UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

ALEX ALVES DE SOUSA

DESENVOLVIMENTO DE UMA FERRAMENTA PARA APOIO AO PRODUCT OWNER NO REFINAMENTO E PRIORIZAÇÃO DE HISTÓRIAS DE USUÁRIO

QUIXADÁ
2026

ALEX ALVES DE SOUSA

DESENVOLVIMENTO DE UMA FERRAMENTA PARA APOIO AO PRODUCT OWNER NO REFINAMENTO E PRIORIZAÇÃO DE HISTÓRIAS DE USUÁRIO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Orientadora: Prof. Ma. Leonara de Medeiros Braz

QUIXADÁ

2026

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

S696d Sousa, Alex Alves de.
Desenvolvimento de uma ferramenta para apoio ao Product Owner no refinamento e priorização de histórias de usuário / Alex Alves de Sousa. – 2026.
76 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Sistemas de Informação, Quixadá, 2026.
Orientação: Profa. Ma. Leonara de Medeiros Braz.

1. Product Owner. 2. Histórias de usuário. 3. Refinamento de backlog. 4. Priorização. I. Título.
CDD 005

ALEX ALVES DE SOUSA

DESENVOLVIMENTO DE UMA FERRAMENTA PARA APOIO AO PRODUCT OWNER NO REFINAMENTO E PRIORIZAÇÃO DE HISTÓRIAS DE USUÁRIO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação.

Aprovada em: 20/01/2026.

BANCA EXAMINADORA

_____

Profa. Ma. Leonara de Medeiros Braz (Orientadora)
Universidade Federal do Ceará (UFC)

_____

Profa. Ma. Antonia Diana Braga Nogueira
Universidade Federal do Ceará (UFC)

_____

Prof. Me. Ítalo Mendes da Silva Ribeiro
Universidade Estadual do Ceará (UECE)

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, por ter me concedido saúde, força e perseverança ao longo de toda a minha trajetória acadêmica.

Aos meus pais e avós pelo apoio constante, incentivo, cobrança e compreensão durante todo o período da graduação, especialmente nos momentos de maior dedicação e esforço. A minha namorada que sempre esteve comigo nos momentos bons e ruins me apoiando em toda caminhada, aos meus amigos que sempre estavam a meu dispor quando precisei e a minha orientadora Leonara pelo acompanhamento, pelas orientações precisas, pela paciência e pelas contribuições fundamentais para o amadurecimento deste trabalho.

RESUMO

O refinamento e a priorização de histórias de usuário são atividades essenciais no trabalho do *Product Owner* em ambientes ágeis, pois influenciam diretamente a qualidade do *backlog* e a previsibilidade das entregas. Apesar da ampla adoção de métodos ágeis, práticas recomendadas pela Engenharia de Requisitos Ágil, como critérios de prontidão e técnicas estruturadas de priorização, ainda são aplicadas de forma pouco sistemática. Diante desse cenário, este trabalho tem como objetivo desenvolver uma ferramenta para apoio ao *Product Owner* no refinamento e na priorização de histórias de usuário. A metodologia adotada envolve o entendimento do contexto por meio de questionário online e entrevistas com *Product Owners*, a elicitação dos requisitos da ferramenta, sua implementação como uma aplicação web e uma avaliação exploratória baseada na coleta de *feedbacks* qualitativos de usuários. Como resultado, foi desenvolvida a ferramenta Refyne, que apoia o refinamento do *backlog* por meio da aplicação sistemática do checklist INVEST e da priorização guiada utilizando técnicas como MoSCoW, CSD e GUT, além de oferecer recursos visuais para acompanhamento do estado das histórias. Conclui-se que a proposta demonstra a viabilidade de operacionalizar conceitos da Engenharia de Requisitos Ágil em uma solução prática de apoio ao trabalho do *Product Owner*.

Palavras-chave: Product Owner; histórias de usuário; refinamento de backlog; priorização.

ABSTRACT

User story refinement and prioritization are essential activities in the Product Owner's role within agile environments, as they directly affect backlog quality and delivery predictability. Although agile methods are widely adopted, practices recommended by Agile Requirements Engineering are often applied inconsistently. In this context, this work aims to develop a tool to support the Product Owner in the refinement and prioritization of user stories. The methodology includes context analysis through an online questionnaire and interviews, requirements elicitation, web application development, and an exploratory evaluation based on qualitative user feedback. As a result, the Refyne tool was developed, supporting backlog refinement through the systematic use of the INVEST checklist and guided prioritization techniques such as MoSCoW, CSD, and GUT, as well as visual support for monitoring user stories. The results indicate that the proposed solution is a feasible way to operationalize Agile Requirements Engineering concepts in practice.

Keywords: Product Owner; user stories; backlog refinement; prioritization.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

DoR	Definition Of Ready
API	Application Programming Interface

SUMÁRIO

1 INTRODUÇÃO

A busca por ciclos de entrega cada vez mais curtos, somada à crescente complexidade dos produtos digitais, impulsiona organizações de todos os portes a adotar *frameworks* ágeis. Entre eles, o Scrum continua hegemônico: o *17th State of Agile Report* indica que 63% das equipes que praticam métodos ágeis utilizam Scrum em nível de time (Digital.AI, 2023). No entanto, pesquisas nacionais apontam que, mesmo após a adoção formal do *framework*, persistem barreiras culturais e processuais que dificultam sua plena aplicação (Medeiros de Assis; Larieira; Costa, 2017).

A literatura acadêmica reforça o problema: estudos recentes identificam a ausência de consenso sobre critérios de Definição de Pronto (do inglês *Definition of Ready* - DoR) e sobre a aplicação da abordagem INVEST[1] como fatores que contribuem para retrabalho, perda de foco e baixa previsibilidade nas entregas (Riesen; Wagenaar, 2024). Esses dois processos são fundamentais para o refinamento do *backlog* voltado a entregas de valor ao cliente. No entanto, sua ausência ou aplicação inadequada compromete o planejamento da equipe e afeta a previsibilidade das entregas.

Parabol (2024), em seu estudo, reforça essa barreira: 51,8% das equipes relatam realizar sessões de refinamento síncronas, enquanto 18,5% afirmam fazê-las de forma assíncrona, deixando quase um terço das equipes sem qualquer ritual estruturado. Cabe observar que, embora os dados sejam representativos do cenário prático, o relatório não explicita o tamanho da amostra, o que impõe uma limitação metodológica à sua utilização.

Schwaber e Sutherland (2020) definem o refinamento como o processo contínuo de decompor, detalhar e priorizar itens até que estejam "prontos" (*ready*) para seleção na *Sprint Planning*[2]. Essa atividade visa adicionar informações como descrição, ordem e tamanho dos itens (Schwaber; Sutherland, 2020). Cabe ao *Product Owner* (PO) garantir a integridade do *Product Backlog*, equilibrando valor de negócio, qualidade e capacidade do time. Estudos mais recentes ampliam a compreensão desse papel, mapeando quatro domínios de atuação do PO: produto, eventos, comunicação e desenvolvimento (Heuer; Kremer; Kurtz, 2025).

Para que o *Product Owner* possa assegurar entregas estratégicas e alinhadas às capacidades do time, é necessário o uso de técnicas que apoiem tanto o refinamento quanto a priorização dos itens do *backlog*. A ausência de critérios sistematizados como *Definition of Ready* (DoR) e INVEST, nas ferramentas atualmente utilizadas, contribui para o surgimento

[1] Acrônimo do inglês - Independente, Negociável, Valioso, Estimável e Pequeno (small).
[2] Cerimônia para planejamento da *sprint*

de antipadrões, como *backlogs* superdimensionados e itens mal definidos, que podem comprometer a previsibilidade das entregas.

Nesse contexto, formula-se a seguinte questão de pesquisa: como apoiar o *Product Owner* na aplicação sistemática de critérios de refinamento (DoR/INVEST) e na priorização de itens do *backlog*, de modo a reduzir antipadrões e melhorar a previsibilidade das entregas?

Parte-se da hipótese de que o uso de uma ferramenta que integre mecanismos de verificação baseados na *Definition of Ready* e na abordagem INVEST, aliados a métodos estruturados de priorização como MoSCoW[3], Matriz CSD[4] e Matriz GUT[5] reduz a incidência de histórias mal definidas, promovendo entregas com maior previsibilidade e do alinhamento das entregas realizadas pelo *Product Owner* e pelo time ágil.

Diante desse cenário, a proposta é criar uma aplicação web que possibilite aos *Product Owners* o refinamento sistemático de seu *backlog*. Abaixo serão detalhados os objetivos geral e específicos deste trabalho.

1.1 OBJETIVOS

1.1.1 Objetivo geral

Desenvolver uma ferramenta que apoie times ágeis no refinamento e na priorização de histórias de usuário, por meio da aplicação de checagens baseadas em *Definition of Ready* (DoR) e INVEST.

1.1.2 Objetivos específicos

1. Identificar e documentar os requisitos funcionais e não funcionais da ferramenta.
2. Implementar a ferramenta como uma aplicação web independente.
3. Validar o uso da ferramenta junto a usuários reais.
4. Elaborar a documentação técnica e funcional da ferramenta.

[3] Técnica de priorização que classifica os requisitos em quatro categorias: Must have (deve ter), Should have (deveria ter), Could have (poderia ter) e Won't have (não terá).
[4] Método de apoio ao refinamento de backlog que organiza os itens em três colunas: Certezas, Suposições e Dúvidas.
[5] Método de priorização que avalia os itens com base em três critérios: Gravidade, Urgência e Tendência.

2 REFERENCIAL TEÓRICO

Esta seção apresenta os fundamentos que sustentam este trabalho, abordando a transição dos modelos tradicionais para as abordagens ágeis (2.1), os princípios e a estrutura do *framework* Scrum (2.2), práticas de refinamento como *Definition of Ready* e INVEST (2.3) e técnicas de priorização aplicadas ao *Product Backlog* (2.4).

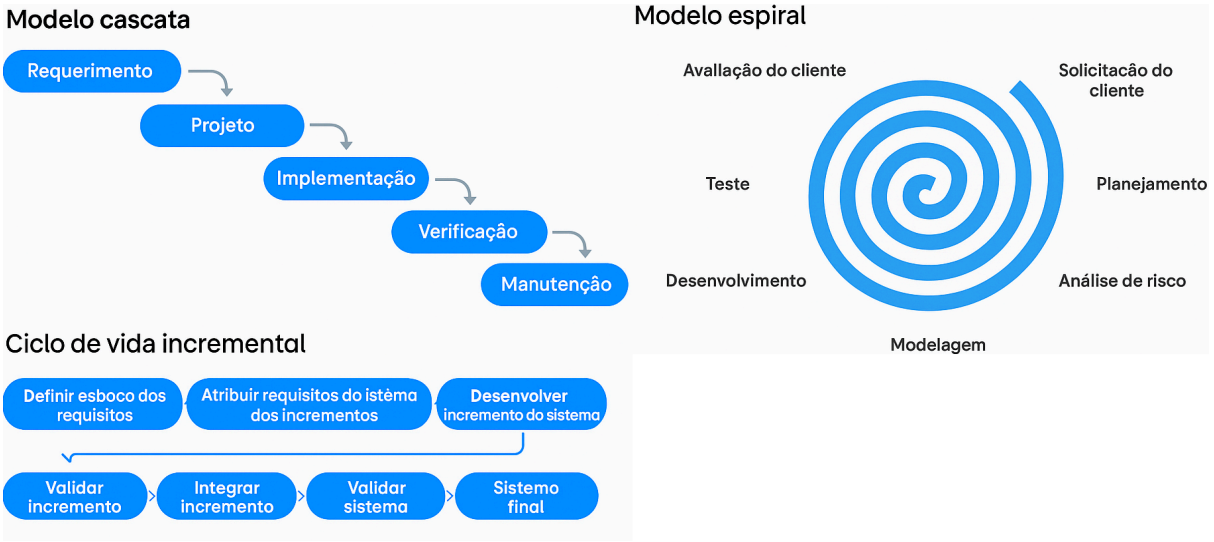
2.1 Engenharia de Software: Do Modelo Clássico às Abordagens Ágeis

A Engenharia de Software surgiu como disciplina formal na década de 1960, com o objetivo de aplicar métodos sistemáticos ao desenvolvimento de sistemas computacionais complexos. Seu propósito era mitigar os problemas recorrentes da chamada "crise do software", marcada por atrasos, custos excessivos e falhas críticas em projetos de software (Pressman; Maxim, 2016).

Durante décadas, modelos clássicos como o cascata, espiral e incremental dominaram o processo de desenvolvimento. O modelo cascata, em particular, consolidou-se como uma estrutura sequencial, composta por etapas rígidas: levantamento de requisitos, projeto, implementação, testes, implantação e manutenção. Esse modelo presume que os requisitos estejam completamente definidos desde o início do projeto e que mudanças posteriores sejam mínimas (Sommerville, 2011). Um dos principais problemas desses modelos é a baixa capacidade de adaptação a mudanças de requisitos, o que torna o processo lento e muitas vezes desalinhado com os objetivos reais dos usuários ao final do ciclo de entrega (Pressman; Maxim, 2016). Entretanto, no cenário contemporâneo de inovação contínua e necessidades voláteis do mercado, as abordagens tradicionais mostram-se cada vez menos eficazes.

Como tentativa de superar as limitações do modelo cascata, surgiram modelos como o incremental e o espiral. O modelo incremental propõe o desenvolvimento do sistema por meio de ciclos curtos com entregas parciais, permitindo certo grau de adaptação. No entanto, ainda enfrenta dificuldades ao lidar com mudanças significativas após o início dos incrementos, pois alterações podem exigir reestruturações em componentes já entregues (Pressman; Maxim, 2016). Já o modelo espiral, introduzido por Barry Boehm, combina elementos do desenvolvimento incremental com uma forte ênfase na análise e mitigação de riscos. Apesar de sua robustez conceitual, o espiral é considerado complexo e oneroso de ser implementado, o que limita sua adoção em projetos de menor porte (Sommerville, 2011). A Figura 1 representa os modelos Cascata, Incremental e Espiral.

Figura 1 – Modelos de Desenvolvimento de Software: Cascata, Incremental e Espiral



Fonte**:** adaptado de UDS Tecnologia(2023)

Com base nas limitações dos métodos tradicionais de desenvolvimento, surgiram ao final da década de 1990 movimentos que propunham uma nova abordagem de gestão e construção de software. Esses movimentos valorizavam ciclos iterativos, colaboração direta com o cliente e foco na geração de valor contínuo. Essas ideias viriam a se consolidar em 2001, com a formulação do Manifesto Ágil, documento que estabeleceu os princípios norteadores das metodologias ágeis contemporâneas (Beck et al., 2001).

A adoção das metodologias ágeis representa, portanto, uma resposta direta às deficiências dos modelos tradicionais. Em vez de tratar o software como um produto totalmente especificado a priori, os métodos ágeis reconhecem sua natureza evolutiva, especialmente em projetos digitais sujeitos a mudanças frequentes no escopo, na estratégia de negócio e na base de usuários. Com a sua difusão, vários métodos ágeis foram implementados, tais como XP, Kanban e Scrum. A próxima seção apresenta o *framework* Scrum, atualmente o método ágil mais utilizado pelas equipes de desenvolvimento.

2.2 O Framework Scrum: Origem, Princípios e Estrutura

O Scrum consolidou-se, ao longo das últimas décadas, como um dos *frameworks* ágeis mais difundidos na indústria de software, especialmente após a publicação do Manifesto Ágil em 2001, que estabeleceu os valores e princípios fundamentais da agilidade (Beck et al., 2001). Inspirado por esses valores, o Scrum propõe um modelo empírico e adaptativo, baseado em entregas incrementais, inspeção contínua e colaboração constante entre os

membros do time. Ao longo dos anos, sua estrutura evoluiu com base na prática de milhares de equipes em diversos contextos, mantendo como essência a entrega frequente de valor e a capacidade de adaptação a ambientes complexos e em constante mudança.

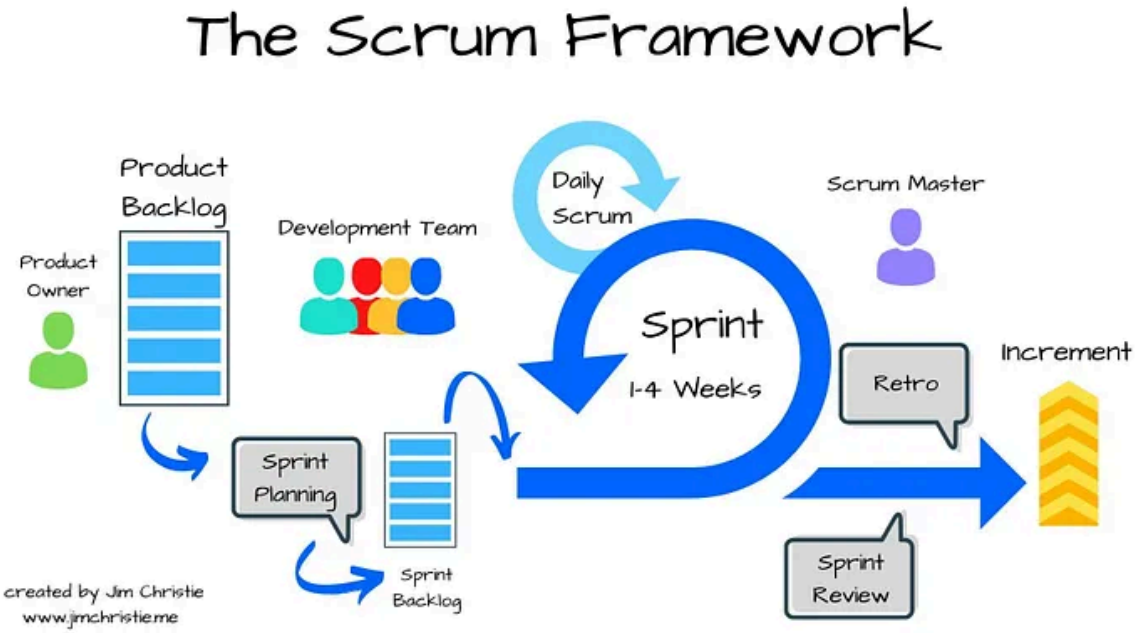
Embora tenha ganhado ampla popularidade após o Manifesto Ágil, o Scrum foi formalmente introduzido por Ken Schwaber e Jeff Sutherland ainda na década de 1990. Sua concepção original foi inspirada em uma analogia com o *rugby*, conforme proposto por Takeuchi e Nonaka (1986), sugerindo uma abordagem colaborativa, onde o time avança de forma conjunta rumo ao objetivo, com autonomia e responsabilidade compartilhada. Em oposição aos modelos sequenciais tradicionais, o Scrum promove ciclos curtos de trabalho, *feedback* frequente e entregas incrementais, que possibilitam a inspeção contínua tanto do produto quanto do processo.

De acordo com Schwaber e Sutherland (2020), o Scrum é um *framework* leve que ajuda pessoas, equipes e organizações a gerar valor por meio de soluções adaptativas para problemas complexos. Seu funcionamento baseia-se nos pilares da transparência, inspeção e adaptação, e seu ciclo de trabalho é estruturado em eventos fixos (*Sprints*) que promovem ritmo constante de entrega e melhoria contínua.

O Scrum define três papéis fundamentais para a condução do trabalho de forma colaborativa e adaptativa. O *Product Owner* (PO) é o responsável por maximizar o valor do produto, sendo incumbido de gerenciar e priorizar o *Product Backlog* com base no valor de negócio, nas necessidades dos *stakeholders* e na capacidade do time. O *Scrum Master*, por sua vez, atua como um facilitador do processo Scrum, removendo impedimentos, promovendo a adoção dos valores e práticas ágeis e assegurando que a equipe compreenda e aplique corretamente o *framework*. Já o Time de Desenvolvimento (*Developers*) é composto pelos profissionais que irão transformar os itens selecionados do *backlog* em incrementos de produto utilizáveis e potencialmente entregáveis ao final de cada *Sprint*. Juntos, esses papéis formam um time multifuncional e auto-organizado, essencial para garantir entregas iterativas, valor contínuo e capacidade de adaptação às mudanças.

Além disso, o Scrum organiza-se por meio de eventos regulares: *Sprint Planning*, *Daily Scrum*, *Sprint Review* e *Sprint Retrospective*, e artefatos centrais como o *Product Backlog*, o *Sprint Backlog* e o *Increment*. Na Figura 2, é possível visualizar uma representação geral do fluxo de trabalho dentro do Scrum, com os principais eventos e artefatos. Um aspecto crucial é o *Product Backlog Refinement*, uma atividade contínua opcional que permite ao time revisar, detalhar e preparar os itens do *backlog* de forma progressiva, tornando-os prontos para as futuras *Sprints*.

Figura 2 – Representação do framework Scrum



Fonte: Christie (2020)

Cada um desses eventos desempenha um papel específico dentro do ciclo iterativo, promovendo os pilares fundamentais do Scrum: transparência, inspeção e adaptação. A *Sprint Planning* é o evento que inaugura cada *Sprint*, no qual o *Product Owner* apresenta os itens priorizados do *Product Backlog*, e o time, em colaboração, define o que será entregue e como o trabalho será executado. Ao final desse encontro, estabelece-se o *Sprint Backlog*, contendo o objetivo da *Sprint* e o plano de ação para atingi-lo. Em seguida, durante a execução da *Sprint*, ocorre diariamente o *Daily Scrum*, um breve encontro com duração máxima de quinze minutos. Nessa reunião, os desenvolvedores sincronizam suas atividades, compartilham o progresso realizado, identificam possíveis impedimentos e ajustam o plano da *Sprint* conforme necessário.

