitos,	para ter uma navegação mais intuitiva

Fonte: Autores, 2022

APÊNDICE B – Identidade Visual

Para representar a ferramenta, foi elaborado um logotipo que buscasse transmitir aos seus *stakeholders* a noção do processo proveniente da Engenharia de Requisitos. Por isso, foi escolhida uma onda para ser o principal símbolo para compor a identidade da ferramenta. Tal símbolo tem traços finos e arrojados que buscam expressar um ar mais moderno e dinâmico. As cores escolhidas buscam fornecer um tom mais elegante e minimalista, para conferir vida a ferramenta.

Todos os elementos e definições presentes na identidade visual do *iFlow* podem ser visualizados no documento na página a seguir.

Identidade Visual

Logotipos

Opção 1

Opção 2

Tipografia

A B C D E F G H I J K L M N O P Q R S T U V X W Y Z

a b c d e f g h i j k l m n o p q r s t u v x w y z

Fonte Secundária Roboto - Thin

A B C D E F G H I J K L M N O P Q R S T U V X W Y Z

a b c d e f g h i j k l m n o p q r s t u v x w y z

Cores

100%	75%	50%	
			#F4F1DE
			#E17A5F
			#3E405B
			#81B29A
			#F2CC8E

Símbolos

APÊNDICE C – Código da Base de Dados da Ferramenta *IFlow*

Código 3 – Código SQL referente ao banco de dados do iFlow

```
CREATE TYPE enum_artifact_stage AS ENUM ('pre-traceability', 'elicitation');
CREATE TYPE enum_functional_level_type AS ENUM ('epic', 'feature', 'user story');
CREATE TYPE enum_lexical_type AS ENUM ('verb', 'object', 'state');
CREATE TYPE enum_checkpoint_result_type AS ENUM ('boolean', 'int', 'text');

CREATE TABLE USER (
    user_id uuid DEFAULT uuid_generate_v4() NOT NULL,
    email varchar(250) NOT NULL,
    password varchar(255) NOT NULL,
    name varchar(200) NOT NULL,

    CONSTRAINT user_pk PRIMARY KEY (user_id),
);

CREATE TABLE REQUIREMENT (
    requirement_id uuid DEFAULT uuid_generate_v4() NOT NULL,
    name text NOT NULL,
    who text NOT NULL,
    what text NOT NULL,
    why text NOT NULL,
    artifact_id uuid DEFAULT uuid_generate_v4() NOT NULL,
    project_id uuid DEFAULT uuid_generate_v4() NOT NULL,
    requirement_evolves_id uuid DEFAULT,uuid_generate_v4()

