



Universidade de Brasília – UnB
Faculdade de Ciência e Tecnologia em Engenharia – FCTE
Engenharia de Software

Hubot: Um chatbot para elaboração de histórias de usuário

Autores: Antonio Igor Carvalho e Lameque Fernandes Azevedo

Orientadora: Prof.a Dr.a Edna Dias Canedo

Brasília, DF

2025



Antonio Igor Carvalho e Lameque Fernandes Azevedo

Hubot: Um chatbot para elaboração de histórias de usuário

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

Universidade de Brasília – UnB
Faculdade de Ciência e Tecnologia em Engenharia – FCTE

Orientador: Prof.a Dr.a Edna Dias Canedo
Coorientador: Prof.a M. Sc. Geovana Ramos Sousa Silva

Brasília, DF
2025

Antonio Igor Carvalho e Lameque Fernandes Azevedo

Hubot: Um chatbot para elaboração de histórias de usuário/ Antonio Igor Carvalho e Lameque Fernandes Azevedo. – Brasília, DF, 2025-

98 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof.a Dr.a Edna Dias Canedo

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade de Ciência e Tecnologia em Engenharia – FCTE , 2025.

1. Chatbot. 2. Histórias de usuário. I. Prof.a Dr.a Edna Dias Canedo. II. Universidade de Brasília. III. Faculdade de Ciência e Tecnologia em Engenharia. IV. Hubot: Um chatbot para elaboração de histórias de usuário

CDU 02:141:005.6

Antonio Igor Carvalho e Lameque Fernandes Azevedo

Hubot: Um chatbot para elaboração de histórias de usuário

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

Prof.a Dr.a Edna Dias Canedo
Orientador

Prof.a Dr.a Milene Serrano
Convidado 1

Prof. Dr. André Barros de Sales
Convidado 2

Brasília, DF
2025

Agradecimentos

Agradecemos, primeiramente, a Deus, por nos ter sustentado ao longo de todo o curso, concedendo-nos força e perseverança diante de todos os desafios enfrentados durante nossa jornada de graduação.

Manifestamos nossa imensa gratidão aos nossos familiares e amigos, que sempre estiveram ao nosso lado, oferecendo apoio e motivação em cada etapa desse percurso.

Agradecemos especialmente à professora Dr.a Edna Dias Canedo e à professora MSc. Geovana Ramos Sousa Silva, por terem aceitado nos orientar com tanta disposição. Sua dedicação, paciência e comprometimento em nos guiar ao longo do desenvolvimento desta pesquisa foram inestimáveis. Seus ensinamentos e conhecimentos foram fundamentais para o nosso crescimento acadêmico.

Estendemos também nossos agradecimentos à professora Dr.a Milene Serrano e ao professor Dr. André Barros, que gentilmente aceitaram o convite para compor a banca de avaliação. Suas análises e contribuições foram igualmente indispensáveis para a conclusão deste trabalho.

Por fim, nosso sincero agradecimento a todos os professores da Faculdade do Gama, da Universidade de Brasília, cujo ensino e dedicação foram essenciais para nos trazer até aqui.

Resumo

No contexto de metodologias ágeis, as histórias de usuário são amplamente utilizadas para documentar requisitos de software, mas o desenvolvimento dessas histórias pode se tornar difícil devido à inexperiência na área ou falta de atenção. O principal objetivo deste trabalho foi o desenvolvimento de um *chatbot*, chamado de HuBot, para auxiliar usuários na criação e avaliação de histórias de usuário, visando o desenvolvimento de histórias de usuário padronizadas e estruturadas, além de fornecer informações importantes para usuários com pouco conhecimento em requisitos de software. Além disso o *chatbot* também permite a avaliação das histórias de usuário desenvolvidas, com o auxílio de um *prompt* estruturado com técnicas de engenharia de *prompt* para receber as devidas melhorias e correções de um *large language model* (LLM) auxiliador. Espera-se que o HuBot promova uma curva de aprendizado mais rápida para usuários menos experientes, ajudando na criação de histórias de usuário com qualidade, além de fornecer uma forma automatizada de verificação da qualidade das histórias, facilitando e melhorando o processo de desenvolvimento de histórias de usuário como um todo. Os resultados obtidos por meio de um questionário indicaram que a maioria dos usuários considerou sua experiência com o HuBot “satisfatória” ou “muito satisfatória”, destacando a avaliação de histórias de usuário. No entanto, também foram apontadas fragilidades, especialmente por usuários com pouca experiência em histórias de usuário ou familiaridade com *chatbots*.

Palavras-chave: chatbot, histórias de usuário, metodologias ágeis, engenharia de prompt, requisitos de software.

Abstract

In the context of agile methodologies, user stories are widely used to document software requirements, but developing these stories can become challenging due to inexperience in the area or lack of attention. The main objective of this work was to develop a chatbot, called HuBot, to assist users in creating and evaluating user stories, aiming to develop standardized and structured user stories, and to provide important information for users with little knowledge in software requirements. Additionally, the chatbot will also allow the evaluation of developed user stories with the aid of a structured prompt using prompt engineering techniques to receive the necessary improvements and corrections from a supporting large language model (LLM). It is expected that HuBot will promote a faster learning curve for less experienced users, helping in the creation of quality user stories, and providing an automated way to verify the quality of the stories, facilitating and improving the overall user story development process. The results obtained through a questionnaire indicated that most users considered their experience with HuBot “satisfactory” or “very satisfactory”, highlighting the evaluation of user stories. However, weaknesses were also pointed out, especially by users with little experience in user stories or familiarity with chatbots.

Key-words: chatbot, user stories, agile methodologies, prompt engineering, software requirements.

Lista de ilustrações

Figura 1 – Arquitetura da Solução	41
Figura 2 – Fluxo 1: Menu de ações do HuBot e seleção de opção.	62
Figura 3 – Fluxo 1, pt. 1: Preenchimento do tipo de usuário, objetivo e motivo.	63
Figura 4 – Fluxo 2, pt. 2: Analise e sugestões para a HU criada.	64
Figura 5 – Fluxo 2, pt. 3: Resultado final da HU, com melhorias aderidas.	64
Figura 6 – Fluxo 3, pt. 1: Preenchimento do tipo de usuário, objetivo e motivo.	65
Figura 7 – Fluxo 3, pt. 2: Analise e sugestões para a HU criada.	66
Figura 8 – Fluxo 3, pt. 3: Resultado final da HU, com algumas melhorias aderidas.	66
Figura 9 – Distribuição das respostas sobre a concordância com a participação no estudo.	68
Figura 10 – Confirmação de uso do chatbot pelos participantes.	68
Figura 11 – Distribuição da familiaridade com chatbots.	69
Figura 12 – Nível de conhecimento dos participantes sobre histórias de usuário.	69
Figura 13 – Nível de experiência dos participantes em desenvolvimento de software.	70
Figura 14 – Papel ou função desempenhada por participantes com experiência em desenvolvimento de software.	71
Figura 15 – Amigabilidade da interação inicial com o HuBot.	71
Figura 16 – Facilidade de entendimento do propósito do HuBot.	72
Figura 17 – Clareza no fluxo de criação de histórias de usuário no HuBot.	74
Figura 18 – Nível de satisfação com o fluxo de definição das partes que compõem a história de usuário.	75
Figura 19 – Avaliação das sugestões de melhorias para as histórias de usuário levantadas pelo HuBot.	76
Figura 20 – Avaliação da clareza das explicações fornecidas pelo HuBot.	77
Figura 21 – Avaliação geral da experiência com o HuBot.	79
Figura 22 – Avaliação da utilidade do HuBot na criação de histórias de usuário.	80
Figura 23 – Primeira tentativa de prompt para avaliação de histórias de usuário	94
Figura 24 – Segunda tentativa de prompt para avaliação de histórias de usuário	95
Figura 25 – Terceira tentativa de prompt para avaliação de histórias de usuário	98

Lista de tabelas

Tabela 1 – Requisitos	40
Tabela 2 – Código e Descrição	57
Tabela 3 – Backlog do Produto com Prioridades	58
Tabela 4 – Tabela de Ritos	58
Tabela 5 – Tabela de Impacto	59
Tabela 6 – Tabela de Riscos e Medidas	60

Lista de abreviaturas e siglas

API	Application Programming Interface
ER	Engenheiro de Requisitos
GPT	Generative Pre-trained Transformer
HTTPS	HyperText Transfer Protocol Secure
HU	História de Usuário
IA	Inteligência artificial
IBM	International Business Machines Corporation
LLM	Large Language Model
NLP	Natural Language Processing
PO	Product Owner
SSL	Secure Sockets Layer
TLS	Transport Layer Security
US	User story
XP	Extreme Programming

Sumário

1	INTRODUÇÃO	21
1.1	Contextualização	21
1.2	Objetivos	21
1.2.1	Objetivo Geral	21
1.2.2	Objetivos Específicos	22
1.3	Problema	22
1.4	Resultado esperado	23
1.5	Metodologia de Pesquisa	23
1.6	Estrutura da Monografia	24
2	REFERENCIAL TEÓRICO	27
2.1	Chatbot	27
2.1.1	Tecnologias para desenvolvimento de chatbots	28
2.2	Prompt Engineering	29
2.3	Engenharia de requisitos	30
2.3.1	Elicitação de Requisitos	31
2.3.2	Engenharia de Requisitos no Contexto Ágil	32
2.3.2.1	Metodologia Ágil	32
2.3.2.2	Scrum	34
2.3.2.3	História de Usuário	35
2.4	Trabalhos Relacionados	36
2.4.1	User stories collection via interactive chatbot to support requirements gathering	36
2.4.2	Intelligent Chatbot for Requirements Elicitation and Classification	36
2.4.3	Framework for automating requirement elicitation using a chatbot	37
2.4.4	LadderBot: A Requirements Self-Elicitation System	37
3	PROPOSTA DO SISTEMA	39
3.1	Contextualização	39
3.2	Requisitos	39
3.3	Arquitetura	41
3.4	Visão geral das tecnologias e ferramentas utilizadas	42
3.4.1	Git	42
3.4.2	GitHub	43
3.4.3	Python (Versão 3.10)	43
3.4.4	Rasa Open Source (Versão 3.6)	43