Ao término do ciclo, realiza-se a *Sprint Review*, momento dedicado à inspeção do incremento desenvolvido e à coleta de *feedback* dos *stakeholders*. Essa reunião tem como finalidade apresentar o que foi concluído ao *stakeholder*, adaptar o *Product Backlog* se necessário e promover um alinhamento contínuo entre o time e os objetivos do produto. Por fim, encerra-se o ciclo com a *Sprint Retrospective*, evento reservado à inspeção do processo de trabalho da equipe. Nessa ocasião, os membros discutem abertamente o que funcionou bem, o que pode ser melhorado e quais ações serão implementadas para aprimorar a próxima *Sprint*. Segundo Schwaber e Sutherland (2020), esses eventos não apenas estruturam o ritmo

do trabalho em Scrum, mas também consolidam um ambiente de aprendizado contínuo e entrega de valor sustentável.

Os três artefatos centrais do Scrum, *Product Backlog*, *Sprint Backlog* e *Increment*, oferecem transparência e rastreabilidade ao progresso do trabalho, sendo essenciais para a entrega iterativa de valor. O *Product Backlog* é uma lista ordenada e dinâmica de tudo o que é necessário para o produto, sendo a única fonte de requisitos para qualquer alteração a ser feita. Ele é continuamente atualizado pelo *Product Owner*, à medida que novas informações surgem, permitindo que o *backlog* evolua junto com o produto. Já o *Sprint Backlog* é o subconjunto do *Product Backlog* selecionado para a *Sprint*, acompanhado por um plano detalhado sobre como os desenvolvedores irão transformar esses itens em um incremento funcional. Ele reflete a previsão do time sobre o que será entregue e é frequentemente ajustado durante a *Sprint* conforme o trabalho avança.

O terceiro artefato, o *Increment*, representa a soma de todos os itens do *Product Backlog* completados durante a *Sprint* atual e anteriores. Ele deve estar em condições de uso, independentemente de o *Product Owner* optar ou não por liberá-lo. Um incremento só é considerado pronto quando cumpre a *Definition of Done* acordada pela equipe, garantindo um padrão mínimo de qualidade e completude.

Embora o *Product Backlog Refinement* não seja um evento formalmente prescrito no Scrum, ele é considerado uma prática essencial para manter a qualidade do *backlog*. Segundo Schwaber e Sutherland (2020), o refinamento é uma atividade contínua em que o *Product Owner*, com apoio do time, revisa, detalha, decompõe e reordena os itens do *backlog*, a fim de garantir que estejam suficientemente claros e viáveis para serem selecionados em uma *Sprint*. Rubin (2013) complementa que a ausência de um refinamento estruturado pode gerar desperdícios, retrabalho e baixa previsibilidade, comprometendo a eficiência do time. Assim, mesmo não sendo um evento obrigatório, o *Product Backlog Refinement* desempenha papel fundamental na maturidade dos itens do *backlog* e na fluidez do processo de desenvolvimento. O *backlog* do produto é composto por pequenas histórias de usuário que contemplam o que deve ser feito e quais os critérios de aceitação para que a funcionalidade seja considerada correta.

No contexto ágil, histórias de usuário são amplamente utilizadas como forma de representar os requisitos, por descreverem funcionalidades sob a perspectiva do usuário final. Elas têm origem nos métodos ágeis como XP (Extreme Programming), mas foram amplamente adotadas no Scrum por sua simplicidade, flexibilidade e foco no valor de negócio. Uma história bem escrita facilita o entendimento compartilhado entre equipe e

stakeholders, favorecendo a colaboração e a adaptação às mudanças.

Segundo Cohn (2004), uma boa história de usuário deve expressar quem precisa de determinada funcionalidade, o que deseja fazer e por quê, frequentemente utilizando a estrutura: *Como [tipo de usuário], eu quero [ação] para que [benefício]*. Essa forma narrativa promove empatia com o usuário e ajuda o time a compreender o propósito por trás da funcionalidade.

Além disso, histórias de usuário devem ser refinadas continuamente para garantir clareza, viabilidade e valor. Como serão detalhadas adiante, práticas como *Definition of Ready* e INVEST auxiliam nesse processo, fornecendo critérios objetivos que indicam se uma história está pronta para ser selecionada em uma *Sprint*.

2.3 Práticas Ágeis de Refinamento: Definition of Ready e INVEST

O *Product Backlog Refinement* é uma prática essencial no Scrum, embora não configurado como evento oficial nem artefato, mas sim uma atividade contínua que complementa os artefatos e eventos formais do *framework*. Deste modo, o refinamento do *backlog* do produto é uma prática essencial, onde a qualidade e clareza dos itens a serem desenvolvidos impactam diretamente na previsibilidade, produtividade e valor entregue ao cliente. Embora não seja formalmente um evento do Scrum, o *Product Backlog Refinement* é recomendado como uma atividade contínua realizada pelo *Product Owner* com apoio do time, com o objetivo de decompor, detalhar e priorizar os itens até que estejam suficientemente claros e viáveis para desenvolvimento (Schwaber; Sutherland, 2020).

Neste contexto, emergem práticas complementares que buscam apoiar a eficiência do refinamento. Entre as mais utilizadas estão a *Definition of Ready* (DoR) e a abordagem INVEST, ambas voltadas a garantir que os itens do *backlog* estejam maduros o suficiente para serem puxados para a *Sprint Planning*.

A *Definition of Ready* (do inglês, definição de pronto) consiste em um conjunto de critérios que um item do *backlog* (geralmente uma história de usuário) deve satisfazer antes de ser considerado “pronto” para desenvolvimento. A DoR não é prescrita no Scrum, conforme definido por Schwaber e Sutherland (2020), mas surge como uma forma de mitigar retrabalho, reduzir ambiguidade e alinhar expectativas entre os membros da equipe (Rubin, 2013). Em geral, a DoR é representada por um checklist personalizado por cada time, contendo pontos como: a história está bem descrita? Há critérios de aceitação? O item foi estimado? Há dependências resolvidas?

Apesar de sua adoção disseminada, a aplicação da *Definition of Ready* apresenta desafios práticos. Um estudo recente de Riesen e Wagenaar (2024) investigou como as equipes ágeis definem o que significa um item estar “pronto”, identificando significativa heterogeneidade nos critérios utilizados, nos responsáveis por aplicá-los e nas regras adotadas. A pesquisa revelou que muitos times operam com concepções ambíguas de prontidão, o que compromete a previsibilidade das entregas e aumenta o risco de retrabalho. Esses achados reforçam a necessidade de padronizar a DoR por meio de mecanismos sistemáticos, como aqueles propostos neste trabalho.

Nesse sentido, van der Meer (2022), apresenta evidências empíricas sobre a aplicação da DoR em equipes Scrum de diferentes contextos organizacionais. O estudo demonstra que a DoR é comumente utilizada como um checklist durante o refinamento ou incorporada por meio de templates estruturados de requisitos, atuando como um mecanismo sistemático para verificar a prontidão das histórias antes de sua inclusão na *Sprint*. Além disso, o autor aponta que a ausência de critérios claros favorece a entrada de itens pouco compreendidos no *backlog* da *Sprint*, aumentando o risco de retrabalho e comprometendo a previsibilidade das entregas.

Por sua vez, a abordagem INVEST, proposta por Bill Wake (2003), define um conjunto de atributos desejáveis para a escrita de histórias de usuário eficazes. O acrônimo representa os termos *Independent* (Independente), que indica que a história pode ser desenvolvida isoladamente; *Negotiable* (Negociável), pois ainda está aberta a diálogo e ajustes; *Valuable* (Valiosa), por entregar valor ao cliente ou usuário; *Estimable* (Estimável), no sentido de que o esforço necessário pode ser avaliado pelo time; *Small* (Pequena), o suficiente para ser concluída dentro de uma *Sprint*; e *Testable* (Testável), o que exige critérios de aceitação claros e objetivos. Segundo Caroli e Aguiar (2021), a adoção de critérios claros no *backlog* contribui não apenas para maior eficiência do time, mas também para maior alinhamento com o valor de negócio, além de evitar antipadrões como *backlog* inflado ou pouco priorizado.

É importante destacar que essas práticas são especialmente valiosas em ambientes complexos, onde a incerteza é alta e as mudanças são frequentes. Dessa forma, práticas como DoR e INVEST não apenas aumentam a maturidade dos itens do *backlog*, mas também fortalecem a previsibilidade e a sustentabilidade do processo ágil de entrega contínua.

2.4 Técnicas de priorização no Backlog

Em projetos de desenvolvimento ágil, a priorização dos itens do *Product Backlog* é uma atividade estratégica (Schwaber; Sutherland, 2020). Com o crescimento da complexidade dos produtos digitais e das demandas dos *stakeholders*, torna-se essencial adotar critérios sistemáticos para orientar a ordem das entregas. Nesse cenário, o *Product Owner* assume papel estratégico ao utilizar métodos de apoio à priorização que reforcem a objetividade das decisões.

Entre as abordagens estruturadas para esse fim, destacam-se a técnica MoSCoW, a Matriz CSD e a Matriz GUT, todas aplicáveis no contexto ágil por permitirem classificar itens com base em múltiplos critérios, como valor, esforço, urgência, impacto e risco.

A técnica MoSCoW surgiu no contexto de desenvolvimento Incremental RAD (do inglês *Rapid Application Development*) e foi consolidada posteriormente no método ágil DSDM (*Dynamic Systems Development Method*), conforme relata Oliveira (2014). Seu acrônimo representa quatro níveis de prioridade, mostrados no Quadro 1, que permitem classificar requisitos de modo a preservar, mesmo sob severas restrições de tempo e recursos, aquilo que é indispensável ao sucesso da entrega.

Quadro 1 – Acrônimo MoSCoW

Deve ter (*Must have*)	Descreve um requisito que deve ser atendido na solução final para que a mesma seja considerada um sucesso
Deveria ter (*Should have*)	Representa um item de alta prioridade que deveria ser incluído na solução, caso possível
Poderia ter (*Could have*)	Descreve um requisito que é considerado desejável, mas não necessário, e que será incluído caso o tempo e os recursos permitam
Não terá (*Won't have*)	Representa um requisito que as partes interessadas concordaram em não implementar em uma determinada entrega, mas que pode ser considerado no futuro

Fonte: Adaptado de Oliveira (2014)

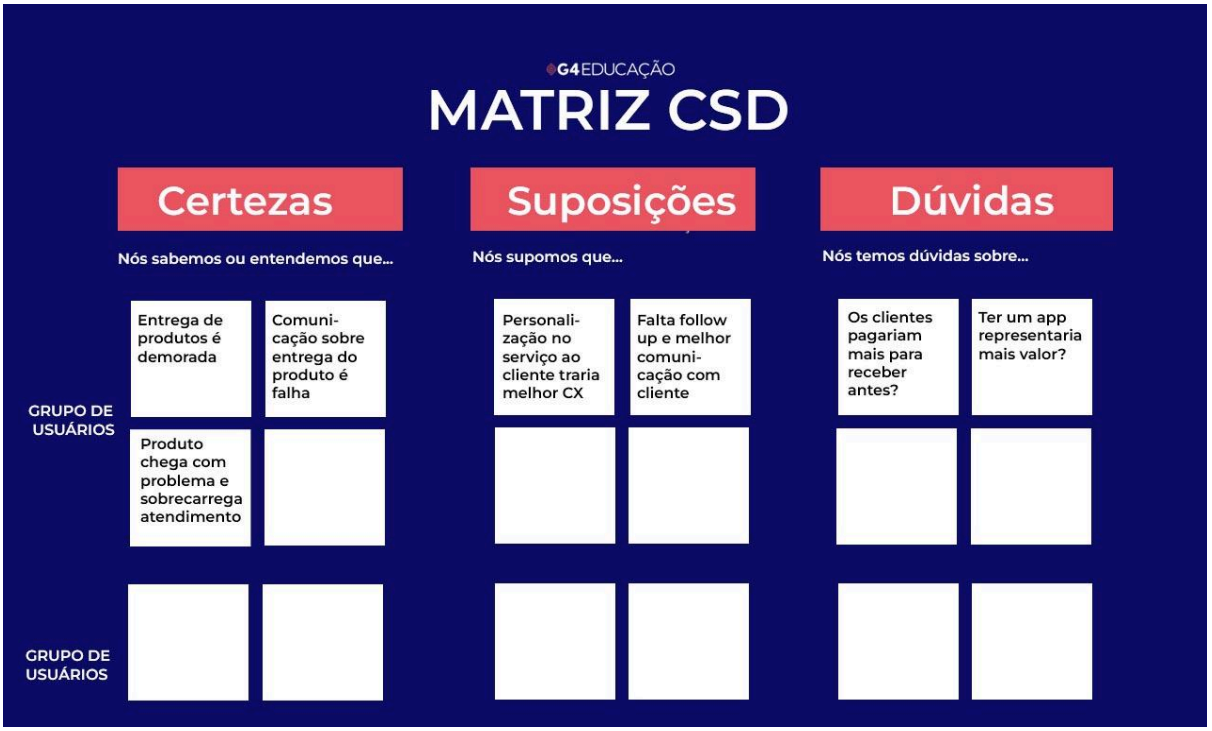
Segundo Oliveira (2014), a MoSCoW promove o alinhamento entre as expectativas das partes interessadas e a capacidade real de entrega da equipe ao estimular discussões

colaborativas durante *workshops* de definição de escopo. Ao distinguir nitidamente o que é essencial do que pode ser adiado ou descartado, a técnica fornece um critério objetivo para decisões de corte ou ajuste de escopo ao longo dos ciclos de desenvolvimento, garantindo maior previsibilidade e racionalidade ao processo de gestão de projetos iterativos.

Neste trabalho, a técnica MoSCoW será utilizada como apoio à priorização de histórias de usuário, permitindo distinguir, de forma objetiva, requisitos essenciais daqueles que podem ser adiados ou descartados, conforme as restrições da *Sprint*.

A Matriz CSD organiza a análise em três colunas: Certezas, Suposições e Dúvidas, ilustradas na Figura 3. Essa estrutura leva o grupo a explicitar o que já é comprovado, o que ainda requer validação e o que permanece desconhecido. Bretas (2015, apud Santos et al., 2018) descreve a ferramenta como um diagnóstico que amplia a visão sobre o problema, evidenciando riscos e evitando decisões baseadas apenas na intuição. Ao exigir que cada ponto seja classificado nesses eixos, a matriz estimula reflexão crítica antes da definição de estratégias, reduzindo vieses de confirmação e favorecendo discussões mais informadas sobre prioridades.

Figura 3 – Exemplo de Matriz CSD



Fonte: G4 Educação (2023)

Embora a CSD tenha sido concebida como ferramenta de diagnóstico em gestão da informação (Santos et al., 2018), neste trabalho ela é adaptada para servir de insumo à priorização do *Product Backlog*. A lógica proposta consiste em o *Product Owner* distinguir histórias ancoradas em evidências, Certezas, que tendem a receber prioridade, daquelas ainda sustentadas por hipóteses (Suposições) ou por pontos desconhecidos (Dúvidas). Dessa forma, a ordem de desenvolvimento passa a refletir não só valor de negócio, mas também o grau de confiança nos requisitos, possibilitando decisões mais conscientes e fundamentadas sobre a sequência de entregas.

A Matriz GUT é uma técnica que apoia a priorização de problemas com base em três critérios fundamentais: Gravidade (G), Urgência (U) e Tendência (T). Cada critério é avaliado por meio de uma escala de 1 a 5, onde 1 representa impacto mínimo e 5 indica impacto máximo. O valor final da prioridade é calculado pela multiplicação dos três fatores (G × U × T), resultando em um escore que indica a criticidade de cada item.

Os significados atribuídos a cada pontuação estão organizados no Quadro 2, enquanto o Quadro 3 apresenta a forma de cálculo e a classificação da prioridade com base no escore obtido. Esses quadros foram adaptados de Camargo (2018), conforme apresentados por Cevada e Damy-Benedetti (2022), que aplicaram a técnica em auditorias internas com foco na priorização de não conformidades críticas.

Quadro 2 – Significados dos critérios da Matriz GUT

Pontos	**Gravidade (G)**	**Urgência (U)**	**Tendência (T)**
1	Sem gravidade	Não tem pressa	Não vai piorar ou pode até melhorar
2	Pouco grave	Pode esperar um pouco	Vai piorar a longo prazo
3	Grave	O mais cedo possível	Vai piorar a médio prazo
4	Muito grave	Com alguma urgência	Vai piorar a curto prazo
5	Extremamente grave	Exige ação imediata	Agravamento imediato se nada for feito

Fonte**:** Adaptado de Cevada e Damy-Benedetti (2022), com base em Camargo (2018)

Quadro 3 – Classificação da prioridade com base no escore G × U × T

Fórmula aplicada	**Classificação da prioridade**
$0 < G \times U \times T \leq 45$	Baixa
$46 \leq G \times U \times T \leq 95$	Média
$96 \leq G \times U \times T \leq 125$	Alta

Fonte**:** Adaptado de Cevada e Damy-Benedetti (2022), com base em Camargo (2018)

A força da Matriz GUT reside em sua objetividade e simplicidade. Ao atribuir pesos numéricos claros, a técnica contribui para o alinhamento entre áreas técnicas e gerenciais, minimizando decisões subjetivas baseadas em percepções individuais ou pressões externas. Camargo (2018) destaca que a matriz torna os critérios de impacto transparentes e comparáveis, favorecendo uma visão estratégica compartilhada por todos os envolvidos.

Neste trabalho, propõe-se a adaptação da GUT ao contexto ágil, utilizando-a como um recurso adicional de priorização no *Product Backlog*, especialmente para classificar dúvidas técnicas, problemas de usabilidade ou melhorias essenciais. Essa abordagem mostra-se particularmente útil em cenários de alta demanda e restrição de recursos, fornecendo um processo racional e comunicável para decisões de priorização especialmente em ambientes de desenvolvimento ágil, onde decisões rápidas e bem fundamentadas impactam diretamente na previsibilidade e no valor entregue ao cliente.

3 TRABALHOS RELACIONADOS

Esta seção revisita cinco estudos que dialogam com o problema deste TCC. O objetivo é evidenciar como cada trabalho aborda aspectos de estruturação, refinamento ou priorização do *Product Backlog*, justificando a necessidade de uma proposta integrada.

3.1 Backlog maker

Almeida e Queiroz (2021) propõem o desenvolvimento da ferramenta Backlog Maker, destinada a apoiar a criação e manutenção estruturada do *Product Backlog* em projetos ágeis. A ferramenta oferece organização hierárquica dos requisitos (Épico, Feature, História e Tarefa), bem como recursos como atribuição de membros, checklists, estimativas, anexos e comentários.

O desenvolvimento da solução utilizou uma abordagem híbrida, integrando práticas ágeis consagradas, como Scrum, Kanban e Extreme Programming (XP). Inicialmente, foi realizada uma Revisão Sistemática da Literatura (RSL) para identificar problemas frequentes enfrentados pelas equipes ágeis na gestão do *backlog* e levantar práticas recomendadas para mitigá-los. Em seguida, as autoras realizaram atividades para elicitação e especificação de requisitos, definição da arquitetura tecnológica da ferramenta, e implementação utilizando tecnologias web modernas, como React, Next.js, Node.js e Firebase.

Para validar a ferramenta desenvolvida, o trabalho adotou a abordagem qualitativa de pesquisa-ação, realizada em dois ciclos iterativos com usuários reais. O *feedback* obtido possibilitou ajustes incrementais na interface e nas funcionalidades, demonstrando eficácia no atendimento às necessidades das equipes Scrum em contextos reais.

Embora a ferramenta se mostre adequada para equipes pequenas e médias que adotam Scrum, a Backlog Maker não tem como foco a incorporação de mecanismos explícitos de verificação de critérios de refinamento, como *Definition of Ready* (DoR) e INVEST, nem a adoção de métodos sistematizados de priorização. Tais aspectos, que extrapolam o escopo da proposta original das autoras, constituem elementos centrais da presente pesquisa. Enquanto a Backlog Maker enfatiza a documentação e a organização dos requisitos, a presente proposta amplia esse enfoque, ao incorporar mecanismos de avaliação automatizada da qualidade dos itens e oferecer apoio estruturado à priorização.

3.2 Estudo de caso: O uso de user stories aplicado na construção de um backlog de produto

O estudo de Alves (2022) aborda a construção do *backlog* do sistema ToSeguro, utilizando técnicas da Engenharia de Requisitos Ágeis. Com base em um protótipo de alta fidelidade e entrevistas com usuários, os requisitos foram priorizados e documentados como histórias de usuário, aplicando critérios de aceitação, *Definition of Ready* (DoR) e INVEST. A adoção dessas técnicas visa melhorar a clareza, testabilidade e relevância das histórias de usuário, conforme orientam as boas práticas da Engenharia de Requisitos Ágeis, contribuindo potencialmente para a qualidade do produto.

A metodologia empregada por Alves (2022) consistiu inicialmente no levantamento dos requisitos a partir da análise de um protótipo de alta fidelidade e da aplicação de questionários junto aos usuários, visando identificar as funcionalidades mais relevantes e utilizadas. Após esse levantamento inicial, as histórias de usuário foram estruturadas conforme a técnica dos 3C's (Cartão, Conversa e Confirmação) e organizadas na ferramenta Trello. Cada história passou por uma validação utilizando o modelo INVEST, sendo avaliada quanto à sua Independência, Negociabilidade, Valor para o negócio, possibilidade de Estimativa, Pequenez (adequação de tamanho) e Testabilidade. Além disso, foram estabelecidos critérios específicos de *Definition of Ready* (DoR), garantindo que cada história estivesse devidamente preparada para inclusão em futuras reuniões de planejamento das releases.