```

```
25      CONSTRAINT requirement_pk PRIMARY KEY (requirement_id),
26      CONSTRAINT REQUIREMENT_ARTIFACT_FK FOREIGN KEY (artifact_id
            )
27          REFERENCES ARTIFACT(artifact_id),
28      CONSTRAINT REQUIREMENT_PROJECT_FK FOREIGN KEY (project_id)
29          REFERENCES PROJECT(project_id),
30      CONSTRAINT REQUIREMENT_REQUIREMENT_FK FOREIGN KEY (
            requirement_evolves_id)
31          REFERENCES REQUIREMENT(requirement_evolves_id),
32
33  );
34
35  CREATE TABLE ARTIFACT (
36      artifact_id uuid DEFAULT uuid_generate_v4() NOT NULL,
37      name varchar(50) NOT NULL,
38      description text NOT NULL,
39      stage enum_artifact_stage NOT NULL,
40      artifact_evolves_id uuid DEFAULT,uuid_generate_v4()
41      project_id uuid DEFAULT uuid_generate_v4() NOT NULL,
42
43      CONSTRAINT artifact_pk PRIMARY KEY (artifact_id),
44      CONSTRAINT ARTIFACT_PROJECT_FK FOREIGN KEY (project_id)
45          REFERENCES PROJECT(project_id),
46      CONSTRAINT ARTIFACT_ARTIFACT_FK FOREIGN KEY (
            artifact_evolves_id)
47          REFERENCES ARTIFACT(artifact_evolves_id),
48  );
49
50  CREATE TABLE PROJECT (
51      project_id uuid DEFAULT uuid_generate_v4() NOT NULL,
52      name varchar(50) NOT NULL,
53      user_id uuid DEFAULT uuid_generate_v4() NOT NULL,
54
55      CONSTRAINT project_pk PRIMARY KEY (project_id),
56      CONSTRAINT PROJECT_USER_FK FOREIGN KEY (user_id)
```

```
57          REFERENCES USER(user_id),
58
59 );
60
61 CREATE TABLE CONTENT (
62     content_id uuid DEFAULT uuid_generate_v4() NOT NULL,
63     content bytea NOT NULL,
64     type text NOT NULL,
65     artifact_id uuid DEFAULT uuid_generate_v4() NOT NULL,
66
67     CONSTRAINT content_pk PRIMARY KEY (content_id),
68     CONSTRAINT CONTENT_ARTIFACT_FK FOREIGN KEY (artifact_id)
69         REFERENCES ARTIFACT(artifact_id),
70 );
71
72 CREATE TABLE NON_FUNCTIONAL (
73     nfunctional_id uuid DEFAULT uuid_generate_v4() NOT NULL,
74     identifier varchar(5) NOT NULL,
75     priority smallint NOT NULL,
76     requirement_id uuid DEFAULT uuid_generate_v4() NOT NULL,
77
78     CONSTRAINT non_functional_pk PRIMARY KEY (nfunctional_id),
79     CONSTRAINT NON_FUNCTIONAL_REQUIREMENT_FK FOREIGN KEY (
           requirement_id)
80         REFERENCES REQUIREMENT(requirement_id),
81 );
82
83 CREATE TABLE FUNCTIONAL (
84     functional_id uuid DEFAULT uuid_generate_v4() NOT NULL,
85     level_type enum_functional_level_type NOT NULL,
86     identifier varchar(5) NOT NULL,
87     requirement_id uuid DEFAULT uuid_generate_v4() NOT NULL,
88
89     CONSTRAINT functional_pk PRIMARY KEY (functional_id),
90     CONSTRAINT FUNCTIONAL_REQUIREMENT_FK FOREIGN KEY (
```

```
        requirement_id)
91          REFERENCES REQUIREMENT(requirement_id),
92 );
93
94 CREATE TABLE LEXICAL (
95     lexical_id uuid DEFAULT uuid_generate_v4() NOT NULL,
96     notion text NOT NULL,
97     impact text NOT NULL,
98     type enum_lexical_type NOT NULL,
99     project_id uuid DEFAULT uuid_generate_v4() NOT NULL,
100
101    CONSTRAINT lexical_pk PRIMARY KEY (lexical_id),
102    CONSTRAINT LEXICAL_PROJECT_FK FOREIGN KEY (project_id)
103        REFERENCES PROJECT(project_id),
104 );
105
106 CREATE TABLE VERIFICATION (
107    verification_id uuid DEFAULT uuid_generate_v4() PRIMARY KEY
           NOT NULL,
108    name text NOT NULL,
109    artifact_id uuid DEFAULT uuid_generate_v4() NOT NULL,
110
111    CONSTRAINT verification_pk PRIMARY KEY (verification_id),
112    CONSTRAINT VERIFICATION_ARTIFACT_FK FOREIGN KEY (
           artifact_id)
113        REFERENCES ARTIFACT(artifact_id),
114 );
115
116 CREATE TABLE CHECKPOINT (
117    result text NOT NULL,
118    result_type enum_checkpoint_result_type NOT NULL,
119    criteria text NOT NULL,
120    verification_id uuid DEFAULT uuid_generate_v4() NOT NULL,
121
122    CONSTRAINT CHECKPOINT_VERIFICATION_FK FOREIGN KEY (
```

```
        verification_id)
123         REFERENCES VERIFICATION(verification_id),
124 );
125
126 CREATE TABLE house_of_quality (
127     nfunctional_id uuid DEFAULT uuid_generate_v4() NOT NULL,
128     functional_id uuid DEFAULT uuid_generate_v4() NOT NULL,
129     weight smallint NOT NULL,
130
131     CONSTRAINT house_of_quality_NON_FUNCTIONAL_FK FOREIGN KEY (
        nfunctional_id)
132         REFERENCES NON_FUNCTIONAL(nfunctional_id),
133     CONSTRAINT house_of_quality_FUNCTIONAL_FK FOREIGN KEY (
        functional_id)
134         REFERENCES FUNCTIONAL(functional_id),
135 );
```