3.4.5	ChatGPT (Versão 4)	44
3.4.6	Nginx	44
4	GERAÇÃO DO PROMPT	45
4.1	Fundamentação e Definição da Solução	45
4.2	Técnicas Utilizadas	45
4.3	Tentativas	46
4.3.1	Prompt 1	47
4.3.2	Prompt 2	50
4.3.3	Prompt 3 - Final	53
5	DESENVOLVIMENTO DO SISTEMA	57
5.1	Planejamento de Desenvolvimento de Software	57
5.1.1	Backlog do Produto	57
5.1.2	Plano de Iterações	57
5.1.3	Cronograma de Desenvolvimento Simplificado	58
5.1.4	Comunicação	59
5.1.5	Gestão de Riscos	59
5.1.5.1	Métricas	59
5.1.6	Código-Fonte	59
5.1.7	Implementação do Chatbot	60
5.1.8	Configuração da Máquina Virtual	61
5.1.9	Registro de Subdomínio e Configuração de Certificado SSL	61
5.1.10	Configuração do Servidor Nginx	61
5.1.11	Execução do Chatbot	61
5.1.12	Integração com o Telegram	61
5.1.13	HuBot em funcionamento	62
6	VALIDAÇÃO E RESULTADOS	67
6.1	Método de Avaliação do HuBot	67
6.2	Formulário e Resultados	67
6.2.1	Confirmação de Participação e Uso do Chatbot	67
6.2.2	Qual é a sua familiaridade com chatbots?	68
6.2.3	Qual é o seu nível de conhecimento sobre histórias de usuário?	68
6.2.4	Como você avalia sua experiência em desenvolvimento de software?	69
6.2.5	Papel ou Função no Desenvolvimento de Software	70
6.2.6	Quão amigável achou a interação inicial com o HuBot?	71
6.2.7	Quão fácil foi entender o propósito do HuBot?	72
6.2.8	Sugestões para Melhorar a Interação Inicial	72
6.2.9	Clareza no fluxo de criação de histórias de usuário	73

6.2.10	Sugestões e Experiências com o Fluxo de Criação de Histórias de Usuário	74
6.2.11	Satisfação com o Fluxo de Definição das Partes que Compõem a História de Usuário	75
6.2.12	Avaliação das Sugestões de Melhorias para as Histórias de Usuário	75
6.2.13	Clareza nas Explicações Fornecidas pelo HuBot	76
6.2.14	Principais Pontos Positivos do HuBot	77
6.2.15	Dificuldades e Problemas Durante a Interação	78
6.2.16	Sugestões de Melhorias para o HuBot	78
6.2.17	Avaliação Geral do HuBot	79
6.2.18	Utilidade do HuBot para a Criação de Histórias de Usuário	80
7	CONCLUSÃO	81
7.1	Objetivos Alcançados	81
7.2	Competências do HuBot	81
7.3	Fragilidades do HuBot	82
7.4	Trabalhos Futuros	82
	REFERÊNCIAS	85
	APÊNDICES	91
	APÊNDICE A – PRIMEIRO TENTATIVA DE CRIAÇÃO DE PROMPT PARA AVALIAÇÃO DE HISTÓRIAS DE USUÁRIO	93
	APÊNDICE B – SEGUNDA TENTATIVA DE CRIAÇÃO DE PROMPT PARA AVALIAÇÃO DE HISTÓRIAS DE USUÁRIO	95
	APÊNDICE C – TERCEIRA TENTATIVA DE CRIAÇÃO DE PROMPT PARA AVALIAÇÃO DE HISTÓRIAS DE USUÁRIO	97

1 Introdução

Este capítulo apresenta a contextualização, os objetivos e os problemas que este trabalho visa resolver, assim como uma breve explicação sobre o tema.

1.1 Contextualização

Uma técnica amplamente utilizada para documentar requisitos de software são as histórias de usuário, especialmente por sua simplicidade, conceitos oriundos das metodologias ágeis e por trazer os requisitos sob a visão do usuário, ao invés do sistema (MENZINSKY et al., 2018), mantendo assim, uma boa comunicação com o cliente (SANTANA; KUDO; BULCAO-NETO, 2022).

As histórias de usuário são amplamente utilizadas para representar requisitos de maneira simples, seguindo um formato semelhante a “Como <persona>, quero <objetivo desejado> para que <valor a ser alcançado>” (LUCASSEN FABIANO DALPIAZ, 2016). Embora essa abordagem seja relativamente fácil de se aplicar, por seguir um formato previamente definido, sua elaboração pode apresentar falhas, não representando bem as necessidades dos clientes (LUCASSEN et al., 2017).

Outra tecnologia em ascensão são os *chatbots* que utilizam LLM. Eles trazem uma gama de inovações e excelentes meios para o processamento da linguagem natural. Isso fica mais claro ao obter respostas surpreendentemente complexas e muito próximas da linguagem humana (ZHAO et al., 2023), sendo em alguns casos até quase indistinguíveis. Com isso, outra coisa que tem chamado cada vez mais a atenção é a Engenharia de *Prompt*, também conhecida como *Prompt Engineering*. Com a alta demanda de tecnologias que utilizam LLMs, observou-se a importância de um bom *prompt* de entrada para obter respostas mais assertivas dos *chatbots* que usufruem dos LLMs.

1.2 Objetivos

1.2.1 Objetivo Geral

O principal objetivo deste trabalho foi o desenvolvimento de um *chatbot* que seja capaz de ajudar usuários a levantar e avaliar histórias de usuário. O HuBot foi desenvolvido para gerar histórias de usuário (HUs) de uma maneira padronizada e estruturada. Outra funcionalidade visionada para o *chatbot* é fornecer informações essenciais sobre histórias de usuário, auxiliando o público que possui baixo aprendizado em requisitos de software.

Além da elição de HUs, o *chatbot* também possibilita a avaliação das histórias criadas durante o levantamento, evitando possíveis erros de digitação e inconsistências nas ideias. Desta forma, o trabalho procura contribuir na etapa de elição de requisitos, etapa esta, muito relevante para o sucesso ou não de um projeto de software, além de promover maior facilidade e assertividade nesta etapa.

1.2.2 Objetivos Específicos

Para alcançar o objetivo geral proposto, foram definidos os seguintes objetivos específicos:

- Pesquisar as principais formas de se escrever uma história de usuário e de criar *prompts*.
- Criar um *prompt* adequado para obter uma avaliação estruturada da HU em processo de criação, através do LLM que será consumido.
- Validar a eficácia do *prompt* através de testes.
- Fornecer informações básicas sobre histórias de usuário.

1.3 Problema

Dentro do contexto de desenvolvimento de software utilizando metodologias ágeis, a criação de histórias de usuário tornou-se uma prática comum durante o desenvolvimento de software (O’HEOCHA; CONBOY, 2010), para registrar e tentar expressar de forma clara e concisa as necessidades dos usuários finais, a fim de direcionar a equipe de desenvolvimento. Entretanto, a elaboração de boas histórias de usuário pode ser considerada uma tarefa difícil e complexa por diversos fatores, como a falta de experiência de uma equipe na área de levantamento de requisitos, e isso pode levar à criação de histórias de usuário que não representam a real necessidade dos clientes (LUCASSEN et al., 2017).

Histórias de usuário mal formuladas nem sempre apresentam as reais necessidades dos clientes, resultando em retrabalho para a equipe e insatisfação do cliente, impactando fortemente a qualidade do produto final, já que o retrabalho pode ter um impacto significativo no tempo gasto para o desenvolvimento de uma funcionalidade, afetando diretamente o cronograma de um projeto.

A falta de ferramentas que oferecem um processo automatizado de verificação de histórias de usuário, faz com que equipes dependam de revisões manuais, que são suscetíveis a erros humanos e dependem do conhecimento da equipe acerca do assunto, além de consumir um maior tempo da equipe e possíveis retrabalhos, levando a um aumento no

tempo de entrega do produto final. Portanto, o desenvolvimento de uma ferramenta que possa auxiliar no processo de criação de histórias de usuário pode vir a contribuir para a mitigação desse problema e aumentar a chance de sucesso de um projeto.

1.4 Resultado esperado

A proposta de desenvolvimento de um *chatbot* feito para auxiliar o processo de criação de histórias de usuário, visa proporcionar ao seus usuários uma solução para elaboração e avaliação de histórias de usuário. A expectativa é que o *chatbot* desenvolvido atue como um assistente inteligente, auxiliando durante a criação das histórias, respondendo dúvidas e corrigindo possíveis erros e inconsistências. Ao fornecer esse suporte imediato e automatizado espera-se que o HuBot contribua no aumento da qualidade de histórias criadas, em comparação a criação sem nenhum auxílio da ferramenta, além disso, a expectativa é que o HuBot também promova uma curva de aprendizado mais rápida para equipes com menos experiência, visto que elas poderão comparar suas histórias com a versão melhorada pela ferramenta, permitindo que aprendam e aprimorem suas habilidades de forma contínua.

1.5 Metodologia de Pesquisa

Esta seção descreve os métodos e procedimentos utilizados para conduzir a pesquisa, assim como foi classificada em relação à abordagem, natureza, objetivos, procedimentos técnicos e técnicas de coleta de dados.

O trabalho partiu de uma pesquisa bibliográfica e documental, para identificar, avaliar e interpretar diversas pesquisas relevantes sobre a forma de criação, geração e avaliação de histórias de usuários, além de buscar e identificar os impactos que um *prompt* pode gerar ao utilizar um LLM.

Os aspectos definidos para a metodologia foram:

- Abordagem: Pesquisa qualitativa.
- Natureza: Pesquisa aplicada.
- Objetivos: Pesquisa exploratória.
- Procedimentos técnicos: Pesquisa bibliográfica, Pesquisa documental e Validação por questionário.

Aderiu-se à abordagem qualitativa, pois o foco é aprofundar-se na compreensão, transformando conceitos em resultados e resultados em análises (QUINN, 2002). Já a

natureza da pesquisa, qualifica-se como aplicada, visto que o trabalho visa resolver o problema de forma “prática”, na Engenharia de Requisitos, facilitando o processo de criação de HUs através de um *chatbot*. O objetivo da pesquisa segue no mesmo sentido, classificando-se como exploratória, em virtude de que, propõem a ideia de analisar/avaliar histórias de usuário através de um *chatbot*, que se comunicar com um LLM através de *prompts* bem estruturados (MARCONI; LAKATOS, 2003).

A pesquisa bibliográfica foi amplamente utilizada durante o desenvolvimento do trabalho, foram utilizados diversos materiais já elaborados, como livros e artigos científicos qualificados. Já a pesquisa documental possui pouca incidência no trabalho, sendo utilizada em pontos específicos para complementar informações com dados de fontes primárias.