Apesar dos resultados positivos, o estudo de Alves (2022) concentra-se na aplicação dessas técnicas na fase inicial de modelagem do *backlog*, não tendo como objetivo propor uma ferramenta automatizada para apoiar o refinamento contínuo e sistematizado dos itens. Do mesmo modo, não são explorados métodos estruturados de priorização, como MoSCoW ou GUT, os quais integram o escopo da presente proposta. Ainda assim, o estudo fornece uma validação empírica relevante sobre a eficácia das técnicas INVEST e DoR em contextos reais, servindo como base conceitual e motivação para a solução automatizada e contínua proposta por este TCC.

3.3 Proposta de priorização de requisitos de software utilizando a matriz MoSCoW e o método multicritério SAPEVO-M

Varella et al. (2021) apresentam uma abordagem que combina a matriz MoSCoW com o método multicritério SAPEVO-M, objetivando uma priorização mais estruturada e objetiva

dos requisitos. O método proposto busca reduzir a subjetividade em contextos com múltiplos stakeholders, ao considerar critérios como valor de negócio, urgência e viabilidade técnica. A abordagem envolve a definição colaborativa dos critérios, aplicação do método SAPEVO-M para cálculo dos pesos e, posteriormente, utilização da matriz MoSCoW para categorizar requisitos em *Must*, *Should*, *Could* ou *Won't Have*.

Do ponto de vista metodológico, os autores conduzem um estudo de caso em um projeto de help desk para uma empresa do setor de energia. Inicialmente, os *stakeholders* definem de forma colaborativa os critérios de decisão e as escalas de comparação par a par; em seguida, preenchem as matrizes de julgamento no SAPEVO-M, cujo algoritmo transforma preferências ordinais em pesos normalizados. Esses pesos alimentam o cálculo dos escores agregados de cada requisito, o que permite ranquear as funcionalidades de acordo com a importância relativa obtida. Por fim, o ranking é refletido na matriz MoSCoW, estabelecendo um *backlog* ordenado que serve de base para o roadmap do produto.

A contribuição desse trabalho para este TCC reside em demonstrar, com dados reais, como combinar uma técnica quantitativa de apoio à decisão e uma matriz de classificação simples, evidenciando que decisões de *backlog* podem e devem ser sustentadas por análises multicritério quando vários atores e interesses estão em jogo.

Importa destacar, contudo, que o escopo de Varella et al. concentra-se na priorização estratégica dos requisitos, não tendo como objetivo a incorporação de práticas específicas de refinamento de prontidão, como DoR, INVEST ou CSD. Além disso, embora o SAPEVO-M ofereça rigor analítico, sua aplicação envolve comparações par a par e matrizes que podem se mostrar mais complexas para equipes ágeis de menor porte ou em estágios iniciais de maturidade

3.4 REACT-M: uma abordagem ágil para o gerenciamento de requisitos de software

Silva et al. apresentam o método REACT-M, uma abordagem ágil voltada ao apoio sistemático às atividades de gerência de requisitos em projetos de desenvolvimento de software. A proposta resulta de um mapeamento sistemático da literatura e da integração de práticas consagradas da Engenharia de Requisitos Ágil com modelos de qualidade amplamente adotados, como CMMI-DEV e MR-MPS-SW, buscando alinhar rigor de processo com princípios ágeis.

O REACT-M estrutura o processo de requisitos em atividades como identificação de *stakeholders*, elaboração e avaliação do *backlog*, rastreabilidade, tratamento de mudanças e

correção de inconsistências. Um aspecto central da abordagem é a avaliação da prontidão dos requisitos antes de sua inclusão no ciclo de desenvolvimento, realizada por meio da *Definition of Ready* (DoR) como acordo de qualidade entre especialistas de domínio e equipe técnica, combinada ao uso sistemático dos critérios INVEST para verificação da qualidade das histórias de usuário. Os autores propõem a operacionalização desses critérios por meio de perguntas orientadoras aplicadas durante o refinamento, garantindo que os itens sejam independentes, negociáveis, valiosos, estimáveis, pequenos e testáveis.

Do ponto de vista metodológico, o trabalho descreve a construção do método a partir de evidências empíricas extraídas da literatura e sua avaliação por revisão por pares, com especialistas em modelos de qualidade e métodos ágeis, evidenciando preocupações com validade técnica, rastreabilidade e controle de mudanças. A abordagem enfatiza a necessidade de critérios explícitos de qualidade para evitar a entrada de requisitos pouco compreendidos nos ciclos de desenvolvimento, reduzindo riscos de retrabalho e ambiguidades.

Embora o REACT-M contribua ao estruturar o uso de DoR e INVEST no processo de requisitos, sua proposta concentra-se na definição de um método, sem o objetivo de oferecer uma ferramenta computacional para apoiar o refinamento contínuo ou a priorização do *backlog*. Além disso, técnicas de priorização utilizadas por *Product Owners*, como MoSCoW, CSD e GUT, não fazem parte do seu escopo. De forma complementar, este TCC propõe uma plataforma web que operacionaliza esses princípios, integrando verificação de prontidão, priorização guiada e apoio visual ao processo de tomada de decisão.

3.5 Análise e aprimoramento de requisitos: um estudo de user stories na perspectiva dos desenvolvedores

Souza (2024) apresenta um estudo voltado à análise da qualidade de requisitos em projetos de desenvolvimento de software, com foco específico em user stories como principal artefato de especificação em ambientes ágeis. O trabalho parte da constatação de que, embora amplamente utilizadas, as histórias de usuário frequentemente apresentam problemas de clareza, completude e detalhamento técnico, o que compromete a comunicação entre equipes e impacta a previsibilidade das entregas.

A metodologia adotada baseia-se na análise de user stories públicas disponíveis em repositórios na internet, complementada por um questionário aplicado a desenvolvedores, com o objetivo de identificar dificuldades recorrentes e validar práticas de melhoria. A autora estrutura sua avaliação com base nos princípios INVEST e em critérios de qualidade como

clareza e completude, analisando histórias de diferentes domínios, como e-commerce, segurança, APIs e sistemas de gestão.

Os resultados evidenciam problemas frequentes, como ausência de critérios de aceitação claros, falta de informações técnicas relevantes e a presença de histórias excessivamente amplas, que dificultam a estimativa, a testabilidade e a implementação. Como contribuição, o trabalho propõe um conjunto de recomendações práticas e um guia para apoiar equipes no aprimoramento contínuo da escrita de user stories.

Embora o estudo de Souza (2024) forneça uma análise empírica consistente sobre a qualidade das histórias e valide a relevância dos critérios INVEST, sua proposta permanece no nível analítico e orientativo, não contemplando a implementação de uma ferramenta para apoiar o refinamento contínuo ou a aplicação sistemática desses critérios no cotidiano das equipes. Aspectos como priorização estruturada do *backlog* e apoio visual à tomada de decisão não fazem parte de seu escopo. De forma complementar, o presente TCC propõe a materialização desses princípios em uma plataforma web que integra verificação de prontidão, priorização guiada e mecanismos visuais de apoio ao processo de refinamento.

3.6 Análise Comparativa

O Quadro 4 sintetiza as principais características dos trabalhos analisados, destacando suas contribuições ao refinamento, à qualidade e à priorização do *Product Backlog*. A comparação é realizada a partir de sete critérios: natureza da proposta, objetivo principal, uso explícito de *Definition of Ready* (DoR), uso explícito de INVEST, presença de critérios de aceitação ou checklists de qualidade, adoção de priorização estruturada e público-alvo. Essa análise evidencia que, embora os estudos abordem dimensões relevantes do problema de forma isolada como organização do *backlog*, qualidade das histórias ou apoio à priorização.

Quadro 4 – Comparação entre os trabalhos relacionados e a proposta deste TCC

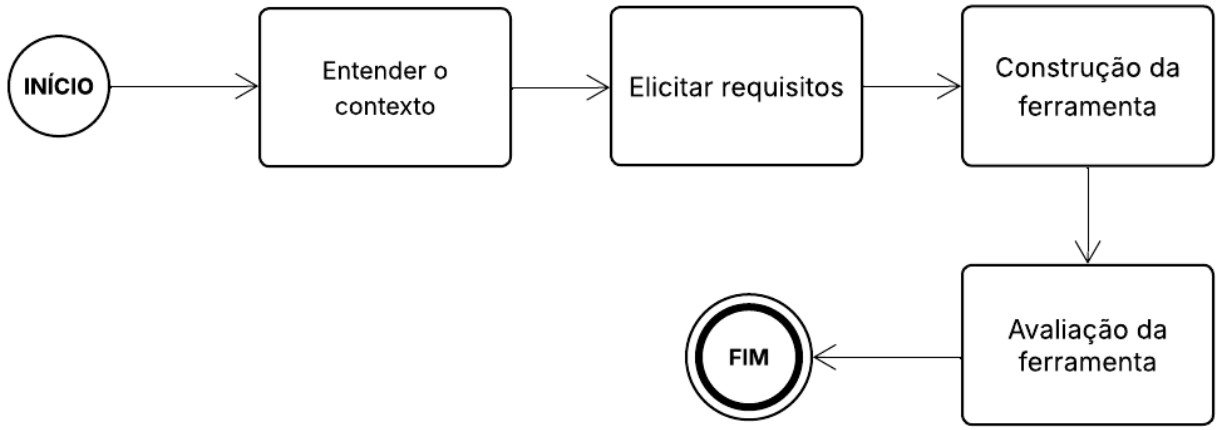
Critério	Backlog Maker (2021)	Alves (2022)	Varella et al. (2021)	REACT-M (Silva et al.)	Souza (2024)	Este TCC (2025)
Natureza da proposta	Ferramenta web	Estudo de caso acadêmico	Método de priorização	Método de Engenharia de Requisitos Ágil	Estudo analítico	Ferramenta web
Objetivo principal	Estruturação e organização do backlog	Aplicação de user stories com DoR e INVEST	Priorização estruturada de requisitos	Gerência de requisitos com foco em qualidade	Análise e melhoria da qualidade das user stories	Refinamento, priorização e apoio à decisão
Uso explícito de DoR	Não	Sim	Não	Sim	Parcial (implícito)	Sim
Uso explícito de INVEST	Não	Sim	Não	Sim	Sim	Sim
Critérios de aceitação / checklist de qualidade	Sim	Sim	Não	Sim	Sim	Sim
Priorização estruturada	Não	Não	Sim (MoSCoW + SAPEVO-M)	Não	Não	Sim (MoSCoW, CSD, GUT)
Público-alvo	Equipes ágeis	Contexto acadêmico	Stakeholders em decisões complexas	Engenheiros de requisitos e equipes ágeis	Desenvolvedores	Product Owners e times ágeis

Fonte: Elaborado pelo autor

4 METODOLOGIA

Esta seção apresenta os procedimentos metodológicos adotados ao longo do trabalho, organizados em quatro etapas principais, conforme ilustrado na Figura 4: (i) entendimento do contexto, (ii) elicitação de requisitos, (iii) construção da ferramenta e (iv) avaliação da solução.

Figura 4 – Fluxograma das atividades executadas



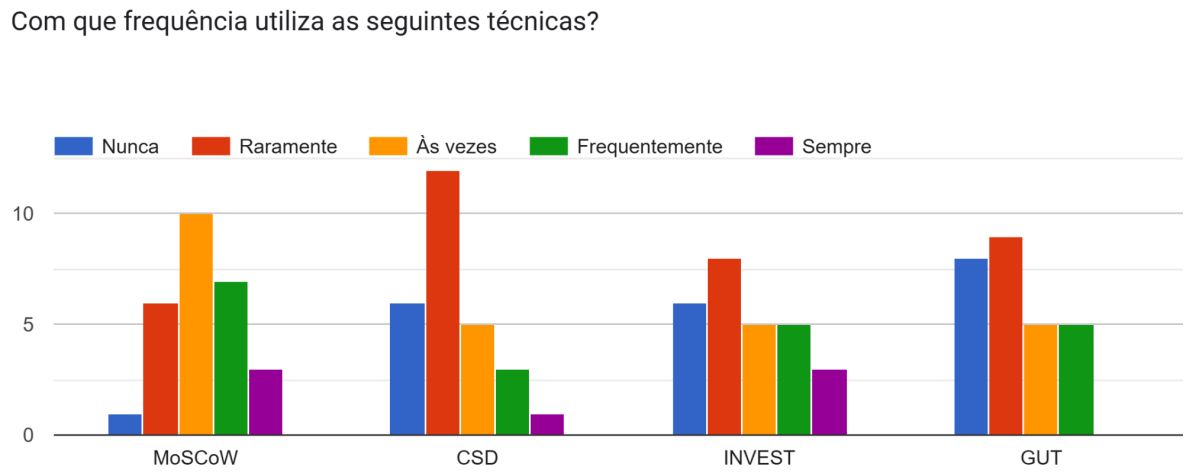
Fonte: Elaborada pelo autor

4.1 Entendimento do contexto

A etapa de entendimento do contexto teve como finalidade compreender as dificuldades enfrentadas por *Product Owners* durante o refinamento e a priorização de histórias de usuário em ambientes ágeis. Para isso, foram utilizados dois instrumentos de coleta de dados: um questionário online, com 27 respostas válidas, e entrevistas semiestruturadas conduzidas com três *Product Owners*, permitindo a combinação de dados quantitativos e qualitativos.

Os resultados provenientes do questionário indicam que técnicas como MoSCoW, INVEST, CSD e GUT apresentam níveis variados de adoção prática entre os participantes. Conforme ilustrado na Figura 5, observa-se que a técnica MoSCoW possui maior frequência relativa de uso, concentrando-se principalmente nas categorias “às vezes” e “frequentemente”. Em contrapartida, técnicas como CSD e GUT apresentam baixa utilização por parte de uma parcela significativa dos respondentes. O INVEST, apesar de reconhecido como uma boa prática para a escrita de histórias de usuário, não é aplicado de forma sistemática pela maioria dos participantes do questionário.

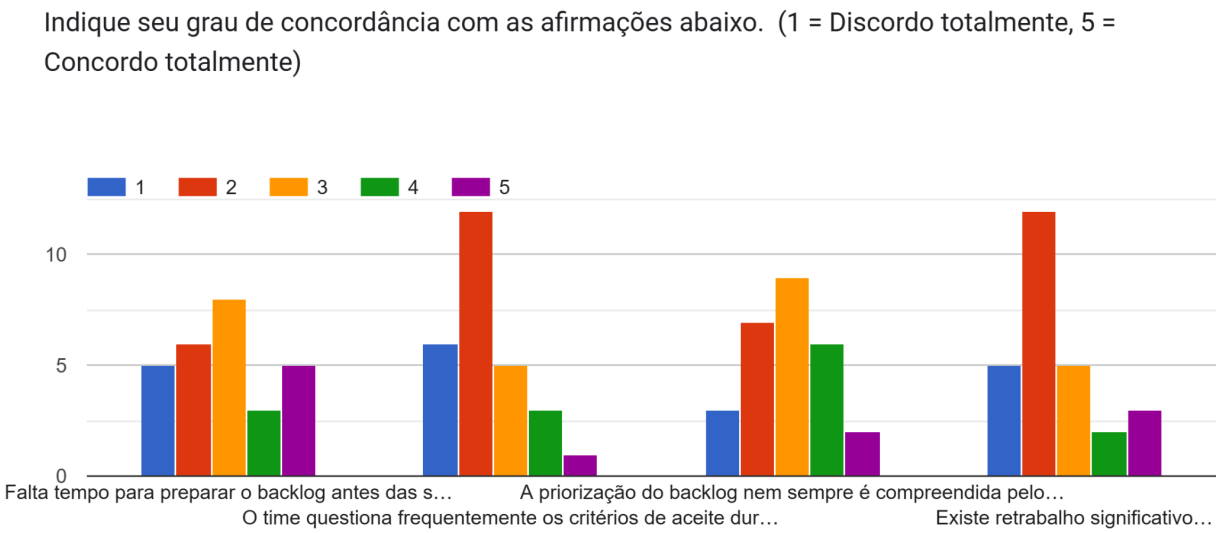
Figura 5 – Frequência de uso de técnicas de refinamento e priorização (MoSCoW, CSD, INVEST e GUT).



Fonte: Elaborado pelo autor com base nos dados da pesquisa (2025).

Em relação às dificuldades percebidas no processo de refinamento e priorização, os dados do questionário, apresentados na Figura 6, evidenciam a recorrência de problemas como falta de tempo para preparação do *backlog*, questionamentos frequentes sobre critérios de aceite durante a implementação, incompreensão da priorização por parte dos *stakeholders* e ocorrência de retrabalho decorrente de falhas no refinamento. Embora parte dos respondentes discorde parcialmente dessas afirmações, observa-se uma concentração relevante de respostas nos níveis intermediários e superiores da escala de concordância, indicando que tais dificuldades estão presentes no cotidiano de parte significativa dos participantes.

Figura 6 – Percepções dos Product Owners sobre dificuldades no refinamento e na priorização do backlog.



Fonte: Elaborado pelo autor com base nos dados da pesquisa (2025).

As entrevistas semiestruturadas complementam os dados quantitativos ao permitir uma compreensão mais aprofundada do contexto organizacional e das práticas adotadas pelos *Product Owners*. Os relatos evidenciaram dificuldades relacionadas à definição consistente de critérios de aceitação, à ausência de registro formal das decisões de priorização e à carência de mecanismos visuais que apoiem a visão macro do *backlog* e a comunicação com stakeholders.

De forma integrada, os dados do questionário e das entrevistas permitiram identificar lacunas entre as práticas recomendadas pela literatura e aquelas efetivamente adotadas nos contextos investigados. Esses achados serviram como base empírica para a definição do problema e para a elicitação dos requisitos da ferramenta proposta, reforçando a necessidade de mecanismos que sistematizam o refinamento, a priorização e a rastreabilidade das decisões no *Product Backlog*.

Os resultados completos do questionário encontram-se disponíveis no Apêndice A, enquanto as transcrições integrais das entrevistas estão apresentadas no Apêndice B.

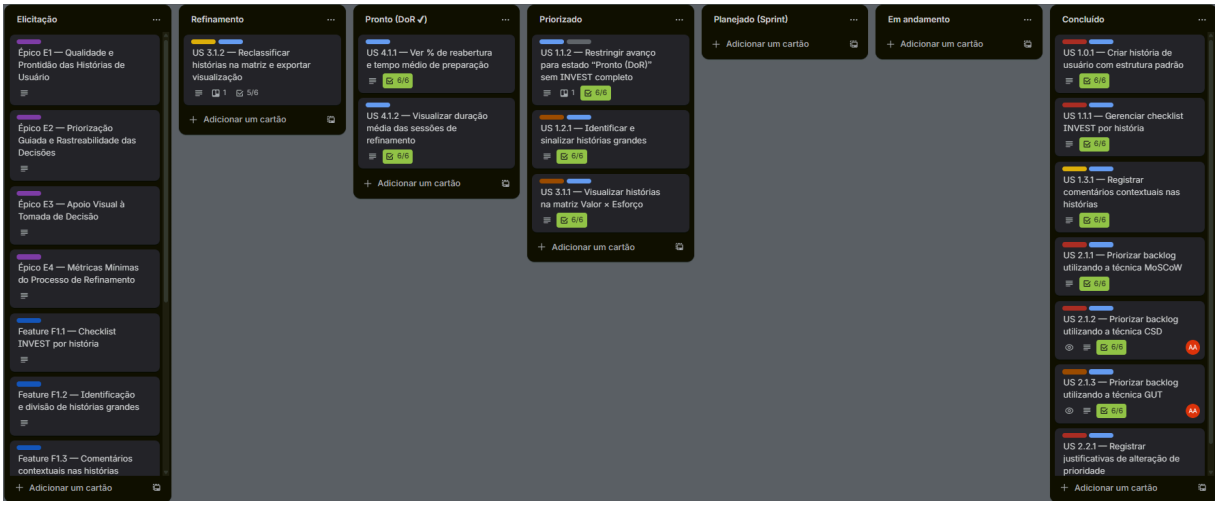
4.2 Elicitação de requisitos

Com base no entendimento do contexto, procedeu-se à elicitação dos requisitos da ferramenta proposta. Essa etapa foi conduzida de forma iterativa, utilizando histórias de usuário como principal artefato de especificação, alinhando-se às práticas do Scrum. O processo de elicitação foi apoiado por um quadro no Trello, que permitiu organizar, refinar e acompanhar a evolução do *backlog* ao longo do projeto.

Antes da consolidação do *backlog* no Trello, foram elaborados artefatos iniciais de elicitação e análise de requisitos, nos quais as necessidades identificadas a partir do questionário e das entrevistas foram traduzidas em requisitos funcionais, não funcionais e histórias de usuário preliminares. Esses artefatos tiveram como objetivo apoiar a compreensão do domínio do problema e orientar a posterior consolidação do *backlog*, não representando, necessariamente, o conjunto final de funcionalidades implementadas na ferramenta. O documento completo desses artefatos encontra-se disponível no Apêndice C.