A busca por artigos e publicações foi realizada principalmente em duas plataformas: Google Scholar e IEEE Xplore. Os termos de busca utilizados incluíram palavras-chave como “user stories”, “prompt engineering”, “large language models”, “chatbot”, “requirements engineering”, entre outros relacionados ao escopo do projeto. O processo de análise seguiu a seguinte abordagem: inicialmente, os resumos dos artigos foram lidos para verificar sua relevância, e, caso o conteúdo estivesse alinhado aos objetivos do trabalho, procedeu-se à leitura integral do artigo para extrair as informações necessárias. Assim, as publicações selecionadas foram utilizadas como referências ao longo de todo o desenvolvimento deste trabalho. Vale ressaltar que também foram utilizadas algumas publicações de literatura cinzenta, especialmente em descrições de ferramentas e tecnologias, onde os próprios proprietários e mantenedores disponibilizam explicações sobre suas criações. Além da pesquisa bibliográfica e documental, foi realizada uma validação da ferramenta desenvolvida por meio da aplicação de um formulário. O questionário foi enviado a dois grupos distintos: alunos de Engenharia de Software e indivíduos sem conhecimento prévio na área. O objetivo foi avaliar o fluxo de criação das HUs, a avaliação de HUs geradas, a utilidade das explicações fornecidas pelo chatbot e a facilidade de uso da ferramenta.

1.6 Estrutura da Monografia

Este trabalho está dividido em outros 6 capítulos, sendo eles:

- **Referencial Teórico:** onde é apresentado os conceitos relacionados e importantes para o entendimento do tema.
- **Proposta do Sistema:** onde é apresentado a proposta de desenvolvimento do *chatbot* (HuBot), e apresenta conceitos como a arquitetura de software, etc.

- **Geração do *Prompt*:** neste capítulo é apresentado o *prompt* levantado para a avaliação das histórias de usuário e o caminho percorrido até chegar no *prompt* ideal.
- **Desenvolvimento do Sistema:** este capítulo apresenta o processo de desenvolvimento do HuBot, detalhando desde o planejamento inicial até sua implementação.
- **Validação e Resultados:** neste capítulo são descritos os métodos utilizados para validar o *chatbot* e os resultados obtidos em um formulário de avaliação aplicado a usuários reais.
- **Conclusão:** apresenta as conclusões do tema e possíveis evoluções.

2 Referencial Teórico

Este capítulo apresenta os fundamentos teóricos que sustentam este trabalho, abordando conceitos sobre *chatbots*, técnicas de *prompt engineering*, princípios da engenharia de requisitos e, por fim, uma revisão de trabalhos relacionados ao tema.

2.1 Chatbot

Um *chatbot* é um software que responde como uma entidade inteligente (KHANNA et al., 2015). Trata-se de uma tecnologia presente na história desde 1966 (WEIZENBAUM, 1966) e se mostra com potencial, sendo melhorada continuamente ao longo dos anos, conseguindo gerar respostas de maneira específica em relação ao que foi perguntado (ADAMOPOULOU; MOUSSIADES, 2020). Os *chatbots* possuem a capacidade de processar e interpretar linguagem natural (CAHN, 2017). O objetivo principal de sua criação é o desenvolvimento de um agente que consiga conversar de uma maneira que usuários pensem que estão se comunicando com outra pessoa (SHARMA; GOYAL; MALIK, 2017) e não uma máquina, trazendo uma alternativa de automatização de cenários em que a interação humana é utilizada.

A utilização de *chatbots* está se tornando cada vez mais popular (ABDELLATIF, 2021). Seu uso é variado e pode ser aplicado em diversas áreas, por exemplo, serviços relacionados a viagens, setor bancário, saúde, centros de atendimento ao cliente e comércio eletrônico (SAVANUR et al., 2021), fornecendo a possibilidade de suporte imediato e sem tempo de espera, devido à capacidade de fornecer respostas de maneira quase instantânea para o usuário (CUNNINGHAM et al., 2019) e suportar a escalabilidade de processamento para lidar com o possível aumento da demanda, a depender da forma que foi implementado. Com isso, a utilização de *chatbots* pode contribuir tanto para a experiência do usuário quanto para a redução de custos de uma empresa (SAVANUR et al., 2021), já que podem demandar menos recursos financeiros comparados a contratação de atendentes humanos.

Um exemplo de *chatbot* disponível na indústria é o ChatGPT, desenvolvido pela empresa OpenAI. Trata-se de um *chatbot* que utiliza inteligência artificial para responder a perguntas feitas pelo usuário de maneira detalhada e muito semelhante a um texto escrito por um humano (HÖRNEMALM, 2023). Ele vem se tornando uma ferramenta popular utilizada por pessoas de diversas áreas. Especificamente no contexto de desenvolvimento de software, ele se tornou algo utilizado no dia a dia por inúmeros desenvolvedores como apoio a tarefas relacionadas à programação, mesmo que às vezes ele forneça informações incorretas ou falsas (HÖRNEMALM, 2023).

O desenvolvimento de *chatbots* varia conforme a tecnologia utilizada. Uma das principais diferenças está entre utilizar ou não inteligência artificial. Aqueles que não utilizam dessa tecnologia são baseados em regras (KUMAR; ALI, 2020), sendo programados para responder perguntas específicas definidas inicialmente, tornando o seu escopo limitado ao que foi feito previamente. Já os *chatbots* que utilizam inteligência artificial usam do algoritmo de inteligência artificial para entregar respostas para o usuário (WAILTHARE et al., 2018), possibilitando uma interação mais fluida e com uma maior gama de perguntas que podem ser respondidas de uma maneira satisfatória e adaptativa, sem a necessidade do desenvolvedor mapear de maneira específica todas as possibilidades de conversação.

Uma das tecnologias que podem ser integradas a *chatbots* para se utilizar da inteligência artificial é o *Large Language Model* (LLM), esse é um tipo de modelo de linguagem (KALYAN, 2024) que demonstrou um desempenho impressionante em diversas tarefas relacionadas ao processamento de linguagem natural (ZHAO et al., 2024), sendo capazes de executar tarefas como a geração de textos, tradução, criação de resumos, responder perguntas (KALYAN, 2024), entre outras possibilidades. São essencialmente modelos de *deep learning*, pré-treinados com uma grande quantidade de dados de texto e alinhado para as preferências humanas (KALYAN, 2024) e conseguem realizar tarefas que modelos menores não são capazes (WEI et al., 2022), a integração de LLMs em *chatbots* pode aprimorar a capacidade do sistema em entender e responder a consultas de usuários de forma mais natural e assertiva, já as atividades que podem ser realizadas por um LLM e não por modelos menores mudam com o tempo, conforme a evolução da tecnologia de arquitetura, dados com maior qualidade ou com o aprimoramento dos procedimentos de treino (WEI et al., 2022).

2.1.1 Tecnologias para desenvolvimento de chatbots

RASA

O Rasa Open Source é uma plataforma de código aberto baseada em inteligência artificial que fornece ferramentas para a construção de assistentes virtuais ou *chatbots* (RASA, 2024), permitindo com que desenvolvedores consigam criar, treinar e implantar *chatbots* baseados no aprendizado de máquina (BOCKLISCH et al., 2017). O RASA simplifica o processo de criação de *chatbots*, facilitando a tarefa de processamento e compreensão de linguagem natural presente nas mensagens recebidas de usuários e auxiliando na identificação de intenções e entidades presentes. Por ser uma plataforma de código aberto, ele oferece a possibilidade de compreensão do seu funcionamento e a maior personalização das suas funcionalidades, caso necessário. Por permitir a criação de assistentes que utilizam a inteligência artificial, ele permite que os *chatbots* melhorem continuamente com o tempo, via refinamentos com base em *feedbacks* do usuário e o retreinamento do modelo.

Um dos exemplos de aplicações utilizando o RASA foi presenciado durante a pandemia do coronavírus no Brasil, em que o governo brasileiro criou um programa para fornecer ajuda financeira a pessoas necessitadas, durante esse período, o PicPay, que foi uma das *fintechs* brasileiras utilizadas para receber o benefício, criou em dez dias um *chatbot* que validava se um usuário se qualificava para receber o auxílio do governo brasileiro e o ajuda a reivindicar seus benefícios ([CONNECTING..., 2021](#)).

BotKit

Botkit é uma biblioteca node.js de código aberto para desenvolvimento de *chatbots* ([PAVEL, 2021](#)), aplicativos e integrações para plataformas de mensagens. Desenvolvido pela Howdy.ai, que posteriormente foi adquirida pela Microsoft, o Botkit permite a criação de *chatbots* com suporte para diversas plataformas de mensagens, como o Slack, Facebook, Messenger, dentre outras. Além disso, suas funcionalidades podem ser estendidas por meio de *plugins*.

Dialogflow

O Dialogflow é uma plataforma de desenvolvimento de agentes de conversação que utilizam tecnologias de inteligência artificial oferecidas pela Google, possibilitando a criação de *bots* de voz ou *chatbots* para conversas com clientes de um negócio ([DIALOGFLOW, 2024](#)). O Dialogflow disponibiliza uma interface gráfica ([DIALOGFLOW, 2024](#)), permitindo a criação de agentes por pessoas que não tem experiência técnica avançada com programação. A ferramenta também permite a integração do agente criado com outros sistemas, como o LINE, Discord, Google Chat, Slack, Telegram, Twitter, entre outros. ([DIALOGFLOW, 2024](#)).

2.2 Prompt Engineering

Uma técnica utilizada para obter respostas mais precisas e com maior acurácia em *chatbots* que utilizam LLMs é o *prompt engineering*, um conjunto de técnicas que se concentra na criação, otimização e refinamento de *prompts*, buscando garantir a qualidade do resultado (resposta do *chatbot*) ([ARVIDSSON; AXELL, 2023](#)). Um *prompt* por sua vez é o que define o contexto da conversa com um *chatbot*, podendo ser uma frase, pergunta, instrução. É importante transmitir muito bem o que se deseja para obter uma resposta coerente, uma vez que a qualidade da resposta está diretamente relacionada com a qualidade do *prompt* fornecido ([WHITE et al., 2023](#)), ou seja, para uma boa saída é necessária uma boa entrada.

Diversos comportamentos relacionados a LLMs podem ser observados durante o desenvolvimento de um *prompt*. Uma pequena variação em um *prompt* pode causar mudanças drásticas na resposta do *chatbot*. Isso reforça a importância de criar uma entrada

de forma cuidadosa (RODRIGUEZ; DEARSTYNE; CLELAND-HUANG, 2023). Além disso, LLMs conseguem fornecer explicações detalhadas sobre suas próprias decisões, no entanto, não é possível confirmar se essas respostas são verídicas. Ainda assim, essa análise pode ser usada para aprimorar o *prompt*, tornando-o mais alinhado ao modelo utilizado (RODRIGUEZ; DEARSTYNE; CLELAND-HUANG, 2023).