O fluxo do *backlog* foi estruturado em listas que representam as principais etapas do processo de refinamento e desenvolvimento, possibilitando visualizar o estado de cada item desde sua concepção até sua conclusão. A partir desse fluxo, os requisitos foram organizados hierarquicamente em épicos, *features* e histórias de usuário. Foram definidos quatro épicos principais, Épico E1 — Qualidade e Prontidão das Histórias de Usuário, Épico E2 — Priorização Guiada e Rastreabilidade das Decisões, Épico E3 — Apoio Visual à Tomada de Decisão e Épico E4 — Métricas Mínimas do Processo de Refinamento, cada um associado a problemas identificados na etapa de entendimento do contexto, contemplando aspectos relacionados à qualidade e prontidão das histórias, à priorização guiada, ao apoio visual à decisão e à definição de métricas mínimas do processo de refinamento, o *backlog* completo do Trello encontra-se disponível no Apêndice D. A Figura 7 ilustra a organização e separação entre histórias em refinamento, prontas e priorizadas.

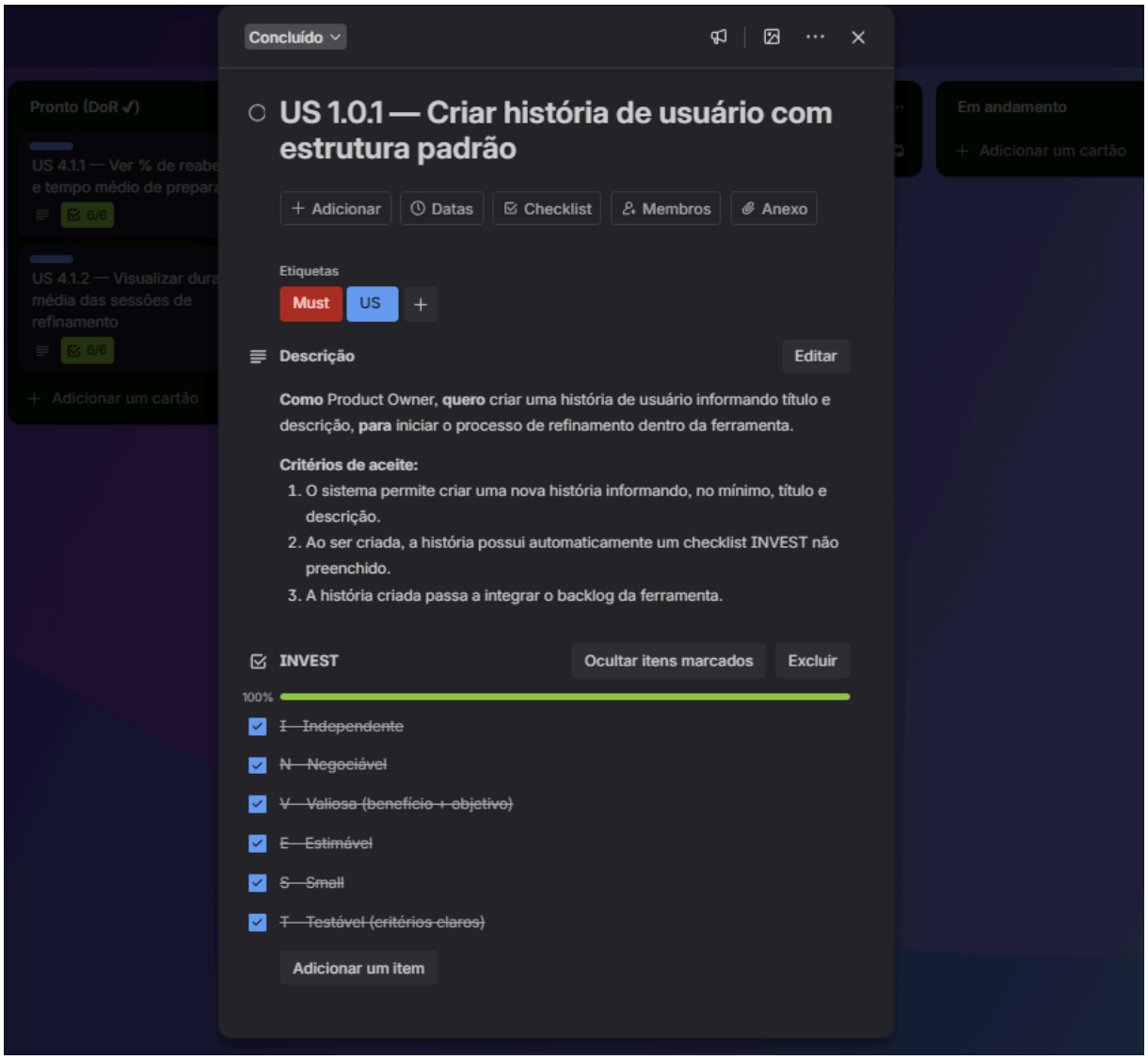
Figura 7 – Organização e fluxo do *backlog* no Trello.



Fonte: Elaborada pelo autor

As histórias de usuário foram escritas no formato padrão “Como [persona], quero [ação], para [benefício]” e acompanhadas de critérios de aceite claros e objetivos. Durante o refinamento, foi adotado o checklist INVEST como instrumento de apoio à avaliação da qualidade das histórias, permitindo verificar aspectos como valor, tamanho adequado, estimabilidade e testabilidade. As histórias que atendiam a esses critérios eram consideradas refinadas, enquanto aquelas que apresentavam lacunas permaneciam em refinamento até atingirem um nível mínimo de prontidão. A Figura 8 ilustra uma das histórias de usuário elaboradas no Trello.

Figura 8 – Exemplo de história de usuário no Trello com critérios de aceite e checklist *INVEST*



Fonte: Elaborada pelo autor

Além disso, foi utilizada a técnica MoSCoW como principal mecanismo de priorização do *backlog*, permitindo classificar as histórias de acordo com sua importância relativa para a construção da ferramenta. Técnicas como CSD e GUT, embora discutidas na fundamentação teórica, não foram aplicadas diretamente no *backlog* do projeto, permanecendo como referência conceitual para análise e comparação.

4.3 Construção da ferramenta

A construção da ferramenta seguiu uma abordagem orientada a *sprints*. Cada *sprint* contemplou um conjunto de histórias previamente refinadas e priorizadas, respeitando os

limites de escopo definidos para o trabalho. O processo de desenvolvimento buscou refletir, na prática, os conceitos defendidos pela própria ferramenta, especialmente no que se refere à importância do refinamento prévio e da clareza dos requisitos antes da implementação. O Quadro 5 apresenta o *roadmap* do projeto, no qual é possível observar a distribuição das histórias ao longo das *sprints*, bem como a separação entre atividades de refinamento e de desenvolvimento.

Quadro 5 – Roadmap do projeto com separação entre etapas de refinamento (R) e desenvolvimento (D)

ÉPICOS	**SPRINT 1 (20/10 - 24/10)**	**SPRINT 2 (27/10 - 07/11)**	**SPRINT 3 (10/11 - 21/11)**	**SPRINT 4 (24/11 - 05/12)**
E1 — Qualidade e Prontidão das Histórias de Usuário	F1.1 - US 1.0.1 : R; F1.1 - US 1.1.1 : R; F1.2 - US 1.1.2 : R; F1.1 - US 1.2.1 : R;	F1.1 - US 1.0.1 : D; F1.1 - US 1.1.1 : D; F1.1 - US 1.3.1 : R;	F1.1 - US 1.3.1 : D;	
E2 — Priorização Guiada e Rastreabilidade das Decisões		F2.1 - US 2.1.1: R; F2.1 - US 2.1.2: R; F2.1 - US 2.1.3: R;	F2.1 - US 2.1.1: D; F2.1 - US 2.1.2: D; F2.2 - US 2.2.1: R;	F2.1 - US 2.1.3: D; F2.2 - US 2.2.1: D;
E3 — Apoio Visual à Tomada de Decisão			F3.1 - US 3.1.1: R; F3.1 - US 3.1.2: R;	
E4 — Métricas Mínimas do Processo de Refinamento				F4.1 - US 4.1.1: R; F4.1 - US 4.1.2: R;

Fonte: Elaborada pelo autor

A solução foi desenvolvida como uma ferramenta web, utilizando Next.js na versão 15 com *App Router* para a estruturação da aplicação, React em conjunto com Chakra UI para a construção da interface, Supabase como backend, com uso de PostgreSQL e serviços de

autenticação, e Clerk para o gerenciamento de autenticação e controle de acesso. Essa *stack* foi escolhida por oferecer uma combinação adequada entre produtividade, escalabilidade e simplicidade arquitetural, compatível com o escopo acadêmico do projeto.

Nem todas as histórias elicitas foram implementadas durante o desenvolvimento. A decisão de restringir a implementação a um subconjunto do *backlog* esteve alinhada aos princípios do Scrum e da técnica MoSCoW, priorizando a entrega de um MVP funcional capaz de demonstrar a viabilidade da proposta e atender aos objetivos centrais do trabalho. As histórias classificadas como "*Must*" e parte das "*Should*" foram priorizadas para implementação, enquanto as demais permaneceram como escopo futuro.

4.4 Avaliação da ferramenta

A avaliação da ferramenta ocorreu de forma exploratória e qualitativa, com foco na obtenção de percepções iniciais sobre sua utilidade e aderência ao processo de refinamento e priorização do backlog. Após a implementação das funcionalidades previstas para o escopo do trabalho, o link de acesso à ferramenta foi disponibilizado a *Product Owners* que participaram das entrevistas semiestruturadas, bem como a colegas e profissionais da área de desenvolvimento de software com experiência em métodos ágeis.

Esses usuários realizaram testes livres da ferramenta e forneceram *feedbacks* informais, de caráter qualitativo, a respeito da clareza da interface, do fluxo de criação e refinamento de histórias de usuário e do apoio oferecido pelas funcionalidades relacionadas ao checklist INVEST e à priorização do *backlog*. Embora não tenham sido aplicados instrumentos formais de avaliação de usabilidade ou aceitação, os retornos obtidos permitiram validar, de forma preliminar, o uso da ferramenta junto a usuários reais, atendendo ao objetivo específico de validação proposto neste trabalho.

Os *feedbacks* indicaram que a ferramenta apresenta potencial para apoiar a organização do *backlog*, tornar mais explícito o estado de refinamento das histórias e auxiliar o *Product Owner* na aplicação sistemática de critérios de qualidade e priorização. Ainda assim, reconhece-se como limitação a ausência de uma avaliação estruturada, o que motiva a realização de estudos futuros com métodos formais de avaliação de usabilidade, aceitação e impacto no processo de refinamento.

5 RESULTADOS

Este capítulo apresenta os resultados obtidos a partir do desenvolvimento da solução proposta neste trabalho. Os resultados aqui apresentados concentram-se na materialização da solução, representada pela ferramenta Refyne, bem como em sua estrutura técnica e funcionalidades principais.

5.1 Projeto Refyne

O projeto Refyne, uma ferramenta web desenvolvida com o objetivo de apoiar *Product Owners* no processo de refinamento e priorização de histórias de usuário. O projeto busca traduzir conceitos teóricos da Engenharia de Requisitos ágil em funcionalidades práticas e acessíveis.

A ferramenta foi desenvolvida utilizando o *framework* Next.js, na versão 15, com adoção do *App Router*, arquitetura moderna de roteamento baseada em componentes e estrutura modular de diretórios. O *App Router* permite a separação entre componentes executados no servidor e no cliente, bem como o uso de renderização híbrida, combinando geração de páginas no servidor e no navegador conforme a necessidade. Essa abordagem contribui para melhor desempenho, organização do código e maior controle sobre o fluxo de navegação e a proteção de rotas da aplicação.

A interface foi implementada com React em conjunto com Chakra UI, priorizando clareza visual, consistência de layout e responsividade. O objetivo foi oferecer uma experiência de uso simples e intuitiva, adequada tanto para uso individual quanto colaborativo, facilitando a visualização de boards, histórias, estados de refinamento e mecanismos de priorização.

Para a camada de persistência e serviços de *backend*, foi utilizado o Supabase, plataforma que provê um banco de dados relacional baseado em PostgreSQL, além de recursos de autenticação e controle de acesso. O sistema adota *Row Level Security* (RLS) como mecanismo de segurança, garantindo que cada usuário tenha acesso apenas aos dados que lhe pertencem. Dessa forma, a aplicação opera de forma multi-tenant, permitindo que diferentes usuários utilizem a mesma infraestrutura sem acesso aos boards, histórias ou informações de outros usuários, assegurando isolamento, privacidade e integridade dos dados.

O gerenciamento de autenticação e controle de sessões é realizado por meio do Clerk, serviço especializado em autenticação de aplicações web, responsável por funcionalidades como login seguro, persistência de sessão e proteção de rotas sensíveis. Essa integração

garante que apenas usuários autenticados possam acessar e manipular os dados armazenados, reforçando os requisitos de segurança da aplicação.

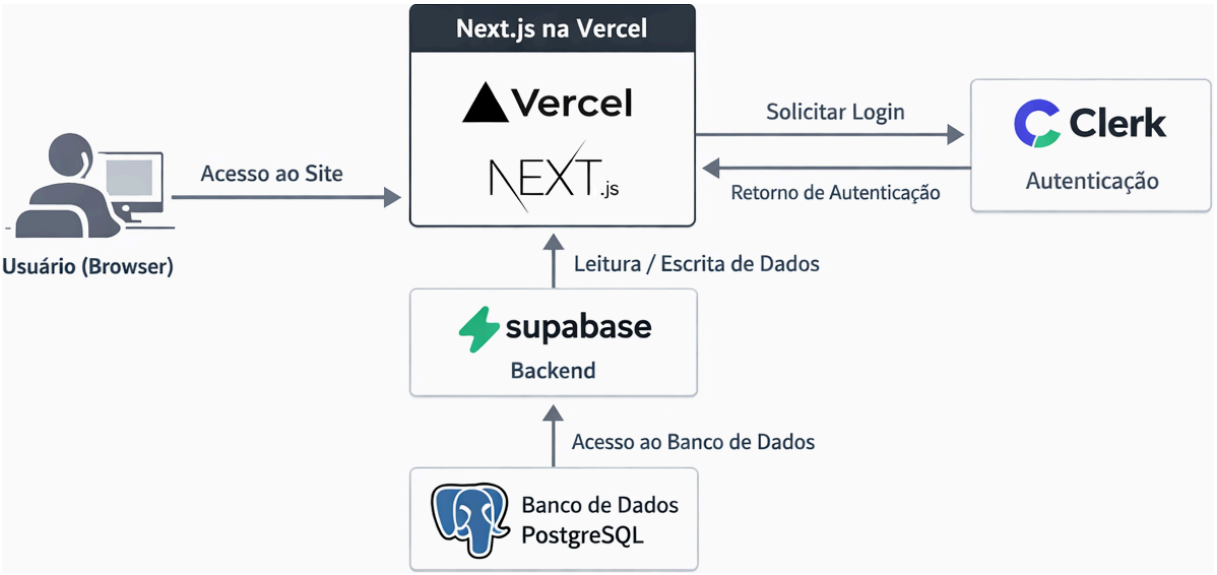
A ferramenta encontra-se disponível publicamente por meio do endereço eletrônico <https://www.refyne.com.br>. O código-fonte do projeto está armazenado em um repositório público no GitHub, disponível em <https://github.com/Alehxalves/refyne-app>, assegurando transparência, reprodutibilidade e possibilidade de evolução futura da solução. A aplicação é hospedada na plataforma Vercel, que oferece suporte nativo ao ecossistema Next.js e facilita o processo de *deploy* contínuo, atualização de versões e escalabilidade da infraestrutura.

5.2 Arquitetura da solução

A arquitetura da solução foi definida de forma a manter simplicidade estrutural, sem abrir mão da separação clara de responsabilidades entre os componentes do sistema. A aplicação é organizada em três camadas principais: interface do usuário, camada de aplicação e serviços externos.

A Figura 9 apresenta uma visão geral da arquitetura da solução, evidenciando a interação entre a aplicação web, os serviços de autenticação e o banco de dados.

Figura 9 – Visão geral da arquitetura da ferramenta Refyne



Fonte: Elaborada pelo autor

5.3 Ferramenta Refyne

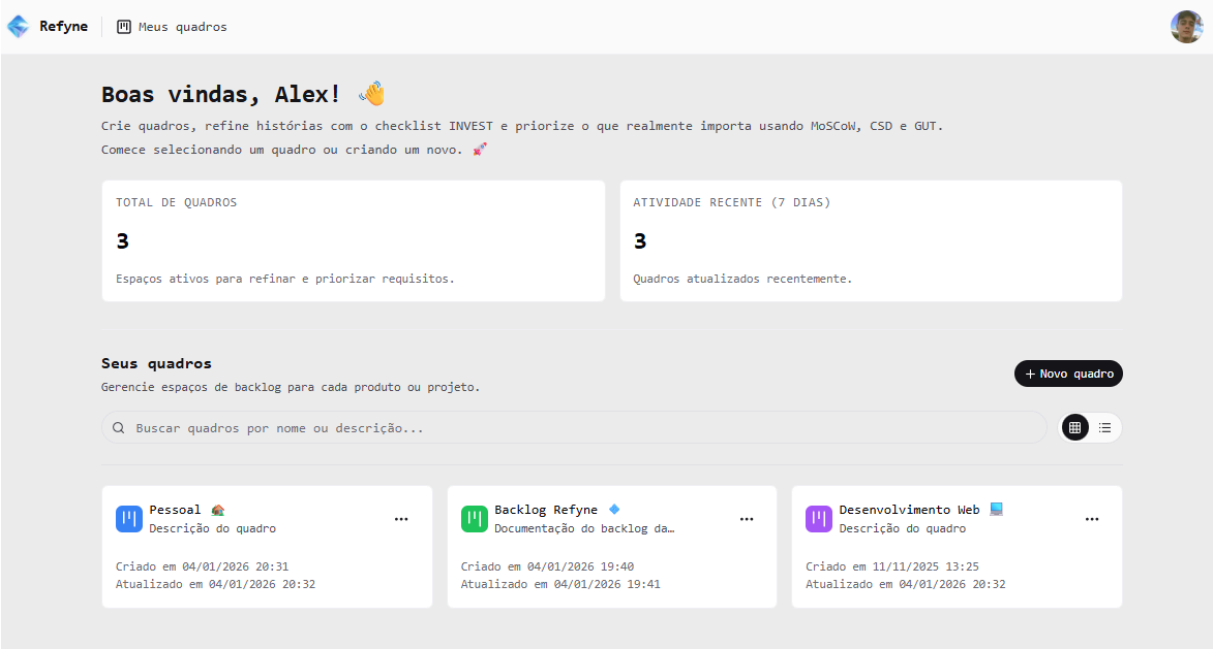
A ferramenta materializa os conceitos discutidos ao longo deste trabalho por meio de funcionalidades que apoiam diretamente o refinamento e a priorização de histórias de usuário. Ao acessar a aplicação, o usuário é apresentado a uma interface inicial que descreve a proposta da ferramenta e oferece acesso ao fluxo de autenticação. A Figura 11 apresenta a página inicial da ferramenta.

Figura 10 – Página inicial da ferramenta Refyne

Fonte: Elaborada pelo autor

Após o login, o usuário tem acesso à área principal da aplicação, na qual pode criar e gerenciar quadros. Cada quadro representa um contexto de trabalho, como um projeto ou produto, sendo caracterizado por um título, uma descrição e uma cor identificadora. Essa organização permite ao *Product Owner* separar diferentes iniciativas sem interferência entre seus respectivos *backlogs*. A Figura 12 ilustra essa visualização.