LLMs distintos podem apresentar comportamentos variados, mesmo quando são treinados com os mesmos dados. Essa variação indica que o processo para se chegar a um *prompt* ideal não pode ser padronizado para todos os casos. A técnica mais otimizada para criar um bom *prompt* depende de vários fatores. Entre eles, destacam-se os recursos disponíveis, o modelo utilizado e o objetivo final do *prompt* (RODRIGUEZ; DEARSTYNE; CLELAND-HUANG, 2023).

Segundo a OpenAI, criadora do ChatGPT, algumas estratégias podem ser utilizadas durante a construção de *prompts*:

- Escrever instruções claras e adaptá-las de acordo com as respostas, por exemplo, se a resposta fornecida pelo modelo for muito grande, especificar que está esperando uma resposta menor para que o modelo saiba como responder (OPENAI, 2024).
- Dividir tarefas complexas em instruções menores e mais simples, geralmente os modelos tendem a errar mais ao tentar responder instruções muito complexas, ao realizar a divisão em algo mais simples e conciso os LLMs podem ter uma taxa de acerto maior (OPENAI, 2024).
- Fornecer tempo para que o modelo possa “pensar”, pois mesmo ao enviar uma instrução que nos pareça extremamente simples o modelo pode demorar para gerar uma resposta adequada, e algo sendo respondido de imediato pode resultar em uma maior margem de erro (OPENAI, 2024).
- Usar ferramentas externas para compensar as fraquezas do modelo que está sendo utilizado, executando tarefas específicas em outras ferramentas que são feitas exatamente para o contexto desejado (OPENAI, 2024).
- Testar mudanças no *prompt* sistematicamente, adaptando este de acordo com a interpretação da resposta fornecida pelo modelo (OPENAI, 2024).

2.3 Engenharia de requisitos

A Engenharia de Requisitos é o processo de levantar, analisar, documentar, gerenciar e controlar a qualidade dos requisitos (FALBO, 2017). É dentro desse processo que os requisitos funcionais e não funcionais, critérios de aceitação, devem ser documentados (POHL, 1996). Tradicionalmente, a Engenharia de Requisitos tem sido vista como

a primeira fase do ciclo de vida de um software, assim como sua relevância durante o desenvolvimento (FALBO, 2017).

É comum que o processo de engenharia de requisitos seja dividido em desenvolvimento de requisitos e gerenciamento de requisitos, onde o desenvolvimento de requisitos se divide em quatro grandes tópicos, sendo eles: elicitação, análise, especificação e validação (SILVA; BARRETO, 2011).

A fase de elicitação de requisitos é de suma importância para o sucesso das etapas seguintes, pois é nela que os requisitos são extraídos. Durante a análise são revolvidos conflitos e interesses do cliente, além de pontuar nuances e inconsistências nos requisitos levantados durante a elicitação. A especificação visa diferenciar e gerar uma documentação detalhada sobre os requisitos. Já a validação procura comprovar a necessidade dos requisitos levantados para o sistema (SILVA; BARRETO, 2011).

É evidente a necessidade de uma elicitação bem feita, com requisitos coerentes e bem escritos para o sucesso das demais fases, uma vez que é a base das demais etapas.

2.3.1 Elicitação de Requisitos

A elicitação de requisitos é uma das etapas da engenharia de requisitos, onde é necessário ter uma compreensão sólida sobre os desejos e necessidades do software que será implementado (PACHECO; GARCÍA; REYES, 2018), de modo que os desenvolvedores possam construir um sistema que atenda as necessidades estabelecidas (ALDAVE et al., 2019).

Uma das principais dificuldades da elicitação de requisitos é entender de forma consistente o que o cliente deseja, pois nem sempre os seus desejos são claros ou realistas, isso porque na maior parte das vezes a interação com os clientes é realizada mediante linguagem natural, aumentando a probabilidade do cliente fornecer ideias incompletas ou ambíguas (ALDAVE et al., 2019). Porém, existem técnicas que podem diminuir a ocorrência de problemas como este.

Existem diversos métodos para eliciar requisitos de software, cada um se adequando melhor para cada caso. Entre os métodos disponíveis, destacam-se os seguintes (SILVA; BARRETO, 2011):

- Métodos observacionais: utiliza a observação de qualquer atividade relevante para o entendimento do software.
- Métodos de conversação: utiliza o meio da conversação verbal entre as pessoas que necessitam do software para obter informações relevantes sobre o mesmo.
- Métodos Analíticos: geralmente utilizam deduções, documentos, e informações já existentes.

- Métodos Sintéticos: utilizam de diferentes formas de comunicação para alcançar seu objetivo, por exemplo: diagramação, prototipação, cenários, etc.

Todos esses métodos utilizam diversas técnicas para realizar o levantamento dos requisitos, por exemplo, os métodos de conversação normalmente utilizam técnicas de entrevista, *workshop*, *brainstorming*.

Já na elicitação de requisitos em métodos ágeis, a abordagem geralmente é mais interativa e iterativa, existindo uma constante comunicação entre os clientes e a equipe de desenvolvimento. Neste contexto, é comum a utilização de técnicas como: entrevistas com os clientes para entender as necessidades do sistema (BANO et al., 2019) e histórias de usuário para escrever e entender os requisitos na perspectiva do usuário (MENZINSKY et al., 2018).

2.3.2 Engenharia de Requisitos no Contexto Ágil

A elicitação de requisitos na metodologia ágil mistura dois conceitos importantes, sendo eles: a metodologia ágil e a engenharia de requisitos, mais especificamente a etapa de elicitação de requisitos.

2.3.2.1 Metodologia Ágil

A fonte da metodologia ágil é o Manifesto Ágil, também conhecido como Manifesto para Desenvolvimento Ágil de Software. O Manifesto Ágil propõe quatro valores fundamentais, onde (MANIFESTO..., 2001):

- Os indivíduos e as interações são mais valorizados do que os processos e ferramentas.
- O software em funcionamento mais que documentações abrangentes.
- As colaborações com o cliente mais que negociações de contratos.
- Responder a mudanças mais do que seguir um plano.

Além desses quatro valores fundamentais, o grupo fundador do manifesto ágil também criou doze princípios. Os doze princípios são uma extensão dos quatro valores fundamentais, que fornece orientações mais detalhadas sobre como aplicar esses valores na prática.

Por mais que as técnicas ágeis possuam uma vasta possibilidade de implementação, em sua maior parte seguem os mesmos princípios declarados no manifesto ágil.

Os métodos ágeis visam entregar softwares de alta qualidade dentro do prazo estipulado, ainda que com mudanças de requisitos no decorrer do desenvolvimento. O

principal benefício do desenvolvimento de um software utilizando metodologias ágeis é que o processo torna-se mais adaptativo ao longo do desenvolvimento, permitindo reagir e lidar com mudanças nos requisitos e especificações, independente da etapa do desenvolvimento (LUCIA; QUSEF et al., 2010).

Entre os métodos ágeis, alguns se destacaram ao longo do tempo, entre eles: Scrum, eXtremming Programming e o Lean Software Development.

Lean Software Development

O Lean Software Development é uma metodologia baseada nos princípios da indústria automobilística japonesa, especialmente da Toyota (POPPENDIECK, 2007). O principal objetivo do Lean é otimizar o processo de desenvolvimento, eliminando desperdícios, maximizando o valor entregue e promovendo uma cultura de evolução. O Lean também possui alguns princípios, sendo eles: otimização do todo, eliminação de desperdício, qualidade de construção, aprendizado constante, entregas rápidas, envolver todos e evolução continua (POPPENDIECK; CUSUMANO, 2012).

Extreme Programming (XP)

O Extreme Programming (XP) é uma metodologia voltada para equipes de pequeno e médio porte, para desenvolver softwares independentemente de sua variabilidade nos requisitos ao longo do desenvolvimento (SOARES, 2004). A metodologia Extreme Programming reuni e alinha um conjunto de práticas que já existiam, ou seja, nenhuma foi criada de fato pelo XP, mas o conjunto delas deu a origem a essa metodologia. Algumas das práticas do XP se concentram em (BECK, 1999):

- Planejamento: a partir das estimativas fornecidas pelos desenvolvedores, o cliente pode decidir o escopo e o momento de lançamento do sistema.
- Pequenos lançamentos: o sistema não precisa estar completamente finalizado para ser colocado em produção, ou seja, são lançadas diversas versões com funcionalidades sendo incrementadas ao longo de cada lançamento.
- Metáfora: o XP utiliza uma metáfora ou um conjunto de metáforas como uma forma de definir o sistema entre o cliente e os desenvolvedores.
- Design simples: evita duplicação de códigos, etc.
- Testes: os desenvolvedores devem escrever uma base de testes.
- Programação em pares: o código deve ser feito por duas pessoas.
- Integração contínua: o novo código deve ser integrado ao sistema rapidamente. Durante essa integração o sistema deve ser reconstruído e todos os casos de testes devem ser aprovados.

- Propriedade coletiva: os desenvolvedores, sempre que possível, devem procurar melhorar o código, independente do lugar do sistema.
- Semanas de 40 horas: os desenvolvedores devem procurar trabalhar sempre 40 horas semanais, e evitar horas extras.

2.3.2.2 Scrum

A metodologia Scrum propõem ideias de flexibilidade, adaptabilidade e produtividade para o desenvolvimento de software. No Scrum, não existe uma técnica específica para o desenvolvimento durante a fase de implementação, seu foco está em orientar os membros da equipe sobre como colaborar para produzir um sistema flexível em um cenário de mudanças continuas (FRANCO, 2007).

O Scrum alcança esses objetivos através de uma estrutura iterativa e incremental, da seguinte forma: no começo de cada iteração, a equipe avalia as tarefas e escolhe aquelas que consideram capazes de agregar valor ao produto até o final do ciclo. Em seguida, a equipe dedica o seu tempo para desenvolver os entregáveis definidos para a iteração, apresentando ao final um incremento de funcionalidades para que os clientes possam revisar e solicitar modificações, caso necessário (LIBARDI; BARBOSA, 2010).

Cada método possui formas diferentes para gerenciar as histórias de usuário, porém, somente o Scrum especifica práticas para organizar histórias de usuário, e possui artefatos desanexados para cada iteração (HERDIKA; BUDIARDJO, 2020).

Hana Rizky Herdika e Eko Budiardjo, encontraram nove práticas principais relacionadas ao desenvolvimento de requisitos dentro do Scrum, sendo eles: história de usuário, backlog do produto, priorização, backlog da sprint, preparação do backlog, planejamento da sprint, divisão de itens, estimativa e revisão da *sprint*. A equipe scrum organiza as histórias de usuário em épicos para rastrear e organizar os recursos em granularidades diferentes (HERDIKA; BUDIARDJO, 2020).

Product Owner (PO)

O PO representa os interesses do cliente no projeto, ele é encarregado de definir os requisitos do sistema para gerar o *backlog* do produto (LIBARDI; BARBOSA, 2010) com as partes interessadas, criando, refinando, estimando e priorizando os itens levantados (HERDIKA; BUDIARDJO, 2020).

Backlog

Um *backlog* é uma lista de tarefas, como, histórias de usuários, bugs, etc, usado pela equipe para se organizar quanto ao que deve ser feito (SEDANO; RALPH; PÉRAIRE, 2019). Logo, o Backlog do Produto é uma lista com todas as funcionalidades desejadas para o produto (LIBARDI; BARBOSA, 2010). Já o *backlog* da *sprint*, é formado por um

conjunto de itens extraídos do *backlog* do produto, que serão incorporados como incrementos ao produto (LIBARDI; BARBOSA, 2010). Geralmente os *backlogs* são ordenados por ordem de prioridade, quando maior a prioridade do item no *backlog*, mais urgente ele é, mais foi pensado e mais consenso existe em relação ao seu valor.

2.3.2.3 História de Usuário

A criação de requisitos a partir de histórias de usuário é uma abordagem muito utilizada na metodologia ágil, pois diferente dos métodos tradicionais da engenharia de requisitos, não são exigidas especificações muito abrangentes (O’HEOCHA; CONBOY, 2010). Uma história de usuário é uma especificação semi estruturada de requisitos escritos em linguagem natural na perspectiva de um usuário (RAHARJANA; SIAHAAN; FATICAH, 2021), é utilizada para auxiliar na compreensão de algo que irá ser feito, trazendo em sua estrutura informações acerca de especificidades de uma forma simples. A história de usuário é dividida em três aspectos, quem, que e porque (WAUTELET et al., 2014), “quem” deseja a funcionalidade, “que” funcionalidade os usuários finais ou partes interessadas desejam que o sistema forneça e o motivo “porque” os usuários finais ou partes interessadas precisam do sistema (WAUTELET et al., 2014).

As histórias de usuário são fortemente defendidas pelas metodologias ágeis, pois são simples, facilitam o planejamento e entendimento e incentivam um diálogo rico entre os clientes e a equipe técnica, para que o sistema reflita o que o usuário gostaria que o sistema faça e não como deveria fazê-lo (O’HEOCHA; CONBOY, 2010).

Uma história de usuário é normalmente expressa com uma frase seguindo o padrão (POKHAREL; VAIDYA, 2020):

Como <persona>, quero <objetivo desejado> para que <valor a ser alcançado>.

- Persona: Indica o tipo de usuário.
- Objetivo desejado: funcionalidade que o usuário deseja.
- Valor a ser alcançado: motivo para ter tal funcionalidade.

Outro ponto importante são os critérios de aceitação de uma história de usuário, os critérios de aceitação são condições que devem ser satisfeitas para que a história seja considerada completa. Muitas vezes esses critérios são utilizados para definir explicitamente o que é esperado da funcionalidade, auxiliando o desenvolvedor no que deve ser construído.

Também existem dificuldades ao eliciar requisitos utilizando histórias de usuários, como (SOARES et al., 2015):

- Falta de informação: o baixo nível de detalhes traz dificuldades para outras atividades durante o desenvolvimento.
- Dependência entre requisitos: na teoria as histórias de usuários podem ser implementadas de forma independente, sem levar em conta suas dependências, caso tenha, porém, na prática, é difícil desenvolver com tal sobreposição.
- Comunicação e colaboração com usuários: nem sempre a comunicação e colaboração do usuário é fácil de ser realizada, trazendo consequências negativas para o fluxo do projeto.
- Validação de requisitos: por conta do baixo nível de detalhamento nas histórias, a validação dos requisitos pode ser prejudicada.

2.4 Trabalhos Relacionados

2.4.1 User stories collection via interactive chatbot to support requirements gathering

O artigo “User stories collection via interactive chatbot to support requirements gathering” (DWITAMA; RUSLI, 2020) propõe o desenvolvimento de um sistema com *chatbot* destinado à criação de histórias de usuário, o sistema é projetado com dois tipos de usuários, os usuários comuns, que interagem com o *chatbot*, e os desenvolvedores, que além do acesso às funções de um usuário comum, também tem acesso completo a todas as respostas. Utilizando a linguagem AIML para a base de conhecimento do *chatbot*, o processo de pré-processamento de texto é explicado, como a normalização do texto recebido pelos usuários e verificação ortográfica.

O artigo também discute a definição de histórias de usuário e o modelo adotado na implementação. Os resultados do uso do *chatbot* são examinados através de análises de métricas, para avaliar seu desempenho, também foi realizado um teste de aceitação para identificar as atitudes dos usuários em relação ao aplicativo, sua utilidade percebida, facilidade e intenção de uso. Também são apontadas possíveis melhorias futuras, como a implementação de métodos mais avançados de processamento de linguagem natural e o desenvolvimento de um painel *web* para a gestão mais eficiente do *chatbot* por parte dos desenvolvedores.

2.4.2 Intelligent Chatbot for Requirements Elicitation and Classification

O artigo “Intelligent Chatbot for Requirements Elicitation and Classification” (SURANA et al., 2019) ressalta a importância e essência da elicitação de requisitos, pois tem

um grande impacto no sucesso das fases posteriores e no desenvolvimento de um projeto em geral. Ele também lista alguns problemas que ocorrem nesta fase, como a baixa qualidade nos requisitos, ambiguidades, requisitos incompletos e inconsistências. O artigo também diz que um fator determinante na qualidade dos requisitos é a experiência dos engenheiros e analistas de requisitos envolvidos, assim como seu estado mental, humor e saúde.

Uma solução proposta pelos autores é a substituição do método manual de elicitação de requisitos, que geralmente é muito suscetível a erros, por um *chatbot*, uma abordagem automatizada para obter os requisitos por meio de uma conversa com o usuário. O *chatbot* proposto utilizaria compreensão de linguagem natural, geração de linguagem natural, pré-processamento de texto e outros algoritmos de aprendizado de máquina para entender e tomar decisão a partir das entradas do usuário. A ideia consiste na utilização da linguagem Python com o *framework* Rasa e algumas outras ferramentas. Sendo assim, o artigo se tornou uma das referências para este trabalho de como um *chatbot* poderia ser utilizado de forma eficiente na engenharia de requisitos, assim como, reforçando a ideia de que ferramentas para eliciar requisitos são de grande relevância durante o processo.

2.4.3 Framework for automating requirement elicitation using a chatbot

O artigo “Framework for automating requirement elicitation using a chatbot” (LAM; HURTADO; PORTILLO, 2022) propõem um *framework* para automatizar a fase de elicitação de requisitos, validando-o com um sistema que consome um serviço de *chatbot* fornecido pela IBM Watson Assistant (IWA). Segundo o artigo, apenas uma pequena parte dos projetos são completamente bem-sucedidos, isso por conta da falta de recursos, planejamento e requisitos de baixa qualidade. Desta forma, visando melhorar e automatizar a etapa de elicitação de requisitos, o framework dispõe de um *chatbot* que por meio de uma conversa estruturada com o usuário realiza a extração de histórias de usuário, levando em conta as intenções e entidades obtidas. O tópico “desenho de entrevista estruturada” foi uma das fontes de estudos para ter uma visão melhor sobre o diálogo entre o usuário e o *chatbot* deste trabalho, pois apresenta um cenário com uma pseudo-conversa entre o *chatbot* e usuário relevante para compreender melhor como funcionaria tal diálogo na “realidade”.

2.4.4 LadderBot: A Requirements Self-Elicitation System

O artigo referente ao LadderBot (RIETZ; MAEDCHE, 2019) apresenta a ideia de um *chatbot* voltado para a elicitação de requisitos, pensado para simular um entrevistador profissional que guia os usuários durante o processo de elicitação de requisitos, os ajudando a descrever o que desejam mediante uma série de perguntas. O artigo apresenta as vantagens da simulação de um entrevistador, como a alta disponibilidade do serviço para

uma abundância de usuários, independente do horário ou localização destes. Também é abordado a dificuldade de guiar pessoas que não estão acostumadas com o processo de elicitação de requisitos, já que estes não estão familiarizados em comunicar as suas necessidades com uma forma de abstração adequada para a atividade de elicitação, e que é necessário levar em consideração esse tipo de perfil no desenvolvimento do *chatbot*.

O LadderBot funciona de forma que o usuário é cumprimentado inicialmente e recebe uma breve explicação do processo de entrevista, durante a entrevista o *chatbot* solicita ao usuário que identifique os recursos mais importantes do sistema, e durante esse processo utiliza técnicas de reafirmação para ajudá-lo a melhorar suas respostas. Para avaliar a eficácia do LadderBot, o artigo cita o planejamento de um experimento com estudantes universitários, onde estes serão solicitados a realizar uma série de atividades e com base nos resultados será possível comparar o LadderBot contra o uso de um questionário.

3 Proposta do Sistema

Neste capítulo, será apresentada a proposta de desenvolvimento de um *chatbot* voltado para o processo de criação de histórias de usuário. Este capítulo será dividido em quatro seções, descrevendo o contexto da proposta, requisitos, arquitetura do sistema e a visão geral das tecnologias e ferramentas utilizadas.

3.1 Contextualização

A elicitação de requisitos é uma atividade de extrema importância para o desenvolvimento de *software* (HOFMANN; LEHNER, 2001). Nesse contexto, o uso de histórias de usuário para a elicitação de requisitos pode ajudar a garantir que as necessidades dos usuários sejam compreendidas e integradas ao processo de desenvolvimento (ZACHARIAS et al., 2019). Porém, desenvolvedores podem enfrentar dificuldades durante o processo de criação dessas histórias, devido a diversos fatores, por exemplo, a falta de experiência nessa área de atuação, levando ao desenvolvimento de histórias de usuário mal escritas (LUCASSEN et al., 2016).

Histórias de usuário mal escritas podem causar problemas que impactam diretamente o sucesso de um projeto, como o gasto de tempo na implementação de algo de forma errônea, gerando retrabalhos e atraso em entregas, nesse contexto, a utilização de um *chatbot* voltado para auxiliar a criação de histórias de usuário pode contribuir para um desenvolvimento mais ágil e certeiro em relação às necessidades do cliente, trazer economias de tempo e custo.