Figura 11 – Meus quadros



Fonte: Elaborada pelo autor

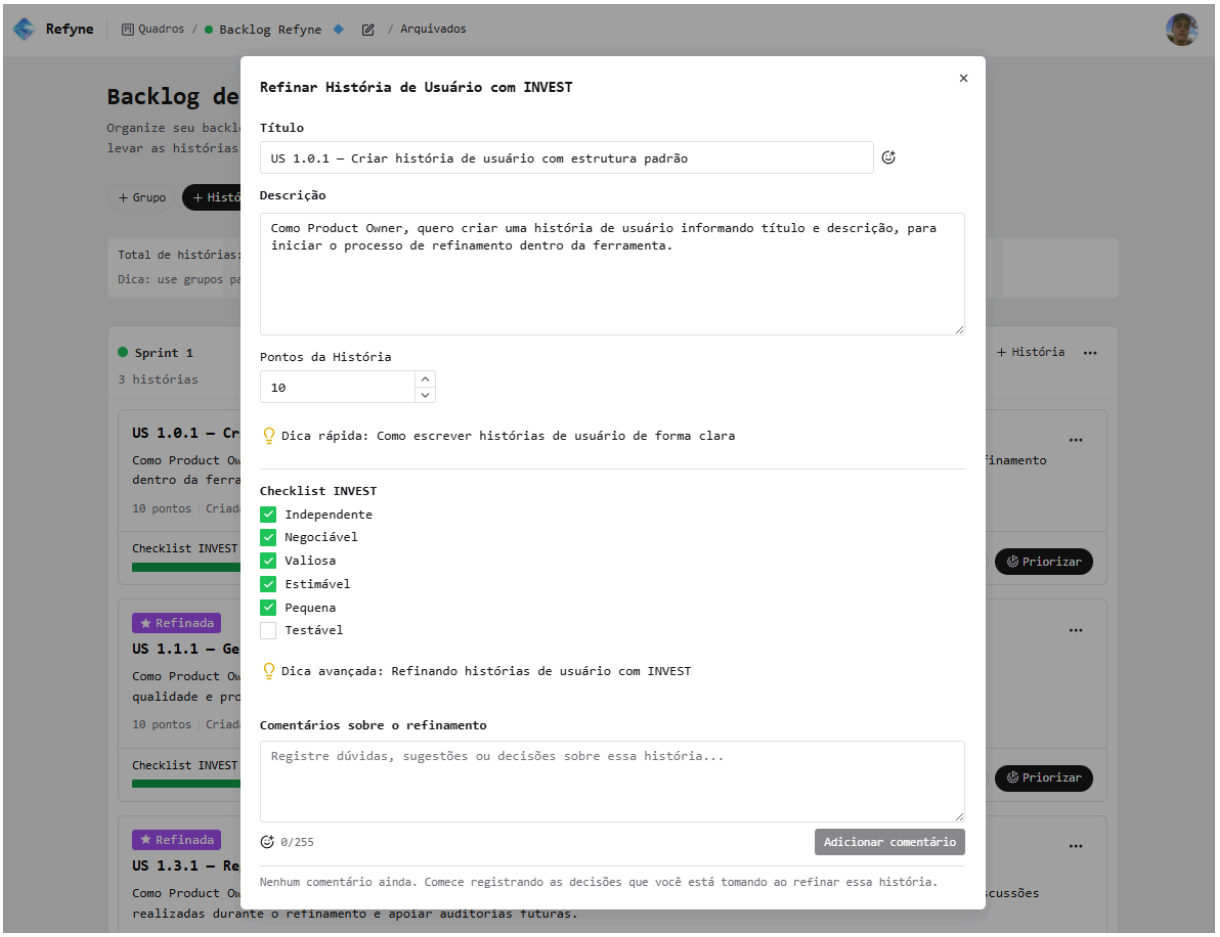
Dentro de cada quadro, o usuário pode criar histórias de usuário diretamente ou organizá-las em grupos definidos livremente. Histórias também podem existir sem associação a grupos específicos, permanecendo em uma área padrão destinada a esse fim. Essa flexibilidade permite que o refinamento ocorra de forma incremental, sem impor uma estrutura rígida ao usuário. A Figura 13 ilustra a visualização de um quadro com seus respectivos grupos e histórias.

Figura 12 – Visualização do quadro com grupos e histórias

Fonte: Elaborada pelo autor

Durante o refinamento, cada história possui um checklist INVEST associado automaticamente no momento de sua criação. O *Product Owner* pode marcar ou desmarcar os critérios do checklist conforme a história evolui, e o estado de prontidão é atualizado visualmente, indicando quando a história é considerada refinada. Esse mecanismo torna explícito um processo que, em muitos contextos, ocorre de forma implícita. A Figura 14 apresenta o processo de refinamento de uma história por meio do checklist INVEST.

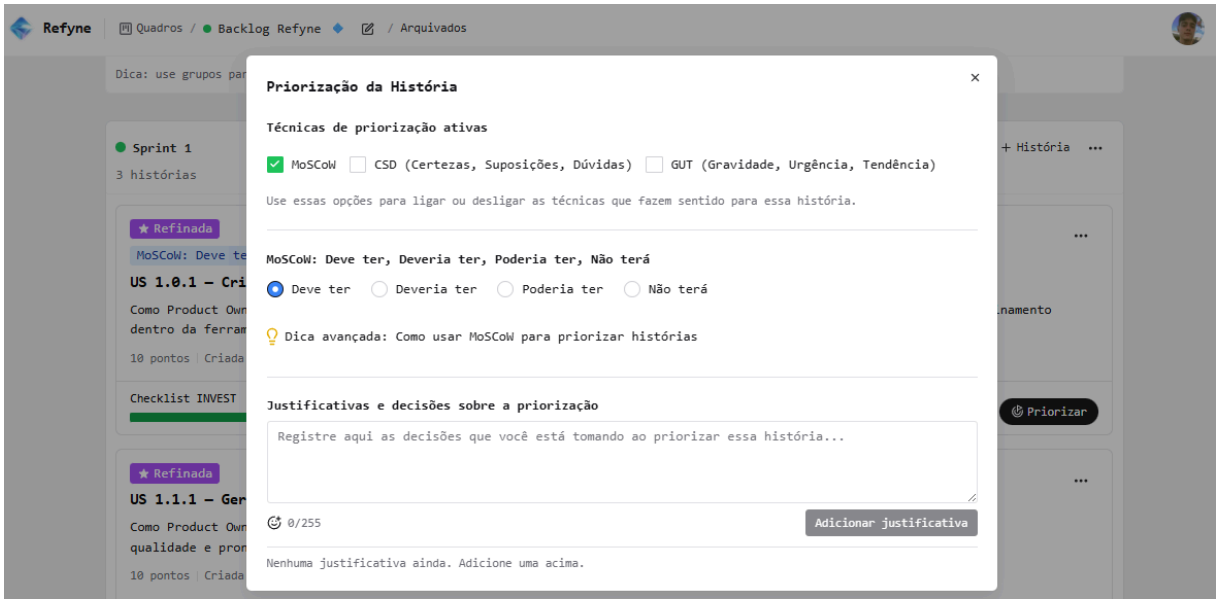
Figura 13 – Refinamento de história com checklist INVEST



Fonte: Elaborada pelo autor

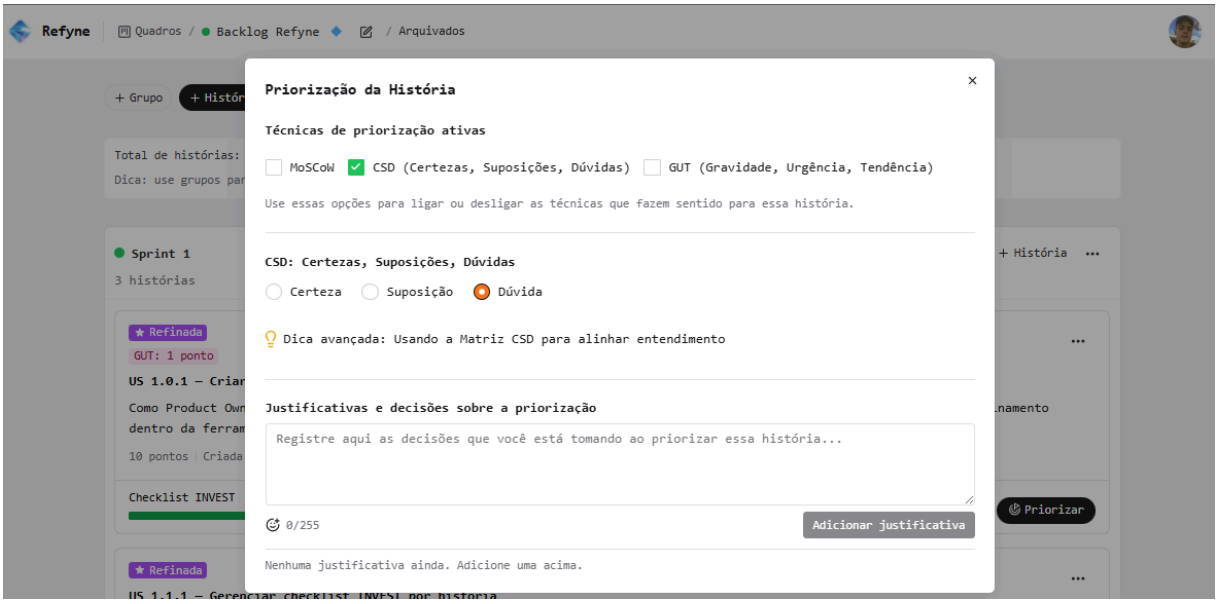
A priorização do *backlog* é apoiada por técnicas leves, como MoSCoW, CSD e GUT, permitindo que o *Product Owner* organize as histórias de acordo com critérios definidos As Figuras 15, 16 e 17 ilustram a aplicação dessas funcionalidades de priorização.

Figura 14 – Priorização das histórias com o método MoSCoW



Fonte: Elaborada pelo autor

Figura 15 – Priorização das histórias com o método CSD



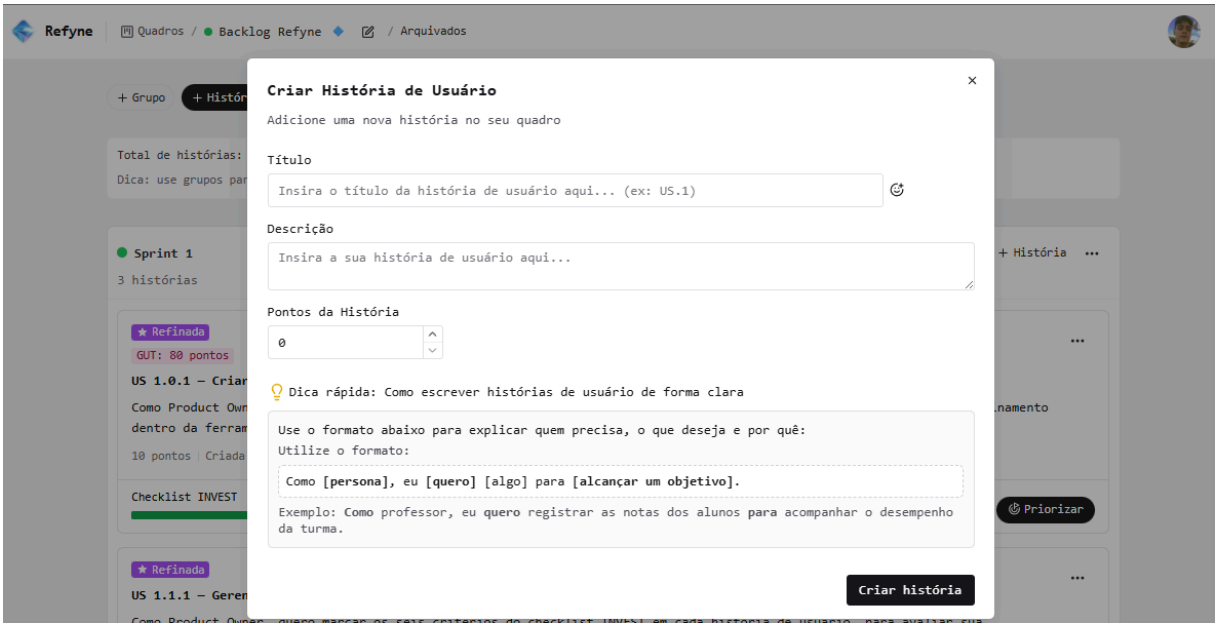
Fonte: Elaborada pelo autor

Figura 16 – Priorização das histórias com o método GUT

Fonte: Elaborada pelo autor

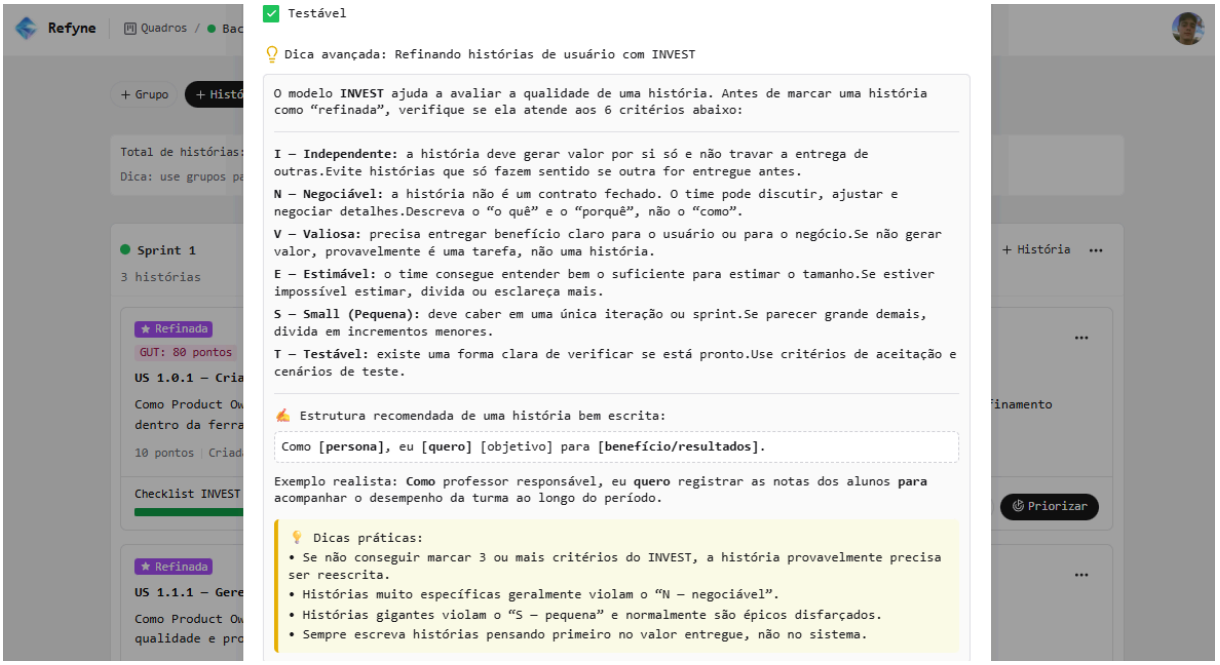
Além das funcionalidades de criação, refinamento e priorização, o Refyne incorpora dicas contextuais diretamente na interface, com o objetivo de orientar o usuário na aplicação correta das práticas abordadas neste trabalho. Durante a escrita de histórias, a ferramenta apresenta o formato recomendado “Como [persona], eu quero [objetivo] para [benefício]”, com exemplos ilustrativos. No refinamento, são oferecidas orientações práticas sobre o modelo INVEST, auxiliando na avaliação da qualidade das histórias. De forma complementar, o sistema disponibiliza explicações e boas práticas para os métodos de priorização MoSCoW, CSD e GUT. Dessa forma, o Refyne aproxima os conceitos teóricos da Engenharia de Requisitos Ágil da prática cotidiana do *Product Owner*. As Figuras 18, 19, 20, 21 e 22 ilustram exemplos dessas dicas integradas à interface da ferramenta.

Figura 17 – Dica contextual para escrita de histórias de usuário no Refyne.



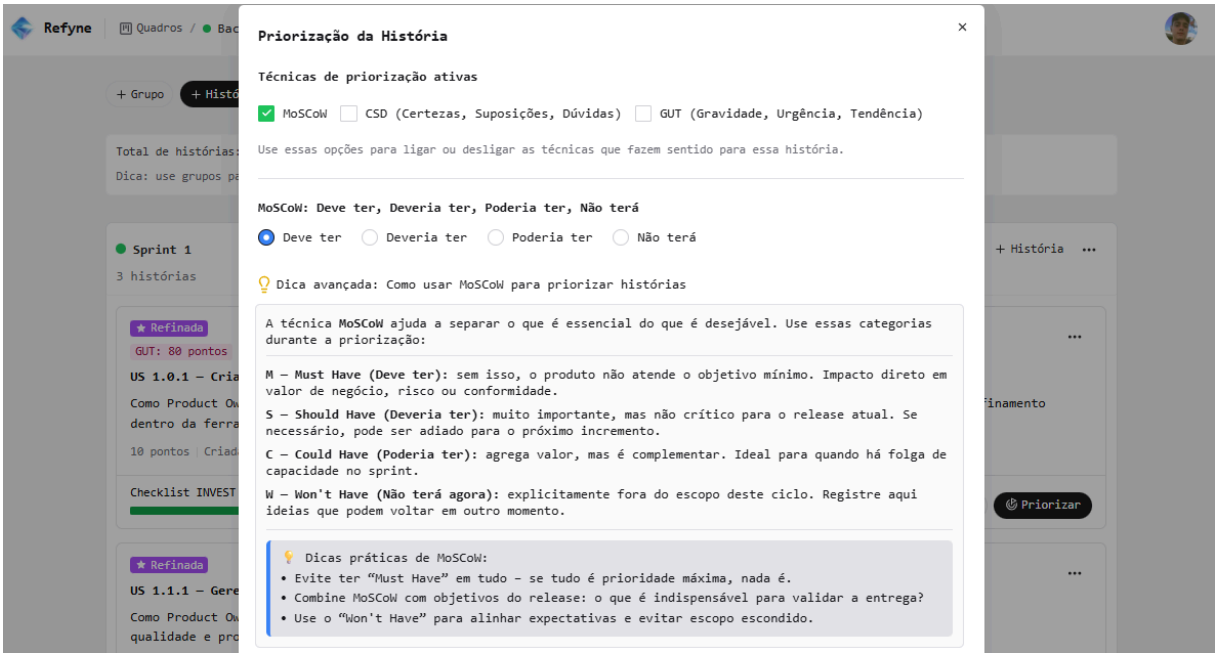
Fonte: Elaborada pelo autor

Figura 18 – Orientações do modelo INVEST integradas ao processo de refinamento.



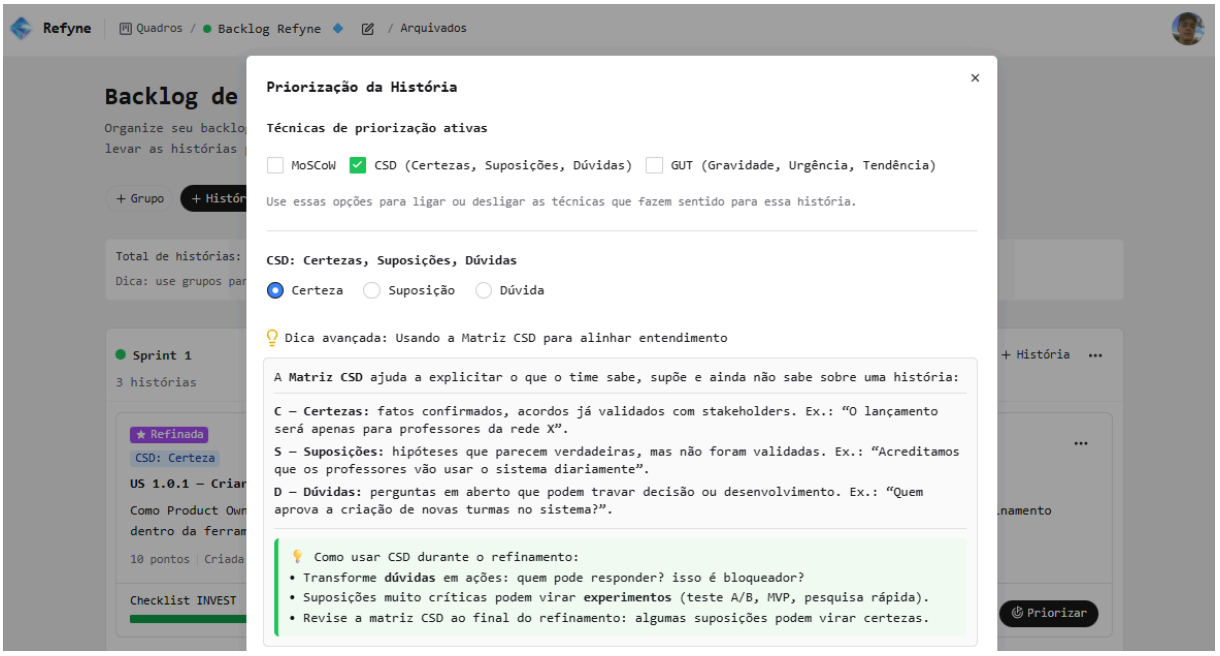
Fonte: Elaborada pelo autor

Figura 19 – Dica de uso da técnica MoSCoW para priorização de histórias no Refyne.



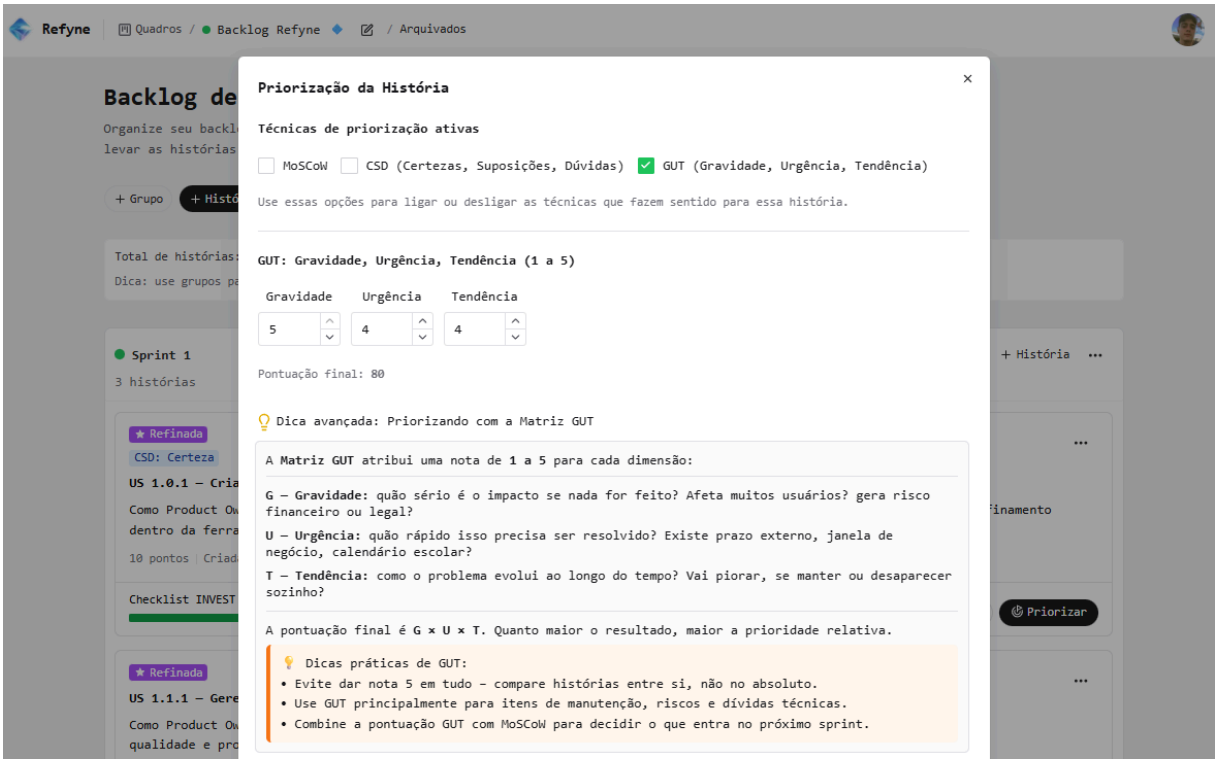
Fonte: Elaborada pelo autor

Figura 20 – Exemplo de orientação para aplicação da Matriz CSD durante o refinamento.