O *chatbot* desenvolvido fornece recursos para auxiliar o usuário na criação de histórias de usuário, assim como a avaliação delas. Ou seja, após o levantamento de uma história, o *chatbot* retorna ao usuário uma avaliação e caso seja encontrado algum erro ou incongruência, é sugerido uma nova versão do tópico (título da história ou critério de aceitação) com a correção aplicada.

3.2 Requisitos

Neste capítulo, apresentamos a Tabela 1, que define as histórias de usuário necessárias para a construção do HuBot. Organizando e detalhando os requisitos principais que devem ser considerados para o desenvolvimento e a avaliação do *chatbot*.

As HUs são agrupadas em duas categorias principais:

- **CHATBOT:** HUs que focam na funcionalidade e características do *chatbot*, incluindo a definição de atores, a importância das funcionalidades da HU que está sendo criada com o auxílio do HuBot, a clareza dos objetivos e a definição de critérios de aceitação. Também aborda a necessidade de informações adicionais sobre histórias de usuário e a capacidade do *chatbot* para suportar ações específicas.
- **AVALIAÇÃO:** Requisitos relacionados ao processo de avaliação das histórias de usuário, incluindo a necessidade de *feedback* sobre a qualidade e a formulação da HU, bem como sugestões para aprimoramento.

Tabela 1 – Requisitos

Épico	História de Usuário
CHATBOT	Eu como usuário, gostaria de especificar qual o ator para a HU, para que fique claro quem está envolvido na história de usuário.
	Eu como usuário, gostaria de especificar qual o motivo da funcionalidade ser implementada, para que se entenda a importância e o propósito da funcionalidade.
	Eu como usuário, gostaria de dizer o objetivo da HU, para que se compreenda claramente o que se deseja alcançar com essa história de usuário.
	Eu como usuário, gostaria de associar critérios de aceitação à HU, para que se tenha uma referência clara de quando a funcionalidade estará completa e atendendo às necessidades.
	Eu como usuário, gostaria de saber mais sobre HUs, para que eu possa criar e entender histórias de usuário de forma mais eficaz.
	Eu como usuário, gostaria de saber quais as ações que o chatbot suporta, para que eu possa utilizar o chatbot de forma eficiente e tirar o máximo proveito de suas funcionalidades.
AVALIAÇÃO	Eu como usuário, gostaria de obter a versão final da HU em markdown, para que eu possa integrá-la facilmente em minha documentação ou em sistemas de controle de versão.
	Eu como usuário, gostaria de receber uma avaliação referente a HU criada, para que eu possa entender se ela está bem formulada e atende aos requisitos necessários.
	Eu como usuário, gostaria de obter sugestões referentes à minha HU, para que eu possa melhorá-la e garantir que ela cumpra seu objetivo de maneira mais eficaz.

Fonte: o autor

3.3 Arquitetura

A arquitetura proposta para realizar o levantamento e análise das histórias de usuário é composta por dois componentes principais que interagem entre si. A Figura 1 apresenta uma visão geral de como tais interações ocorrem.

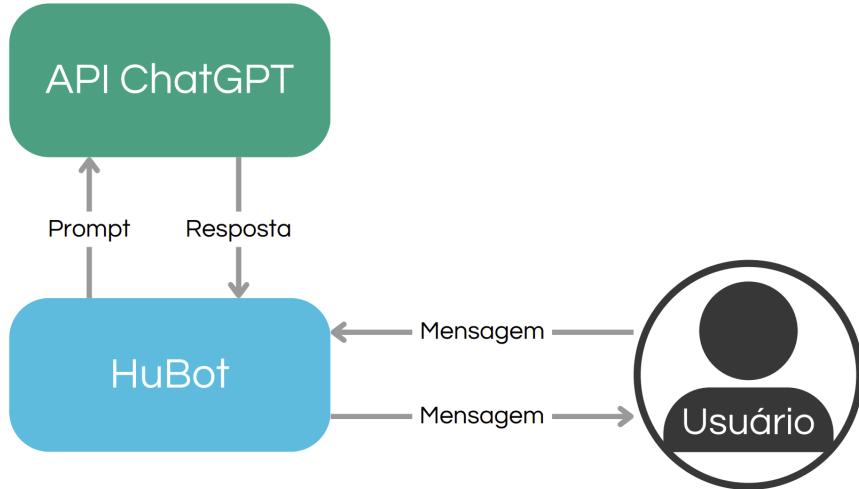


Figura 1 – Arquitetura da Solução

Fonte: o autor

Visto a complexidade para se avaliar uma história de usuário, optou-se por utilizar outro LLM para auxiliar exclusivamente na análise e sugestões de melhorias das HUs levantadas pelo usuário. O ChatGPT é um *chatbot* disponibilizado pela OpenAI que utiliza modelos de geração de texto, muita das vezes chamado de Generative Pre-trained Transformer (GPT), para gerar respostas de acordo com um ou mais prompts recebidos (OpenAI, 2024). Por possuir um excelente processamento de linguagem natural (NLP), ser treinado por uma vasta quantidade de dados (PAREDES; MACHUCA; CLAUDIO, 2023) e possuir um bom suporte e integração com a linguagem Python (mesma usada no Rasa), o ChatGPT foi escolhido como o LLM que auxilia na análise das HUs.

O HuBot utiliza o *framework* Rasa Open Source para o processamento da linguagem natural e gerenciamento dos diálogos com o usuário, além de ser utilizado como intermediário entre o usuário e o ChatGPT no momento da análise das HUs. A escolha do Rasa Open Source foi feita pelo *framework* ser bem estabelecido na comunidade de desenvolvimento de *chatbots*, além de ser um projeto de código aberto, proporcionando transparência e personalização de suas funcionalidades, além disso ele facilita a tarefa de processamento de linguagem natural e gerenciamento de diálogos, sendo um *framework* viável no contexto do projeto.

Por fim, o fluxo da solução de maneira geral pode ser definido da seguinte forma: o usuário interage com o HuBot através de mensagens de texto até que a história de usuário

esteja pronta para avaliação, então o HuBot estrutura a HU levantada em um *prompt* e o envia para o ChatGPT. Após isso, o ChatGPT retorna uma avaliação da HU levantada pelo usuário com seus possíveis pontos de melhorias e correções, então o usuário confirma se deseja aderir ou não às sugestões levantadas. Ou seja, o fluxo básico dividido em etapas:

1. **Interação Inicial:** O usuário interage com o HuBot via mensagens de texto.
2. **Preparação da HU:** O usuário desenvolve a história de usuário (HU) até que ela esteja pronta para avaliação.
3. **Estruturação do Prompt:** O HuBot estrutura a HU em um *prompt* com uma estrutura pré-definida, de acordo com os resultados obtidos no Capítulo 4.
4. **Envio para ChatGPT:** O HuBot envia o *prompt* contendo a HU para a API ChatGPT.
5. **Avaliação pelo ChatGPT:** O ChatGPT avalia a HU e fornece possíveis pontos de melhorias e correções.
6. **Retorno ao Usuário:** O HuBot retorna a avaliação do ChatGPT ao usuário.
7. **Confirmação do Usuário:** O usuário decide se deseja aderir ou não às sugestões levantadas.
8. **Retorno em Markdown:** O HuBot retorna a HU finalizada em formato de *markdown*.

3.4 Visão geral das tecnologias e ferramentas utilizadas

A seleção das tecnologias e ferramentas utilizadas foi guiada por dois fatores principais. O primeiro foi a preferência por soluções *open source*, visando apoiar e valorizar desenvolvedores que disponibilizam ferramentas de forma gratuita. O segundo fator foi a experiência prévia dos desenvolvedores do HuBot. Por exemplo, ambos possuíam uma experiência considerável com Python, o que influenciou a escolha do Rasa, um *framework* baseado nessa linguagem.

3.4.1 Git

Git, criado em 2005 por Linus Torvalds, começou apenas como um sistema de gerenciamento de revisão utilizado para o desenvolvimento do *kernel* Linux (SPINELLIS, 2012), mas se tornou popular com o tempo dentro da indústria de desenvolvimento de software. O Git é um sistema de controle de versão distribuído, diferente de sistemas centralizados, que possuem um único repositório que armazena todas as versões de um

projeto, o Git permite que cada desenvolvedor possua uma cópia do repositório em sua máquina, possibilitando que alterações sejam feitas localmente antes de enviar as atualizações para um servidor, permitindo que diversos desenvolvedores trabalhem em uma versão separada, e mesclem posteriormente o código.

O processo de trabalho simultâneo com diversos colaboradores pode contar com a ramificação (GIT..., 2024) do código, permitindo que as alterações feitas por alguém estejam em uma versão independente, criando a possibilidade que essas sejam revisadas por outros desenvolvedores e então mescladas ao código principal apenas com a aprovação destes, além disso, a ferramenta permite o rastreamento de todas as alterações feitas nos arquivos do projeto, permitindo que outros colaboradores vejam alterações implementadas por outras pessoas, essas que contam também com uma mensagem deixada pelo autor da alteração, chamada de *commit*.

3.4.2 GitHub

O GitHub é uma plataforma de hospedagem de código-fonte baseada em nuvem, construída utilizando o sistema de controle de versão Git (GITHUB..., 2024), ele permite que desenvolvedores hospedem, exponham e utilizem as ferramentas disponíveis em repositórios contendo código-fonte (GITHUB..., 2024), permitindo o desenvolvimento colaborativo, gerenciamento e controle de versão de *software*. A plataforma também disponibiliza ferramentas relacionadas ao desenvolvimento, testes, análise e deploy de aplicações. Com o GitHub Actions, é possível configurar diversos fluxos automáticos que são disparados a partir de uma ação específica escolhida pelo mantenedor do projeto.

3.4.3 Python (Versão 3.10)

O Python é uma linguagem de programação interpretada, interativa e orientada a objetos. O Python possui suporte a vários paradigmas de programação além da programação orientada a objetos, tal como programação procedural e funcional, sendo assim, uma linguagem multi-paradigma (2001 Python Software Foundation, 2025). Python é amplamente utilizado em aplicações *web*, automação de processos, ciência de dados e *machine learning* (Amazon Web Services, Inc., 2025).

3.4.4 Rasa Open Source (Versão 3.6)

O Rasa Open Source é um *framework* de código aberto desenvolvido em Python para a criação de *chatbots*. Ele utiliza técnicas de *machine learning* para compreender e responder de forma eficaz às interações dos usuários. Além disso, o Rasa permite integração com diversos canais de mensagens e sistemas de terceiros por meio de um conjunto

robusto de APIs, oferecendo flexibilidade e escalabilidade no desenvolvimento de *bots* ([Rasa Technologies GmbH, 2025](#)).