Fonte: Elaborada pelo autor

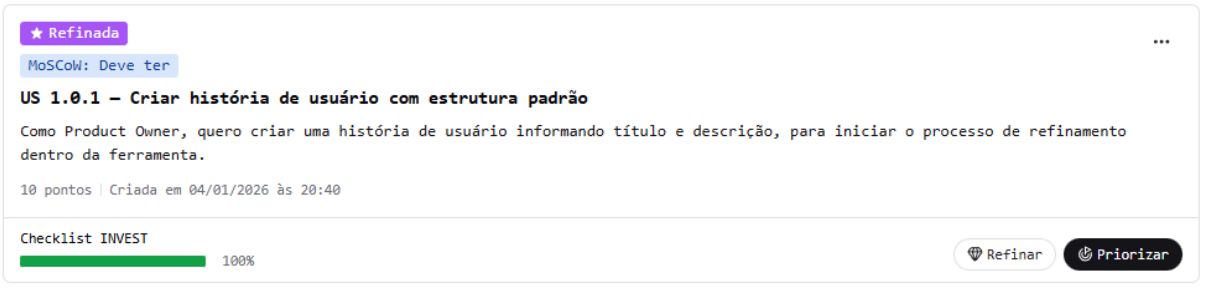
Figura 21 – Dica de priorização por meio da Matriz GUT na ferramenta Refyne.



Fonte: Elaborada pelo autor

Como resultado da integração entre os mecanismos de refinamento e priorização, a ferramenta permite visualizar de forma explícita o estado consolidado de cada história de usuário. Histórias refinadas apresentam um indicativo visual do preenchimento completo do checklist INVEST, enquanto a prioridade definida é exibida por meio de *tags* associadas à técnica escolhida. A Figura 23 ilustra uma história de usuário após o processo de refinamento e priorização, evidenciando simultaneamente seu nível de prontidão e sua classificação no *backlog*.

Figura 22 – História de usuário refinada e priorizada na ferramenta Refyne



Fonte: Elaborada pelo autor

5.4 Considerações sobre os resultados obtidos

A construção da ferramenta Refyne demonstra a viabilidade de operacionalizar conceitos da Engenharia de Requisitos Ágil em uma solução prática, alinhada às necessidades identificadas junto a *Product Owners*. Embora nem todas as funcionalidades inicialmente elicitadas tenham sido implementadas, o conjunto entregue atende aos objetivos centrais do trabalho, oferecendo suporte efetivo ao refinamento, à priorização e à observação do processo.

Entre as funcionalidades previstas e não contempladas nesta etapa, destacam-se: (i) a identificação automática e sinalização de histórias consideradas grandes, com base em um limite configurável de tamanho, que auxiliaria o *Product Owner* na decisão de dividir histórias durante o refinamento; e (ii) a visualização das histórias em uma matriz de Valor × Esforço, destinada a apoiar discussões de priorização com *stakeholders* por meio de uma representação gráfica dos itens do *backlog*. Embora a ferramenta permita atualmente a atribuição de *story points* e a ordenação das histórias com base nesse critério, não foi implementado um mecanismo automático de detecção de histórias excessivamente grandes nem a representação visual em quadrantes de valor e esforço.

Ainda assim, os resultados apresentados neste capítulo evidenciam que a proposta vai além de uma especificação teórica, materializando-se em um artefato funcional que pode ser utilizado e evoluído em contextos reais. As limitações observadas, bem como as possibilidades de ampliação da ferramenta, são discutidas no próximo capítulo, dedicado às conclusões e aos trabalhos futuros.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo desenvolver uma ferramenta para apoio ao refinamento e à priorização de histórias de usuário, a partir das dificuldades enfrentadas por *Product Owners* em contextos ágeis, identificadas e analisadas no TCC 1. Enquanto a primeira etapa do trabalho concentrou-se na compreensão do problema, o TCC 2 teve como foco a materialização de uma solução prática, buscando alinhar conceitos da Engenharia de Requisitos ágil com uma implementação funcional.

A partir dos achados empíricos obtidos por meio de questionários e entrevistas, foi possível identificar lacunas recorrentes relacionadas à qualidade das histórias de usuário, à aplicação inconsistente de critérios de prontidão e à falta de transparência no processo de priorização. Esses elementos fundamentaram a elicitação dos requisitos da ferramenta e orientaram a definição de uma abordagem que privilegiasse a simplicidade, a visualização e a sistematização do processo de refinamento, sem impor rigidez excessiva ao fluxo de trabalho dos *Product Owners*.

Como principal resultado, foi desenvolvida a ferramenta Refyne, uma aplicação web que operacionaliza práticas como o checklist INVEST, a sinalização explícita de prontidão das histórias e a priorização guiada por técnicas leves, como MoSCoW, CSD e GUT. A ferramenta torna visíveis estados e decisões que, em muitos contextos, permanecem implícitos ou dependentes exclusivamente da experiência individual, contribuindo para maior clareza, rastreabilidade e organização do *backlog*. A integração entre refinamento, priorização e apoio visual à decisão mostrou-se viável dentro do escopo proposto, resultando em um MVP funcional e publicamente acessível.

Do ponto de vista metodológico, o trabalho evidencia que conceitos discutidos na literatura podem ser traduzidos em mecanismos práticos de apoio ao trabalho do *Product Owner*, sem a necessidade de ferramentas excessivamente complexas. A Refyne não se propõe a substituir soluções consolidadas de gestão ágil, mas a atuar de forma complementar, focando especificamente nas etapas de refinamento e priorização, que frequentemente recebem menor suporte nas ferramentas tradicionais.

Apesar dos resultados alcançados, este trabalho apresenta limitações. A avaliação da ferramenta ocorreu de forma exploratória e com um número reduzido de participantes, não sendo possível realizar análises quantitativas de impacto ou estudos formais de usabilidade. Além disso, nem todas as funcionalidades inicialmente elicitadas foram implementadas, em função das restrições de tempo e escopo inerentes ao contexto acadêmico, o que limitou a

abrangência da solução entregue.

Como trabalhos futuros, destaca-se a ampliação da avaliação da ferramenta, envolvendo um número maior de *Product Owners* e a aplicação de métodos específicos de análise de usabilidade, aceitação e impacto no processo de refinamento. Também se apresenta como evolução natural da proposta a incorporação do suporte à Definição de Concluído (do inglês *Definition of Done* - DoD), permitindo a definição e o acompanhamento de critérios compartilhados pelo time para considerar uma história como concluída. Essa funcionalidade atuaria como complemento ao uso da Definição de Pronto e do checklist INVEST, estendendo o apoio da ferramenta para etapas posteriores ao desenvolvimento e promovendo maior alinhamento entre refinamento, priorização e conclusão das histórias. A implementação do DoD possibilitaria, ainda, associar critérios de aceite individuais das histórias a uma definição comum de conclusão, fortalecendo a previsibilidade e a consistência das entregas.

Outras possibilidades de evolução incluem o aprimoramento das métricas relacionadas ao processo de refinamento, com visualizações mais detalhadas e análises históricas que permitam acompanhar a maturidade do processo ao longo do tempo, bem como a integração da ferramenta com soluções de gestão ágil já utilizadas pelas equipes. Essas extensões podem ampliar o uso da Refyne em ambientes reais de trabalho, sem descaracterizar seu foco original.

Por fim, espera-se que este trabalho contribua para a discussão sobre a importância do refinamento contínuo e estruturado de histórias de usuário, demonstrando que a combinação entre fundamentos teóricos e soluções práticas pode auxiliar *Product Owners* a lidar de forma mais consciente e sistemática com a complexidade do *backlog*. A ferramenta desenvolvida permanece disponível para uso e evolução, abrindo espaço para investigações futuras e para sua aplicação em diferentes contextos organizacionais.

REFERÊNCIAS

AGUIAR, F.; CAROLI, P. **Product Backlog Building: um guia prático para criação e refinamento de backlog para produtos de sucesso**. 1.ed. Rio de Janeiro: Editora Caroli, 2021. 168 p. ISBN-13 978-6586660111.

ALMEIDA, Maria Luiza Ferreira Assumpção; QUEIROZ, Natália Maria Rodrigues. **Backlog maker: uma ferramenta de apoio à construção e à manutenção do backlog do produto**. 2021. 177 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Software) – Universidade de Brasília, Brasília, 2021. Disponível em: https://bdm.unb.br/handle/10483/30734. Acesso em: 11 jun. 2025.

ALVES, Daniel da Silva. **Estudo de caso: o uso de user stories aplicado na construção de um backlog de produto**. 2022. 177 f. Trabalho de Conclusão de Curso (Bacharelado em Tecnologia da Informação) – Universidade Federal Rural do Semi-Árido, Pau dos Ferros, 2022. Disponível em: https://repositorio.ufersa.edu.br/items/c9b46d09-9343-4d2f-8875-19391fc47b57. Acesso em: 11 jun. 2025.

BECK, K. et al. **Manifesto for Agile Software Development**. 2001. Disponível em: https://agilemanifesto.org/. Acesso em: 1 jun. 2025.

BOEHM, Barry W. A spiral model of software development and enhancement. **Computer**, v. 21, n. 5, p. 61–72, maio 1988. DOI: 10.1109/2.59.

BRETAS, A. **Matriz de Certezas, Suposições e Dúvidas**. Medium, 2015. Disponível em: https://medium.com/educacao-fora-da-caixa/matriz-certezas-suposi%C3%A7%C3%B5es-e-d%C3%BAvidas-fa2263633655/. Acesso em: 22 jun. 2025.

CADLE, J.; YEATES, D. **Project Management for Information Systems**. 5. ed. Harlow: Pearson Education, 2007. 464 p.

CAMARGO, R. F. **Como fazer a Matriz GUT para a resolução de problemas? Conheça a Matriz de Prioridades**. 2018. Disponível em: https://www.treasy.com.br/blog/matriz-gut/. Acesso em: 11 jun. 2025.

CAROLI, P.; AGUIAR, F. **Lean Inception: como alinhar pessoas e construir o produto certo**. São Paulo: Caroli.org, 2018. 160 p.

CEVADA, Luana Zanini; DAMY-BENEDETTI, Patricia de Carvalho. USO DA MATRIZ DE PRIORIZAÇÃO (MATRIZ GUT) COMO ALIADA EM AUDITORIAS. **Revista Científica Unilago**, [S. l.], v. 1, n. 1, 2022. Disponível em: https://revistas.unilago.edu.br/index.php/revista-cientifica/article/view/591. Acesso em: 11 jul. 2025.

CHRISTIE, Jim. **The Scrum Framework**. Jim Christie Blog, 2020. Disponível em: https://jimchristie.me/blog/scrum-graphic/. Acesso em: 16 jul. 2025.

COHN, Mike. **User stories applied: for agile software development**. Boston: Addison-Wesley, 2004. 268 p.

DIGITAL.AI. **17th State of Agile Report**. Raleigh, NC, 2023. Disponível em: https://digital.ai/resource-center/analyst-reports/state-of-agile-report/. Acesso em: 4 jun. 2025.

EQUIPE DE CONTEÚDO – PM3. **Matriz CSD: o que é e como construir junto ao time de Produto**. 21 dez. 2023. Disponível em: https://pm3.com.br/blog/matriz-csd-o-que-e/. Acesso em: 21 jul. 2025.

G4 EDUCAÇÃO. **Matriz CSD**. 2023. Disponível em: https://g4educacao.com/blog/matriz-csd/. Acesso em: 11 jul. 2025.

HEUER, M.; KREMER, N.; KURTZ, C. Clarifying the role of product owners: a comprehensive taxonomy on product owner roles. In: HCI IN BUSINESS, GOVERNMENT AND ORGANIZATIONS, 2025, Cham. **Proceedings**. Cham: Springer, 2025. p. 52–66. (Lecture Notes in Computer Science, v. 14396).

MEDEIROS DE ASSIS, Daniel; LARIEIRA, Cláudio Luis Carvalho; COSTA, Ivanir. As Dificuldades na Adoção e Uso de Método Scrum em Empresas Brasileiras Utilizando Processos Plan-Driven: Estudo de Caso Múltiplo. **Revista de Gestão e Projetos**, São Paulo, v. 8, n. 3, p. 66–79, 2017. DOI: 10.5585/gep.v8i3.544. Disponível em: https://periodicos.uninove.br/gep/article/view/9676. Acesso em: 19 jun. 2025.

OLIVEIRA, Ronielton. **PRINCE2: A Técnica de Priorização MoSCoW**. 2014. DOI: 10.13140/RG.2.2.26072.49925. Disponível em: https://www.researchgate.net/publication/317539229_PRINCE2_A_Tecnica_de_Priorizacao_MoSCoW. Acesso em: 11 jul. 2025.

PARABOL. **300+ Agile & Scrum Statistics**. Chicago, 2024. Disponível em: https://www.parabol.co/resources/agile-statistics/. Acesso em: 2 jun. 2025.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de software: uma abordagem profissional**. 8. ed. Porto Alegre: AMGH, 2016. 784 p.

RIESEN, Mark; WAGENAAR, Gerard. **What is Ready in a DoR? Rationales, Responsibility & Rules in using a Definition of Ready**. 2024. Disponível em: https://www.researchgate.net/publication/381164049_What_is_Ready_in_a_DoR_Rationales_Responsibility_Rules_in_using_a_Definition_of_Ready. Acesso em: 5 jun. 2025.

RUBIN, Kenneth S. **Essential scrum: a practical guide to the most popular agile process**. Boston: Addison-Wesley, 2012. 496 p.

SANTOS, L. T. et al. Ferramentas e metodologias para a gestão inovadora em unidades de informação. In: ENCONTRO INTERDISCIPLINARIDADES: Arquivologia, Biblioteconomia e Museologia, 2., 2018, Salvador. **Anais [...]**. Salvador: UFBA, 2018.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide: the definitive guide to Scrum**. 2020. Disponível em: https://scrumguides.org/scrum-guide.html. Acesso em: 20 jun. 2025.

SILVA, B. W. F. V.; OLIVEIRA, S. R. B.; LIMA, A. F. A.; PINHEIRO, A. L. C. REACT-M: uma abordagem ágil para o gerenciamento de requisitos de software. In: COMPUTER ON THE BEACH (COTB), 11., 2020. **Anais [...]**. Disponível em: https://periodicos.univali.br/index.php/acotb/article/view/16819. Acesso em: 1 jan. 2026.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson Education do Brasil, 2011. 792 p.

SOUZA, Maria Regina. **Análise e aprimoramento de requisitos para desenvolvimento de software: um estudo de user stories na perspectiva dos desenvolvedores**. 2024. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Universidade Federal do Amazonas, Manaus, 2024. Disponível em: https://riu.ufam.edu.br/bitstream/prefix/8232/4/TCC_MariaReginaSouza.pdf. Acesso em: 3 jan. 2026.

TAKEUCHI, H.; NONAKA, I. The new product development game. **Harvard Business Review**, v. 64, n. 1, p. 137–146, 1986.

UDS TECNOLOGIA. **Ciclo de vida do software: entenda os 7 modelos principais**. 2023. Disponível em: https://uds.com.br/blog/ciclo-de-vida-do-software-web/. Acesso em: 13 jul. 2025.

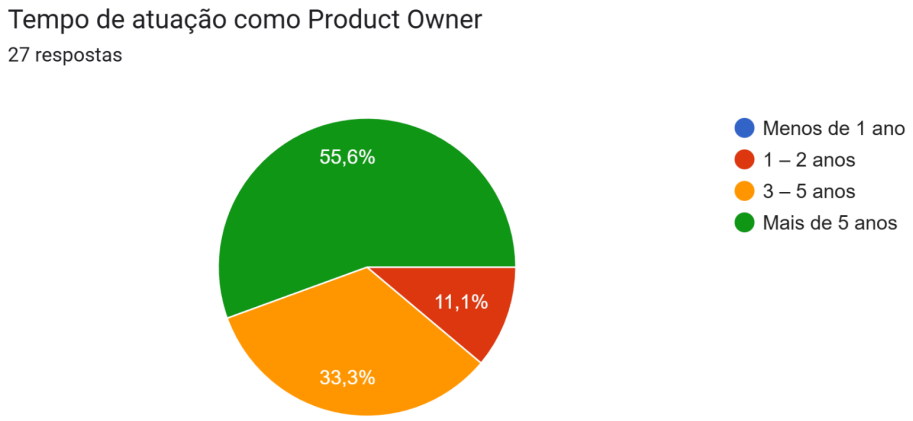
VAN DER MEER, Jeroen. **Definition of Ready in Practice**. 2022. Tese (Doutorado) – Utrecht University, Utrecht, 2022. Disponível em: https://studenttheses.uu.nl/bitstream/handle/20.500.12932/46489/Final_Thesis.pdf?sequence=1&isAllowed=y. Acesso em: 5 jan. 2026.

VARELLA, Guilherme; SANTOS, Marcos; GOMES, Carlos Francisco; CASSETTARI, Eder. Proposta de priorização de requisitos de software utilizando a matriz MoSCoW e o método multicritério SAPEVO-M. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO – ENEGEP, 41., 2021, Foz do Iguaçu. **Anais [...]**. Rio de Janeiro: ABEPRO, 2021. DOI: 10.14488/ENEGEP2021_TN_STO_356_1836_41806. Disponível em: https://www.researchgate.net/publication/355362215_Proposta_de_priorizacao_de_requisitos_de_software_utilizando_a_matriz_MoSCoW_e_o_metodo_multicriterio_SAPEVO-M. Acesso em: 11 jul. 2025.

WAKE, B. **INVEST in Good Stories, and SMART Tasks**. 2003. Disponível em: https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/. Acesso em: 22 jun. 2025.