3.4.5 ChatGPT (Versão 4)

É um *chatbot* desenvolvido pela OpenIA, lançado em 2022, que utiliza inteligência artificial (IA) para o processamento e geração de linguagem natural de forma interativa.

3.4.6 Nginx

O nginx é um servidor web HTTP, *proxy* reverso, cache de conteúdo,平衡ador de carga, servidor *proxy* TCP/UDP e servidor de e-mail ([Nginx, Inc., 2024](#)).

4 Geração do Prompt

Neste capítulo, será apresentado o processo de criação do *prompt*, para a avaliação de histórias de usuários utilizando a API do ChatGPT no contexto do HuBot. Descrevendo as tentativas realizadas para chegar à versão final do *prompt*, discutindo cada tentativa, ajustes feitos ao longo do desenvolvimento e técnicas utilizadas.

4.1 Fundamentação e Definição da Solução

Para criar o HuBot, foi realizado um levantamento de como deveria ser a sequência ideal de passos para a elicição de uma história de usuário. Após o entendimento disso, foi necessário buscar uma forma de avaliar as histórias levantadas, chegando a conclusão de que utilizar a API do ChatGPT como um meio de avaliação das HUs seria viável, desde que fosse utilizado um bom *prompt*. Utilizando técnicas de *prompt engineering*, foi possível gerar avaliações satisfatórias e sugerir melhorias significativas com o ChatGPT.

O *prompt* ideal para a avaliação de uma história de usuário no contexto do sistema deve ser capaz de instruir o ChatGPT a retornar a HU na mesma estrutura em que foi enviada, mantendo a formatação original do usuário, ou seja, a resposta do ChatGPT deve alterar apenas os tópicos para os quais haja sugestões de melhorias ou correções, além de fornecer uma lista das alterações realizadas. Isso permitirá que o HuBot manipule facilmente o retorno e que o usuário receba uma avaliação clara e objetiva com sugestões bem descriptivas.

4.2 Técnicas Utilizadas

Para alcançar o *prompt* idealizado, algumas das técnicas sugeridas pela própria mantenedora do ChatGPT ([OPENAI, 2024](#)) foram aderidas, sendo as mais relevantes durante o desenvolvimento do *prompt* as seguintes:

- **Fornecer exemplos:** Para obter uma resposta coerente com o desejado, foi necessário fornecer um exemplo claro de como seria a estrutura da história a ser analisada, e também um exemplo de como deveria ser a estrutura da análise (resposta). A partir dessa premissa, as respostas seguem um padrão pré-determinado.
- **Especificando as etapas necessárias para concluir uma tarefa:** É importante para o ChatGPT entender quando a resposta está pronta. No caso, a resposta só está pronta quando a análise estiver completa, contendo as melhorias, correções (caso haja) e a visão geral da análise.

- **Indicar claramente partes distintas da entrada:** Outra técnica utilizada foi especificar bem cada parte do *prompt*, definir claramente os objetivos, afirmar instruções e esclarecer trechos que são apenas exemplo.

Por fim, a estrutura do *prompt* pode ser dividida em sete tópicos, constituídos na seguinte sequencia:

1. **Título**, tornando claro o tema do *prompt*.
2. **Instruções iniciais (objetivos)**, com padrão da HU e como devem ser feitas as correções e melhorias.
3. **Exemplos de história a ser analisada (entrada)**, com título e critérios de aceitação.
4. **Instrução declarando que a saída deve ser igual ao exemplo da sequência**, além de definir o padrão de listar as melhorias e correções.
5. **Exemplo de saída**, com análise completa, contendo a HU com suas respectivas sugestões de melhorias e correções.
6. **Instruções reforçando a ideia de como deve ser a saída**.
7. **História de usuário a ser analisada**, com a HU levantado pelo usuário (essa é a única parte do *prompt* que varia).

4.3 Tentativas

Para alcançar o *prompt* ideal para a solução proposta, foram realizadas três tentativas, sendo a terceira a mais assertiva. A seguir, cada tentativa é apresentada em duas partes: primeiro, o *prompt* utilizado, seguido por uma imagem que ilustra a saída gerada pelo ChatGPT. A análise de cada saída será discutida, destacando os ajustes e melhorias implementados em cada iteração.

Para diferenciar as entradas e saídas, utilizamos duas caixas de texto para organizar o conteúdo de forma visualmente distinta:

Prompt

Este é um exemplo de caixa de ****prompt****. Ela possui fundo azul claro e contém o *prompt* de entrada que será processado pela API do ChatGPT.

Saída

Este é um exemplo de caixa da ****saída****. Ela possui fundo verde claro e contém o resultado gerado a partir do *prompt* de entrada.

4.3.1 Prompt 1

Prompt

```
# Análise de História de Usuário
```

Análise a história de usuário a seguir, lembrando que deve estar no seguinte padrão:

Como <persona>, quero <objetivo desejado> para que <valor a ser alcançado>.

Verifique se a gramática está correta e se há possíveis incongruências nos locais de preenchimento da história de usuário. Caso encontre correções ou incongruências, recomende melhorias ou correções. Analise cada critério de aceitação e verifique se há possibilidade de melhoria ou correção. Se houver, sugira as correções necessárias, mencionando claramente o número do critério.

Exemplo:

```
## Historia de usuário (entrada)
```

Como client de um banco on-line, quero acessar meu extrato bancario de forma rápida e segura pelo aplicativo movel, para poder monitorar minhas transações financeiras a qualquer momento.

Critério(s) de Aceitação

1. O usuário deve ser capaz de fazer login com autenticação de dois fatores.
2. O extrato bancário deve ser atualizado de 10 em 10 minutos.

```
## História de Usuário (saída)
```

Como cliente de um banco online, quero acessar meu extrato bancário de forma rápida e segura pelo aplicativo móvel, para poder monitorar minhas transações financeiras a qualquer momento. [Correção gramatical na persona]

Critério(s) de Aceitação

2. O extrato bancário deve ser atualizado em tempo real [Melhoria].

História de usuário a ser analisada

Como usuário de um site de ecommerce, quero pode adicionar produtos ao meu carrinho e olhar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

Critério(s) de Aceitação

1. O usuário deve poder adicionar produtos ao carro diretamente da página do produto.
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável).
3. A prévia do carrinho deve ser acessível em qualquer página do site.

Saída

Análise da História de Usuário

História de Usuário (entrada)

Como usuário de um site de ecommerce, quero pode adicionar produtos ao meu carrinho e olhar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

Crítica e Sugestões

Correção gramatical na Persona e Objetivo:

A história de usuário está quase correta, mas há um pequeno erro gramatical e uma sugestão de melhoria no objetivo desejado para torná-lo mais claro.

Sugestão de Melhoria na História de Usuário:

- Correção do verbo "pode" para "poder".
- Reestruturação da frase para melhor clareza e fluidez.

História de Usuário (saída)

Como usuário de um site de e-commerce, quero poder adicionar produtos ao meu carrinho e visualizar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

Critérios de Aceitação

1. O usuário deve poder adicionar produtos ao carrinho diretamente da página do produto.
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável)
3. A prévia do carrinho deve ser acessível em qualquer página do site.

Crítica e Sugestões nos Critérios de Aceitação

Critério 1:

- Correção do termo "carro" para "carrinho".

Critério 2:

- Está claro e bem formulado. Sem necessidade de correção.

Critério 3:

- Está claro e bem formulado. Sem necessidade de correção.

Critérios de Aceitação (saída)

1. O usuário deve poder adicionar produtos ao carrinho diretamente da página do produto.
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável)

3. A prévia do carrinho deve ser acessível em qualquer página do site.

A análise mostra que a história de usuário e os critérios de aceitação estão bem formulados com pequenos ajustes gramaticais e de clareza necessários. Após a correção, a história de usuário cumpre o padrão desejado e os critérios de aceitação são claros e concisos.

O resultado não saiu como esperado. A resposta foi gerada com dois tópicos principais, sendo eles: História de Usuário (entrada) e História de Usuário (saída). A partir do prompt fornecido, o ChatGPT entendeu que deveria retornar História de Usuário (entrada) com os mesmos dados da história a ser analisada e História de Usuário (saída) como o resultado das correções e melhorias, onde, na verdade, o desejado era apenas o título da história e os critérios de aceitação com as sugestões de melhorias e correção, além de uma visão geral da história levantada.

4.3.2 Prompt 2

Prompt

Análise de História de Usuário

Análise a história de usuário a seguir, lembrando que ela deve estar no seguinte padrão:

Como <persona>, quero <objetivo desejado> para que <valor a ser alcançado>.

Verifique se a gramática está correta e se há possíveis incongruências nos locais de preenchimento da história de usuário. Caso encontre correções ou incongruências, recomende melhorias ou correções. Analise cada critério de aceitação e verifique se há possibilidade de melhoria ou correção. Se houver, sugira as correções necessárias, mencionando claramente o número do critério.

Exemplo de história a ser analisada:

Historia de usuário

Como client de um banco on-line, quero acessar meu extrato bancario de forma rápida e segura pelo aplicativo movel, para poder ver minhas

transações financeiras a qualquer momento.

Critério(s) de Aceitação

1. O usuário deve ser capaz de fazer login com autenticação de dois fatores.
2. O extrato bancário deve ser atualizado de 10 em 10 minutos.

Exemplo de resposta esperada, ou seja, retorna o título e os critérios de aceitação no mesmo formato, porém, com suas devidas melhorias e correções, além de mostrar entre colchetes se a alteração foi uma: melhoria, correção ou ambos.

História de Usuário (saída)

Como cliente de um banco online, quero acessar meu extrato bancário de forma rápida e segura pelo aplicativo móvel, para poder monitorar minhas transações financeiras a qualquer momento. [MELHORIA/CORREÇÃO]

Critério(s) de Aceitação

1. O usuário deve ser capaz de fazer login com autenticação de dois fatores.
2. O extrato bancário deve ser atualizado em tempo real. [MELHORIA]

Visão geral: A análise mostra que a história de usuário e os critérios de aceitação estão bem formulados com pequenos ajustes gramaticais e de clareza necessários. Após a correção, a história de usuário cumpre o padrão desejado e os critérios de aceitação são claros e concisos.

Analise essa história de usuário e responda na mesma estrutura do exemplo de resposta espera:

Historia de usuário a ser analisada

Como usuário de um site de ecommerce, quero pode adicionar produtos ao meu carrinho e olhar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

Critério(s) de Aceitação

1. O usuário deve poder adicionar produtos ao carro diretamente da página do produto.
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável).
3. A prévia do carrinho deve ser acessível em qualquer página do site.