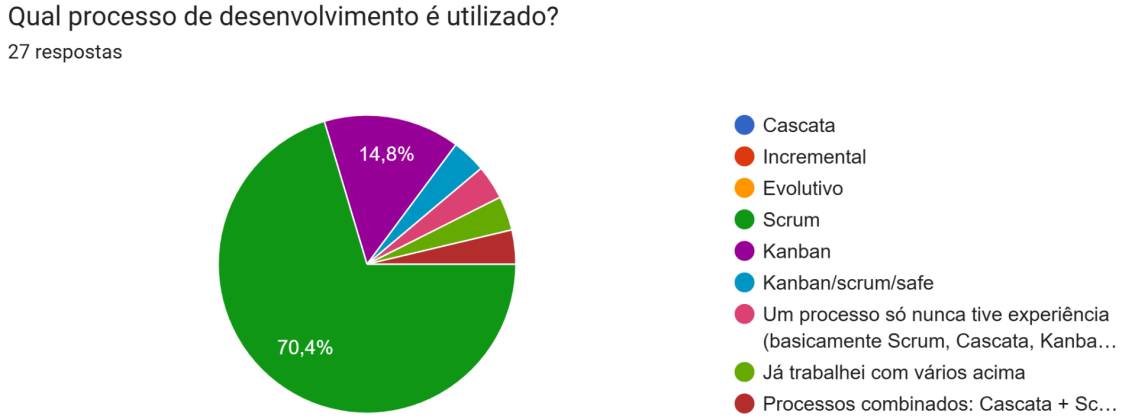
APÊNDICE A – RESULTADOS DO QUESTIONÁRIO ONLINE

Figura A.1 – Perfil de experiência dos *Product Owners* participantes



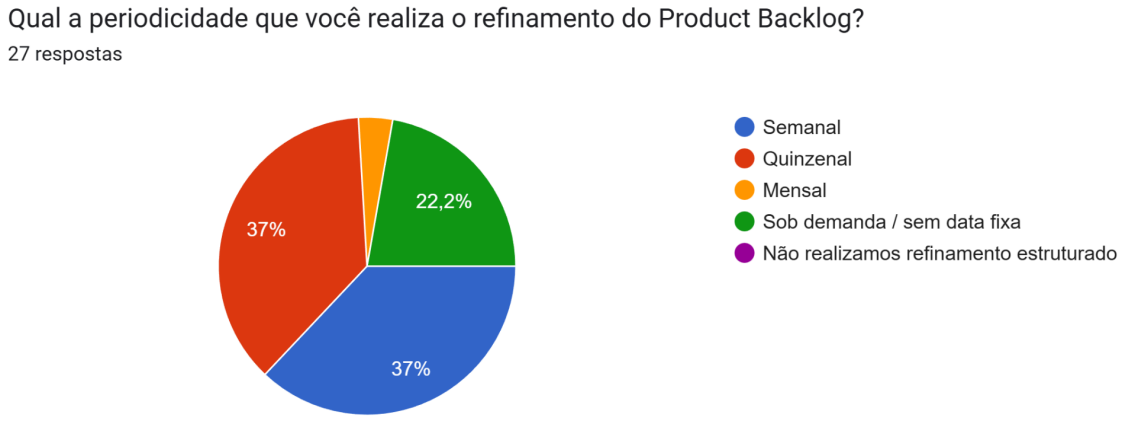
Fonte: Dados da pesquisa (2025)

Figura A.2 – Frameworks de desenvolvimento utilizados pelos participantes



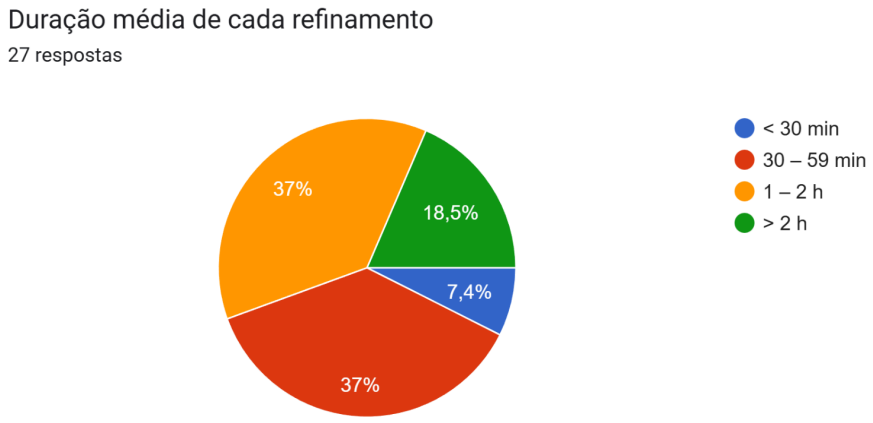
Fonte: Dados da pesquisa (2025)

Figura A.3 – Periodicidade das sessões de refinamento do backlog



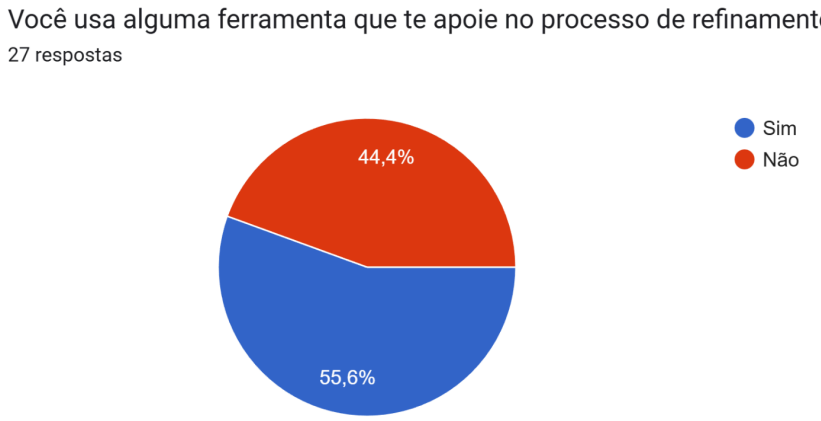
Fonte: Dados da pesquisa (2025)

Figura A.4 – Duração média das sessões de refinamento do backlog



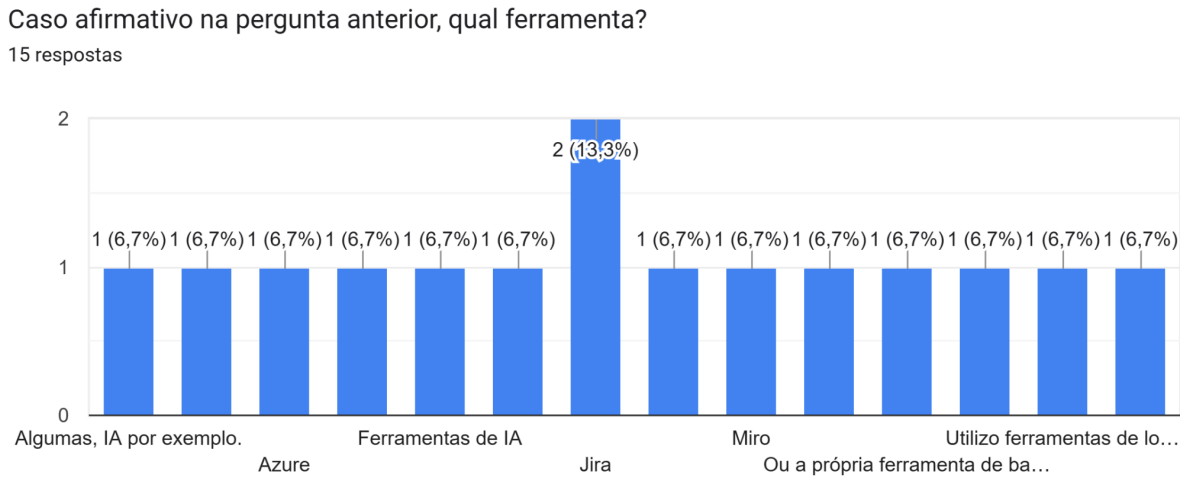
Fonte: Dados da pesquisa (2025)

Figura A.5 – Ferramentas de apoio ao refinamento utilizadas



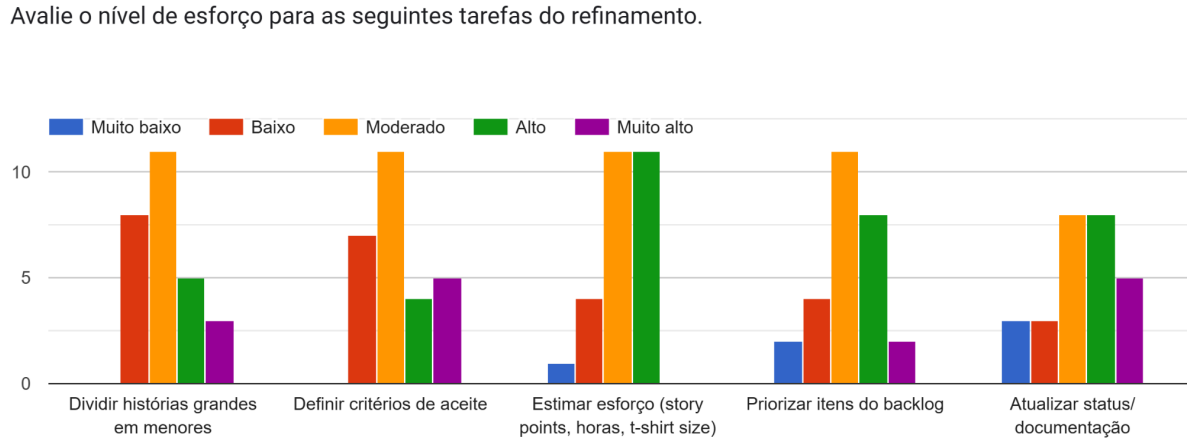
Fonte: Dados da pesquisa (2025)

Figura A.6 – Ferramentas de apoio



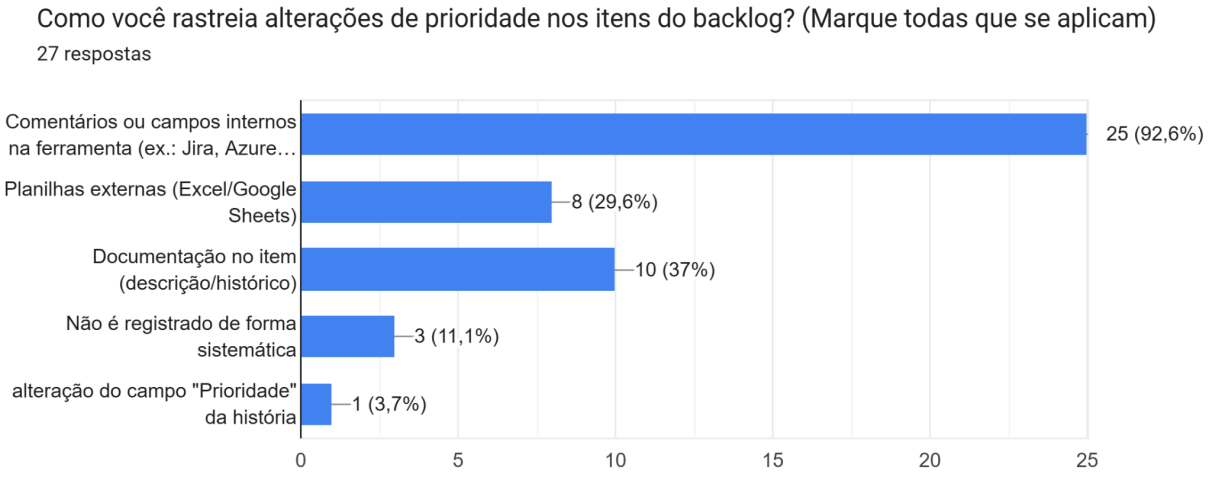
Fonte: Dados da pesquisa (2025)

Figura A.7 – Nível de esforço percebido nas principais atividades de refinamento



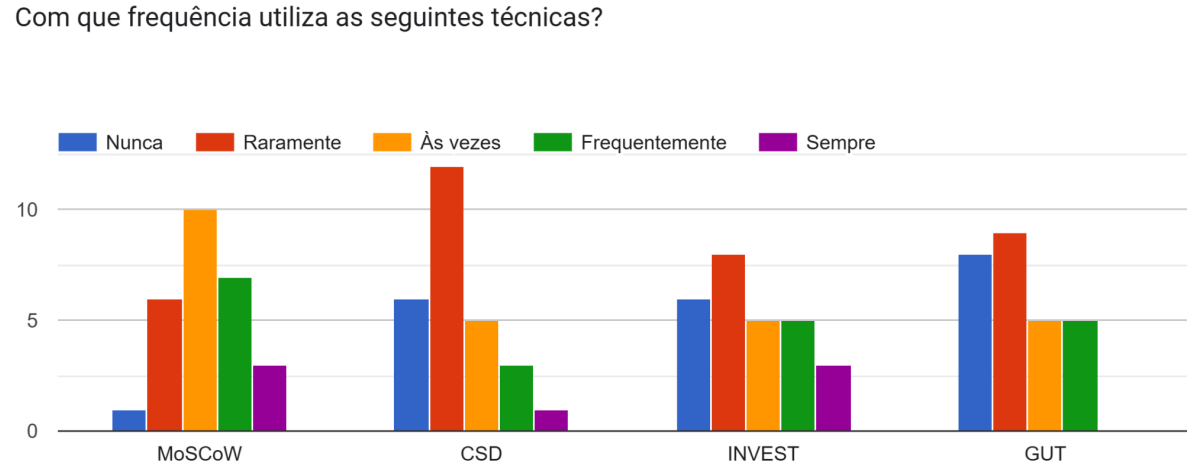
Fonte: Dados da pesquisa (2025)

Figura A.8 – Técnicas de priorização e qualidade



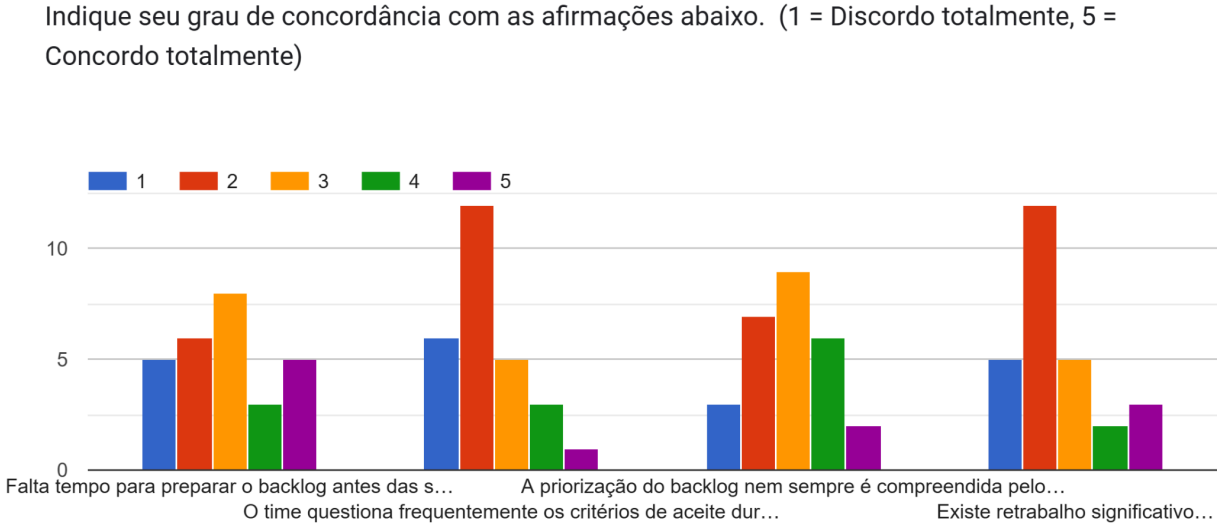
Fonte: Dados da pesquisa (2025)

Figura A.9 – Frequência de uso de técnicas estruturadas (MoSCoW, CSD, INVEST, GUT)



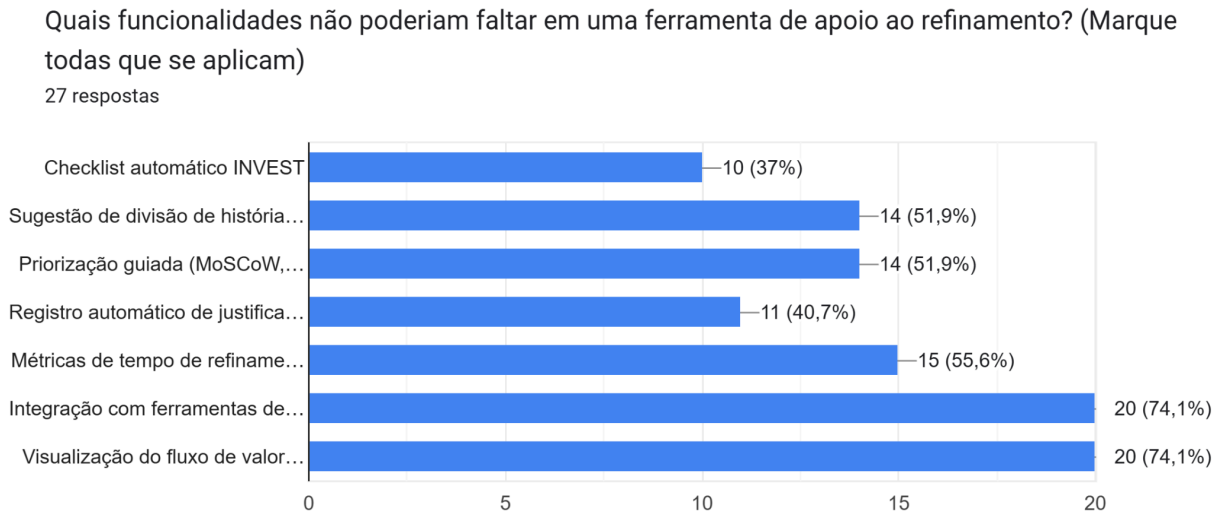
Fonte: Dados da pesquisa (2025)

Figura A.10 – Principais dores mapeadas no processo de refinamento



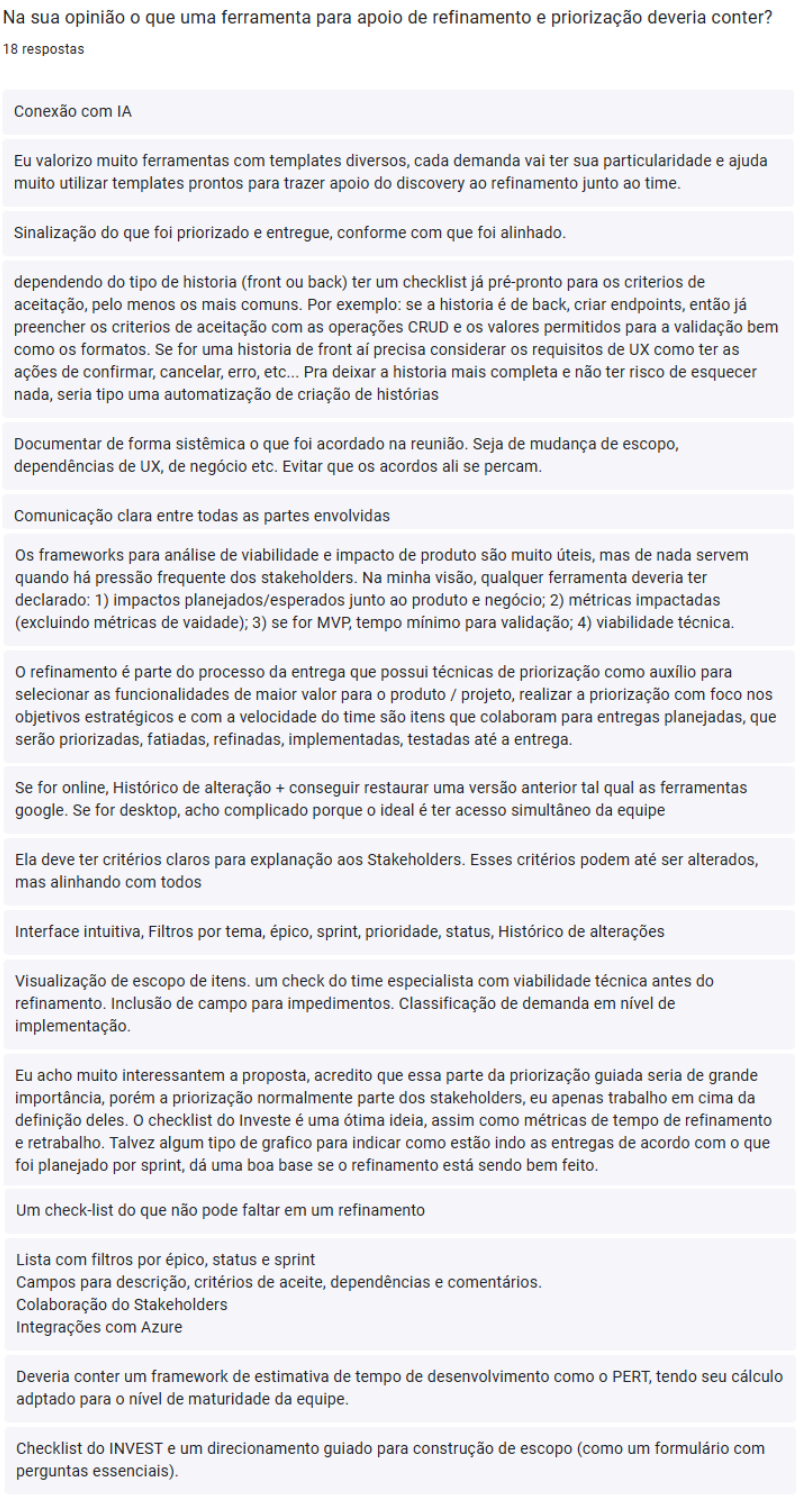
Fonte: Dados da pesquisa (2025)

Figura A.11 – Funcionalidades de apoio desejadas em uma ferramenta de refinamento



Fonte: Dados da pesquisa (2025)

Figura A.12 – Feedback sobre o que a ferramenta deveria conter



Fonte: Dados da pesquisa (2025)

APÊNDICE B – TRANSCRIÇÕES DAS ENTREVISTAS

Para preservar a identidade dos participantes e atender aos princípios éticos da pesquisa, os entrevistados são identificados por códigos (E1, E2 e E3), não sendo divulgados seus nomes ou dados pessoais.

B.1 Entrevista com a participante E1

Entrevistador: Há quanto tempo você atua como *Product Owner*?

Entrevistada: Três anos e dez meses, quase quatro anos.

Entrevistador: Com que periodicidade ocorrem as sessões de refinamento, semanal, quinzenal ou mensal?

Entrevistada: Depende. Geralmente quinzenal, se a sprint for quinzenal. Não tem uma resposta fixa, é vinculada à duração da *sprint*.

Entrevistador: Como vocês registram o porquê de um item do *backlog* subir ou descer a prioridade?

Entrevistada: Comentário. Existe o campo de prioridade na ferramenta (Jira), mas a justificativa geralmente vai no comentário.

Entrevistador: Quais os maiores obstáculos ou ineficiências que você percebe no processo de refinamento?

Entrevistada: Falta de foco. Você precisa definir todos os critérios de aceitação durante a sessão, e às vezes passa batido algum "*corner case*", que é difícil de prever, mas pode acontecer.

Entrevistador: Você aplica MoSCoW, GUT, CSD, INVEST ou outras técnicas?

Entrevistada: Aplico o MoSCoW. Não conheço a GUT e a CSD. O INVEST eu conheço. Uso bastante o MoSCoW, inclusive configurei o Jira para isso.

Entrevistador: Essas técnicas são feitas manualmente ou com suporte de software?

Entrevistada: Tudo dentro do Jira. A gente cria subtarefas para cada critério de aceitação. Quando todas estão fechadas, a história está pronta.

Entrevistador: Vocês usam post-its ou quadros físicos?

Entrevistada: Não mais. Só em sessões presenciais pontuais. No dia a dia usamos ferramentas online.

Entrevistador: Se fôssemos construir sua ferramenta de apoio, o que não poderia faltar?

Entrevistada: Ter planejamento por versões, separação clara entre épicos, features e histórias, possibilidade de mover itens entre épicos com facilidade, visão estruturada do backlog e

cronograma. O Jira mistura tudo e não dá uma boa visão macro.

B.1 Entrevista com a participante E2

Entrevistador: Há quanto tempo você atua como *Product Owner*?
Entrevistada: Três anos e dez meses, quase quatro anos.
Entrevistador: Qual a periodicidade das sessões de refinamento?
Entrevistada: Acontecem de 15 em 15 dias.
Entrevistador: Como você registra o porquê de um item do backlog subir ou descer de prioridade?
Entrevistada: No contexto do Estado é um pouco bagunçado. Temos muitos sistemas e *stakeholders*. Às vezes uma urgência muda tudo, e a justificativa é só "o governador solicitou". A documentação é muito limitada.
Entrevistador: Quais são os maiores obstáculos no processo atual de refinamento?
Entrevistada: Nem todo mundo está na mesma página. Às vezes a equipe interpreta diferente do esperado pelos *stakeholders*. Isso causa retrabalho na homologação. Falta entendimento comum.
Entrevistador: Você aplica MoSCoW, CSD, GUT, INVEST?
Entrevistada: Uso MoSCoW e já utilizei INVEST em outro projeto. Acho o INVEST muito bom.
Entrevistador: Vocês usam quadros físicos ou só ferramentas digitais?
Entrevistada: Só digitais, trabalho home office. Uso planilhas e o Redmine, que é bem limitado.
Entrevistador: Em quais tarefas do refinamento você sente mais esforço?
Entrevistada: Tudo que é mais técnico me consome mais, como design ou entendimento funcional. Sinto sobrecarga por não ter apoio técnico e precisar fazer além da minha função.
Entrevistador: Se fôssemos construir uma ferramenta de apoio, o que ela deveria conter?
Entrevistada: Visualização macro do que está sendo feito, categorização por etapas, possibilidade de rastrear alterações e tempo. Um local unificado para priorização e entendimento do fluxo.

B.3 Entrevista com a participante E3

Entrevistador: Há quanto tempo você atua como Product Owner?
Entrevistada: Atuo como Product Owner há aproximadamente três anos e meio.

Entrevistador: Com qual periodicidade ocorrem as sessões de refinamento?

Entrevistada: Depende do projeto. Quando utilizo Scrum com sprints de duas semanas, realizo o refinamento quinzenalmente. Em projetos que não utilizam sprints, essa periodicidade pode variar.

Entrevistador: Como você registra o motivo de um item do backlog subir ou descer de prioridade?

Entrevistada: Na prática, muitas etapas recomendadas pelas boas práticas acabam sendo simplificadas devido à carga de trabalho. Normalmente, a priorização ocorre pela organização dos itens em sprints no Jira, onde o que está na sprint mais próxima é considerado mais prioritário. Em alguns projetos, a justificativa da prioridade não fica formalmente documentada e é alinhada verbalmente com o time. Considero importante ter essa opção de registro, principalmente para equipes menos maduras.

Entrevistador: Quais os maiores obstáculos ou ineficiências que você percebe no processo de refinamento?

Entrevistada: A principal dificuldade é a baixa colaboração do time técnico nas sessões de refinamento. Muitas vezes, a participação ocorre sem engajamento, o que torna as reuniões mais longas e menos produtivas. Acredito que o problema está mais relacionado à cultura e às pessoas do que às ferramentas ou ao processo em si.

Entrevistador: Você aplica técnicas como MoSCoW, GUT, CSD ou INVEST?

Entrevistada: Utilizo o MoSCoW de forma mais intuitiva, sem seguir rigorosamente o modelo formal. As técnicas GUT e CSD não utilizo. Quanto ao INVEST, não aplico de forma completa, pois considero que escrever cenários detalhados e cartões de conversa demanda muito tempo e conhecimento técnico. Ainda assim, procuro escrever histórias mensuráveis, evitando ambiguidades sempre que possível.

Entrevistador: Essas técnicas são feitas manualmente ou com suporte de software?

Entrevistada: Utilizo apenas ferramentas digitais, pois trabalho de forma totalmente remota.

Entrevistador: Quais ferramentas você utiliza atualmente no processo de refinamento?

Entrevistada: Utilizo o Jira e o Azure DevOps para gerenciamento do backlog. Como apoio, uso ferramentas do pacote Office, como Excel, Word e PowerPoint, para documentação complementar.

Entrevistador: Em quais tarefas do refinamento você sente maior carga cognitiva ou gasto de tempo?

Entrevistada: Principalmente em dois momentos: quando estou refinando requisitos de negócio sem ter todas as informações necessárias, o que exige contato constante com

stakeholders; e quando lido com demandas muito técnicas, nas quais dependo fortemente do time de desenvolvimento para esclarecimentos, o que aumenta o esforço e o tempo gasto.

Entrevistador: Se fosse construída uma ferramenta de apoio ao refinamento e à priorização, o que não poderia faltar?

Entrevistada: Seriam essenciais a separação clara por sessões e sprints, visualização do backlog a longo prazo, organização por colunas, funcionalidades de arrastar e soltar, edição em massa de itens, definição clara de prioridades e a possibilidade de marcar itens como “em investigação”. Também considero fundamental o histórico completo das alterações, comentários com menções a usuários e suporte direto a técnicas de priorização como o MoSCoW.

APÊNDICE C – REQUISITOS DA FERRAMENTA

C.1 Requisitos Funcionais

RF01 – Checklist INVEST

Como Product Owner Quero visualizar e atualizar um checklist INVEST por história Para garantir que ela cumpra DoR e esteja pronta para entrar na sprint

Critérios de Aceite:

- A ferramenta exibe todos os seis itens do INVEST (I, N, V, E, S, T).
- O PO pode marcar/desmarcar cada item manualmente.
- Se algum item estiver desmarcado, a história é sinalizada como "Incompleta".
- Alterações são salvas em tempo real.
- Histórico de alterações é mantido (quem marcou/desmarcou e quando).

RF02 – Sugestão de Divisão de Histórias

Como Product Owner Quero receber sugestões automáticas para dividir histórias grandes Para facilitar o planejamento e evitar histórias grandes na sprint

Critérios de Aceite:

- A ferramenta detecta histórias grandes (ex.: com estimativa acima de X story points).
- Sugere possíveis divisões com base no tipo da história (front, back, integração).
- PO pode aceitar, rejeitar ou editar sugestões.
- Ao aceitar, são criadas sub-histórias automaticamente no backlog.

RF03 – Priorização Guiada

Como Product Owner Quero priorizar backlog usando técnicas como MoSCoW, CSD e GUT Para facilitar o alinhamento com stakeholders

Critérios de Aceite:

- É possível selecionar a técnica desejada (MoSCoW, CSD, GUT).
- A ferramenta exibe interface intuitiva para atribuir valores/prioridades.
- O backlog é ordenado automaticamente conforme a técnica aplicada.
- Ranking pode ser exportado em planilha ou PDF.

RF04 – Registro de Justificativas

Como Product Owner Quero registrar automaticamente justificativas de mudança de prioridade Para manter histórico e transparência das decisões

Critérios de Aceite:

- Ao alterar a prioridade de um item, o sistema solicita uma justificativa obrigatória.
- Histórico é versionado por item do backlog.
- É possível consultar e filtrar histórico por data, autor e motivo.
- Permite exportar histórico em formato PDF/CSV.

RF05 – Métricas de Retrabalho

Como Product Owner Quero visualizar métricas de retrabalho e tempo médio de refinamento Para monitorar a saúde do processo de refinamento

Critérios de Aceite:

- Dashboard exibe: % de histórias reabertas, tempo médio de preparação, tempo médio de sessão.
- Dados são agrupados por sprint ou período customizável.
- É possível exportar métricas em CSV ou PDF.

RF06 – Integração com Jira

Como Product Owner Quero sincronizar histórias com Jira Para evitar retrabalho de cadastro manual

Critérios de Aceite:

- É possível conectar a conta do Jira via API (OAuth).
- O sistema importa backlog existente.
- Alterações feitas na ferramenta são refletidas no Jira (ao menos de forma unidirecional).
- Em caso de erro de sincronização, exibe log e permite tentativa de novo envio.

RF07 – Visualização de Fluxo de Valor e Dependências

Como Product Owner Quero visualizar o fluxo de valor e dependências entre histórias Para evitar bloqueios e planejar releases de forma eficiente

Critérios de Aceite:

- O sistema exibe relações entre histórias em um diagrama interativo (ex.: gráfico de dependências).
- Permite identificar histórias bloqueadoras e bloqueadas.
- Permite filtrar visualização por épico, sprint ou status.
- Atualiza automaticamente conforme mudanças no backlog.

RF08 – Assistente Guiado de Escopo

Como Product Owner Quero responder a um formulário guiado (wizard) com perguntas INVEST Para gerar automaticamente a descrição e os critérios de aceite iniciais de uma história.

Critérios de Aceite:

- Dado que o PO iniciou o assistente de escopo
- Quando ele responder às perguntas de contexto (quem, o quê, por quê)
- Então a história é gerada com título, descrição e checklist INVEST pré-preenchidos
- E é possível editar antes de salvar no backlog.

RF09 – Validação Técnica Pré-Sprint

Como time técnico Quero aprovar ou rejeitar histórias quanto à viabilidade técnica

Para garantir que apenas histórias viáveis entrem no sprint.

Critérios de Aceite:

- Dado que uma história está no estado “Aguardando Validação Técnica”
- Quando o time técnico revisa e aprova
- Então a história muda para “Pronta para Refinamento”
- E se rejeitada, deve ser possível incluir um motivo e enviar de volta ao PO.

RF10 – Matriz de Valor x Esforço

Como Product Owner Quero visualizar os itens do backlog em uma matriz de valor x esforço para facilitar a priorização junto aos stakeholders.

Critérios de Aceite:

- Dado que os itens possuem valor e estimativa de esforço preenchidos
- Quando o PO abre a matriz

- Então o sistema posiciona automaticamente cada item em quadrantes (alto valor / baixo esforço etc.)
- E permite reclassificar arrastando os itens.

RF11 – Histórico Versionado e Restauração

Como Product Owner Quero restaurar uma versão anterior do backlog Para desfazer mudanças incorretas ou revertidas.

Critérios de Aceite:

- Dado que houve alterações no backlog
- Quando o PO acessar o histórico
- Então o sistema lista as versões anteriores com data e autor
- E permite restaurar qualquer versão anterior.

RF12 – Colaboração e Comentários

Como membro do time Quero comentar, mencionar colegas e votar em histórias Para facilitar alinhamento e tomada de decisão.

Critérios de Aceite:

- Permite @menções em comentários
- Exibe histórico de conversas dentro da história
- Permite votação (like, score) para facilitar priorização colaborativa.

RF13 – Dashboards Avançados

Como Product Owner Quero acompanhar burndown, lead time e velocidade do time Para medir a eficiência do processo de refinamento e entrega.

Critérios de Aceite:

- Dashboard apresenta burndown atualizado
- Exibe tempo médio de preparação de histórias
- Mostra comparação de planejado vs entregue por sprint.

RF14 – Integração com Ferramentas de Comunicação

Como PO Quero que mudanças de prioridade sejam notificadas no Slack/Teams Para manter o time informado em tempo real.

Critérios de Aceite:

- Ao alterar prioridade, o sistema envia notificação no canal configurado
- Inclui link direto para o item alterado
- Permite desligar ou configurar notificações.

C.2 Requisitos Não Funcionais

Código	Descrição	Métrica de Aceitação

RNF01 – Usabilidade	A interface deve ser responsiva, intuitiva e acessível, permitindo navegação simples sem necessidade de treinamento extenso.	NPS ≥ 8 em testes de usabilidade e conformidade mínima com WCAG 2.1.
RNF02 – Colaboração em Tempo Real	Alterações no backlog devem ser refletidas para todos os usuários ativos em até 2 segundos.	Latência < 2s em 95% dos eventos de atualização.
RNF03 – Performance	O sistema deve carregar páginas e atualizar backlog rapidamente, mesmo com grandes volumes de dados.	Tempo de resposta < 2s para backlog com até 500 histórias.
RNF04 – Disponibilidade	O sistema deve estar disponível durante o horário comercial para suportar sessões de refinamento.	Uptime ≥ 99,5% mensal.
RNF05 – Segurança e Permissões	Autenticação via OAuth e controle de permissões por papel (PO, Dev, Stakeholder), garantindo confidencialidade dos dados.	100% dos acessos registrados e criptografados (AES-256).
RNF06 – Auditabilidade	Todas as alterações do backlog devem ser logadas, permitindo rastreabilidade completa.	100% das alterações possuem log com data, hora e autor.
RNF07 – Escalabilidade	Suporte a pelo menos 50 usuários simultâneos sem degradação perceptível de performance.	Tempo de resposta < 3s em 95% das requisições em simulação de carga.
RNF08 – Confiabilidade	Alterações feitas offline devem ser armazenadas localmente e sincronizadas após reconexão.	100% das alterações offline enviadas ao servidor após restabelecimento da conexão.

APÊNDICE D – BACKLOG COMPLETO DO TRELLO

D.1 Épicos

Épico E1 — Qualidade e Prontidão das Histórias de Usuário

Apoiar o refinamento das histórias de usuário por meio da aplicação sistemática de critérios de qualidade e prontidão e registrar o contexto das discussões realizadas durante o refinamento.

Épico E2 — Priorização Guiada e Rastreabilidade das Decisões

Apoiar a priorização do backlog por meio da aplicação de técnicas leves e da explicitação das decisões tomadas, permitindo compreender não apenas a ordem das histórias, mas também os motivos que levaram a essa ordem.

Épico E3 — Apoio Visual à Tomada de Decisão

Oferecer visualizações simples que auxiliem a análise comparativa entre histórias, facilitando discussões sobre valor e esforço durante o processo de priorização.

Épico E4 — Métricas Mínimas do Processo de Refinamento

Disponibilizar métricas simples e diretamente relacionadas ao refinamento das histórias, permitindo observar padrões de retrabalho e esforço sem introduzir complexidade excessiva.

D.2 Features

Feature F1.1 — Checklist INVEST por história

Permitir a visualização e atualização dos critérios INVEST em cada história de usuário, sinalizando seu nível de prontidão durante o refinamento.

Feature F1.2 — Identificação e divisão de histórias grandes

Auxiliar na identificação de histórias excessivamente grandes e sugerir sua divisão durante o refinamento.

Feature F1.3 — Comentários contextuais nas histórias

Registrar comentários associados às histórias de usuário, preservando o contexto das discussões realizadas durante o refinamento.

Feature F2.1 — Priorização guiada por técnicas leves

Permitir a aplicação de técnicas de priorização selecionadas pelo Product Owner,

organizando o backlog de acordo com os critérios definidos.

Feature F2.2 — Registro de justificativas de priorização

Registrar justificativas associadas às alterações de prioridade das histórias, promovendo rastreabilidade e transparência das decisões.

Feature F3.1 — Matriz Valor × Esforço

Disponibilizar uma visualização gráfica das histórias em uma matriz de valor versus esforço, permitindo análise comparativa e reclassificação dos itens.

Feature F4.1 — Métricas de preparação e retrabalho

Apresentar métricas simples relacionadas ao processo de refinamento, como percentual de reabertura, tempo médio de preparação das histórias e duração das sessões.

D.3 Histórias de Usuário

US 1.0.1 — Criar história de usuário com estrutura padrão

Como Product Owner, quero criar uma história de usuário informando título e descrição, para iniciar o processo de refinamento dentro da ferramenta.

Critérios de aceite:

1. O sistema permite criar uma nova história informando, no mínimo, título e descrição.
2. Ao ser criada, a história possui automaticamente um checklist INVEST não preenchido.
3. A história criada passa a integrar o backlog da ferramenta.

US 1.1.1 — Gerenciar checklist INVEST por história

Como Product Owner, quero marcar os seis critérios do checklist INVEST em cada história de usuário, para avaliar sua qualidade e prontidão durante o refinamento, antes da priorização.

Critérios de aceite:

1. O sistema exibe os seis critérios do INVEST associados à história.
2. Cada critério pode ser marcado ou desmarcado individualmente pelo usuário.
3. O estado visual da história é atualizado automaticamente conforme o preenchimento do checklist.
4. Quando todos os critérios estiverem marcados, a história é sinalizada como “Refinada”.

US 1.1.2 — Restringir avanço para estado “Pronto (DoR)” sem INVEST completo

Como Product Owner, quero que histórias não possam avançar para o estado “Pronto (DoR)” enquanto o checklist INVEST não estiver completo, para garantir um nível mínimo de qualidade antes do planejamento.

Critérios de aceite:

1. Ao tentar mover a história para “Pronto (DoR)”, o sistema verifica se todos os critérios do INVEST estão marcados.
2. Caso algum critério esteja desmarcado, o avanço é impedido.
3. O sistema exibe uma mensagem informando quais critérios impedem o avanço.
4. Após o preenchimento completo do checklist INVEST, a história pode avançar para “Pronto (DoR)” sem restrições adicionais.

US 1.2.1 — Identificar e sinalizar histórias grandes

Como Product Owner, quero que o sistema identifique automaticamente histórias que ultrapassem um limite de tamanho definido, para apoiar a decisão de dividi-las durante o refinamento.

Critérios de aceite:

1. O limite máximo de tamanho da história é configurável.
2. Quando a estimativa da história ultrapassar o limite definido, o sistema sinaliza visualmente a história como “Grande”.
3. Caso a estimativa seja ajustada para um valor igual ou inferior ao limite, a sinalização é removida automaticamente.

US 1.3.1 — Registrar comentários contextuais nas histórias

Como Product Owner, quero registrar comentários associados às histórias de usuário, para preservar o contexto das discussões realizadas durante o refinamento e apoiar auditorias futuras.

Critérios de aceite:

1. O sistema permite adicionar comentários textuais a uma história.
2. Os comentários são exibidos em ordem cronológica.
3. Cada comentário apresenta autor e data de registro.
4. O autor do comentário pode removê-lo.

US 2.1.1 — Priorizar backlog utilizando a técnica MoSCoW

Como Product Owner, quero priorizar histórias de usuário utilizando a técnica MoSCoW, para facilitar o alinhamento com stakeholders durante a definição de escopo.

Critérios de aceite:

1. O Product Owner pode selecionar a técnica MoSCoW.
2. Cada história pode ser classificada como Must, Should, Could ou Won’t.
3. O backlog é ordenado de acordo com a classificação definida.

US 2.1.2 — Priorizar backlog utilizando a técnica CSD

Como Product Owner, quero priorizar histórias de usuário utilizando a técnica CSD, para

apoiar decisões rápidas em contextos de incerteza.

Critérios de aceite:

1. O Product Owner pode selecionar a técnica CSD.
2. Cada história pode ser classificada como Certeza, Suposição ou Dúvida.
3. O backlog é ordenado conforme a classificação aplicada.

US 2.1.3 — Priorizar backlog utilizando a técnica GUT

Como Product Owner, quero priorizar histórias de usuário utilizando a técnica GUT, para concentrar esforços nas demandas mais críticas.

Critérios de aceite:

1. O Product Owner pode selecionar a técnica GUT.
2. Cada história recebe valores de Gravidade, Urgência e Tendência.
3. O backlog é ordenado com base no resultado da aplicação da técnica.

US 2.2.1 — Registrar justificativas de alteração de prioridade

Como Product Owner, quero registrar justificativas quando a prioridade for alterada, para manter a rastreabilidade e transparência das decisões tomadas.

Critérios de aceite:

1. O sistema permite adicionar justificativas textuais a uma história.
2. As justificativas são exibidas em ordem cronológica.
3. Cada justificativa apresenta autor e data de registro.
4. O autor da justificativa pode removê-la.

US 3.1.1 — Visualizar histórias na matriz Valor × Esforço

Como Product Owner, quero visualizar as histórias de usuário em uma matriz de Valor × Esforço, para apoiar discussões de priorização com stakeholders.

Critérios de aceite:

1. A matriz é composta por quatro quadrantes (alto/baixo valor × alto/baixo esforço).
2. O posicionamento das histórias é realizado automaticamente com base nos campos de valor e esforço.
3. O sistema permite aplicar filtros básicos por épico, feature ou status.

US 3.1.2 — Reclassificar histórias na matriz e exportar visualização

Como Product Owner, quero reclassificar histórias diretamente na matriz Valor × Esforço e exportar a visualização, para ajustar prioridades de forma dinâmica e registrar decisões.

Critérios de aceite:

1. O Product Owner pode mover histórias entre os quadrantes por meio de arrastar e soltar.

2. A reclassificação atualiza automaticamente os campos de valor e esforço da história.
3. O estado da matriz é salvo após alterações.
4. A matriz pode ser exportada como imagem.

US 4.1.1 — Ver % de reabertura e tempo médio de preparação

Como PO, quero ver % de reabertura e o tempo médio de criação → "Pronto/DoR" para melhorar o processo.

Critérios de aceite:

1. Cálculo por período/sprint (filtros).
2. Indicadores + série temporal simples.
3. Exportação CSV/PDF.

US 4.1.2 — Visualizar duração média das sessões de refinamento

Como Product Owner, quero visualizar a duração média das sessões de refinamento, para planejar melhor o esforço dedicado a essa atividade.

1. Critérios de aceite:
2. O sistema registra a duração de cada sessão de refinamento.
3. As informações podem ser agregadas por período ou sprint.
4. Os dados podem ser exportados em formato CSV ou PDF.