Saída**# História de Usuário**

Como usuário de um site de ecommerce, quero poder adicionar produtos ao meu carrinho e olhar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra. [CORREÇÃO]

Critérios(s) de Aceitação

1. O usuário deve poder adicionar produtos ao carrinho diretamente da página do produto. [CORREÇÃO]
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável).
3. A prévia do carrinho deve ser acessível em qualquer página do site.

Visão geral: A análise mostra que a história de usuário e os critérios de aceitação estão bem formulados, com pequenos ajustes gramaticais necessários. Após a correção, a história de usuário cumpre o padrão e os critérios de aceitação são claros e concisos.

O resultado da segunda tentativa foi mais assertivo em relação ao que se esperava, embora os *prompts* sejam similares, a estrutura e os exemplos foram ligeiramente diferentes. A adição de algumas instruções foi essencial para chegar a um resultado mais próximo do ideal, como, por exemplo: “Analise esta história de usuário e responda na mesma estrutura do exemplo de resposta esperada”. Essa instrução deixa claro ao ChatGPT como deve ser estruturada a resposta referente à análise da HU. Outro ponto relevante para uma saída mais padronizada foi enfatizar que a resposta esperada deve incluir os termos “[MELHORIA]” e/ou “[CORREÇÃO]” caso exista sugestão de melhoria ou correção no

tópico analisado.

4.3.3 Prompt 3 - Final

Prompt

```
# Análise de História de Usuário
```

Análise a história de usuário a seguir, lembrando que ela deve estar no seguinte padrão:

Como <persona>, quero <objetivo desejado>, para que <valor a ser alcançado>.

O padrão deve ser seguido para que a análise seja feita corretamente, não altere a estrutura do padrão, não retire vírgulas ou pontos finais impostos no padrão. Verifique se a gramática e a concordância está correta e se há possíveis incongruências nos locais de preenchimento da história de usuário. Caso encontre correções ou incongruências, recomende melhorias e/ou correções. Analise cada critério de aceitação e verifique se há possibilidade de melhoria ou correção. Se houver, sugira as correções necessárias, mencionando claramente o número do critério.

Exemplo de historia a ser analisada:

```
## Historia de usuário
```

Como client de um banco on-line, quero acessar meu extrato bancario de forma rápida e segura pelo aplicativo movel, para poder ver minhas transações financeiras a qualquer momento.

Critério(s) de Aceitação

1. O usuário deve ser capaz de fazer login com autenticação de dois fatores.
2. O extrato bancário deve ser atualizado de 10 em 10 minutos.

Exemplo de resposta esperada, ou seja, retorna o titulo e os criterios de aceitação no mesmo formato, porém, com suas devidas melhorias e correções, além de mostrar mostrar entre colchetes se a alteração foi uma: melhoria, correção ou ambos. Tenho melhoria ou correção também é listado o motivo

dessas correções e melhorias.

Resposta esperada:

História de Usuário (analisada e corrigida)

Como cliente de um banco online, quero acessar meu extrato bancário de forma rápida e segura pelo aplicativo móvel, para poder monitorar minhas transações financeiras a qualquer momento. [MELHORIA/CORREÇÃO]

- client -> cliente.
- on-line -> online.
- bancario -> bancário.
- movel -> móvel.
- para poder ver minhas transações -> para poder monitorar minhas transações.

Critério(s) de Aceitação

1. O usuário deve ser capaz de fazer login com autenticação de dois fatores.
2. O extrato bancário deve ser atualizado em tempo real. [MELHORIA]
 - atualizado de 10 em 10 minutos -> atualizado em tempo real.

Visão geral: A análise mostra que a história de usuário e os critérios de aceitação estão bem formulados com pequenos ajustes gramaticais e de clareza necessários. Após a correção, a história de usuário cumpre o padrão desejado e os critérios de aceitação são claros e concisos.

Analise essa historia de usuário e responda na mesma estrutura do exemplo de resposta espera:

Historia de usuário a ser analisada

Como usuário de um site de ecommerce, quero pode adiciona produtos ao meu carrinho e olhar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

Critério(s) de Aceitação

1. O usuário deve poder adicionar produtos ao carro diretamente da página do produto.
2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável).
3. A prévia do carrinho deve ser acessível em qualquer página do site.

Saída

História de Usuário (analisada e corrigida)

Como usuário de um site de e-commerce, quero poder adicionar produtos ao meu carrinho e visualizar uma prévia do total da compra sem sair da página do produto, para facilitar e agilizar o processo de compra.

[CORREÇÃO]

- pode -> poder.
- adiciona -> adicionar.
- olhar -> visualizar.
- ecommerce -> e-commerce.
- prévia do total da compra sem sair da página do produto -> visualizar uma prévia do total da compra sem sair da página do produto.

Critério(s) de Aceitação

1. O usuário deve poder adicionar produtos ao carrinho diretamente da página do produto. [CORREÇÃO]

- carro -> carrinho.

2. Após adicionar um produto, deve ser exibida uma prévia do total da compra (incluindo impostos e frete, se aplicável).

3. A prévia do carrinho deve ser acessível em qualquer página do site. [CORREÇÃO]

- previa -> prévia.
- pagina -> página.

Visão geral: A análise mostra que a história de usuário e os critérios de aceitação estavam bem formulados com pequenos ajustes gramaticais e de clareza necessários. Após a correção, a história de usuário cumpre o padrão desejado e os critérios de aceitação são claros e concisos.

O resultado da terceira tentativa foi um sucesso, pois além de retornar exatamente a história com as sugestões de correções, melhorias e visão geral, também lista exatamente o que foi alterado em cada etapa da HU. A principal diferença entre o *prompt* da tentativa 2 e o da tentativa 3 foi a inclusão de uma instrução solicitando uma lista das alterações realizadas e demonstrando como deveria ser essa listagem nos exemplos.

5 Desenvolvimento do Sistema

5.1 Planejamento de Desenvolvimento de Software

5.1.1 Backlog do Produto

Os requisitos já foram apresentados previamente na seção 3.2. Nesta seção, eles serão reorganizados e priorizados conforme os critérios estabelecidos. Foram utilizados dois níveis de granularidades na produção do Backlog do Produto, sendo eles: Épicos e Histórias de Usuário.

Épicos

Para fins de facilitar a leitura da tabela do Backlog do Produto foi aderido a utilização de siglas (códigos) para cada um dos épicos e história de usuário.

Código	Descrição
EP01	CHATBOT
EP02	AVALIAÇÃO

Tabela 2 – Código e Descrição

Backlog do Produto Priorizado

Para realizar a priorização do backlog foi utilizada a técnica MoSCoW. Em sintaxe, a técnica MoSCoW busca priorizar os itens em quatro categorias diferentes (OLIVEIRA, 2014), sendo eles:

- Must Have: Deve ter.
- Should have: Deveria ter.
- Could have: Poderia ter.
- Won't have: Não terá agora.

5.1.2 Plano de Iterações

Para desenvolver as histórias de usuário descritas no Backlog do Produto, optou-se por utilizar *sprints* com duração de duas semanas. Foram adotados alguns rituais comumente utilizados no Scrum, como as reuniões diárias, o planejamento da sprint, a revisão da *sprint* e a retrospectiva da *sprint*.

Épico	Código	História de Usuário	Prioridade
EP01	HU01	Eu como usuário, gostaria de especificar qual o ator para a HU, para que fique claro quem está envolvido na história de usuário.	Must
EP01	HU02	Eu como usuário, gostaria de especificar qual o motivo da funcionalidade ser implementada, para que se entenda a importância e o propósito da funcionalidade.	Must
EP01	HU03	Eu como usuário, gostaria de dizer o objetivo da HU, para que se compreenda claramente o que se deseja alcançar com essa história de usuário.	Must
EP02	HU09	Eu como usuário, gostaria de obter sugestões referentes à minha HU, para que eu possa melhorá-la e garantir que ela cumpra seu objetivo de maneira mais eficaz.	Must
EP01	HU04	Eu como usuário, gostaria de associar critérios de aceitação à HU, para que se tenha uma referência clara de quando a funcionalidade estará completa e atendendo às necessidades.	Should
EP01	HU07	Eu como usuário, gostaria de obter a versão final da HU em markdown, para que eu possa integrá-la facilmente em minha documentação ou em sistemas de controle de versão.	Should
EP02	HU08	Eu como usuário, gostaria de receber uma avaliação referente a HU criada, para que eu possa entender se ela está bem formulada e atende aos requisitos necessários.	Should
EP01	HU06	Eu como usuário, gostaria de saber quais as ações que o chatbot suporta, para que eu possa utilizar o chatbot de forma eficiente e tirar o máximo proveito de suas funcionalidades.	Could
EP01	HU05	Eu como usuário, gostaria de saber mais sobre HUs, para que eu possa criar e entender histórias de usuário de forma mais eficaz.	Could

Tabela 3 – Backlog do Produto com Prioridades

Rito	Síncrono	Periodicidade	Dias em que ocorre
Daily	Não	Diário	Todo dia útil
Planejamento da Sprint	Sim	Por sprint	Toda segunda-feira após o término de uma sprint
Revisão da Sprint	Sim	Por sprint	Durante o último dia da sprint
Retrospectiva da Sprint	Sim	Por sprint	Durante o último dia da sprint

Tabela 4 – Tabela de Ritos

5.1.3 Cronograma de Desenvolvimento Simplificado

Para fins de organização e controle nas etapas de desenvolvimento, foram definidas três grandes entregas (versões). A primeira inclui as histórias de usuário com prioridade MUST HAVE pertencentes ao épico denominado CHATBOT; a segunda, as histórias com prioridade MUST HAVE do épico AVALIAÇÃO; e, por último, as histórias com prioridade SHOULD HAVE.

O período de desenvolvimento dessas releases é o seguinte:

- Release 1: 23/09/2024 - 18/10/2024
- Release 2: 21/10/2024 - 15/11/2024
- Release 3: 18/11/2024 - 23/11/2024

5.1.4 Comunicação

As plataformas utilizadas para comunicação durante o desenvolvimento do software foram as seguinte: Discord, Telegram e GitHub.

5.1.5 Gestão de Riscos

Os riscos do projeto representam potenciais obstáculos e desafios que podem surgir durante o desenvolvimento do HuBot, esses riscos podem interferir no planejamento original, impactando tanto o produto final quanto o processo de desenvolvimento. A identificação antecipada desses riscos será feita para fornecer uma visão do que possa acontecer e possíveis alternativas para minimizar seus impactos. As estimativas dos riscos foram elaboradas com base na experiência prévia da equipe.

5.1.5.1 Métricas

Para determinar a relevância de cada risco identificado adotamos um critério principal de avaliação, o impacto. O impacto refere-se ao grau de prejuízo que o projeto pode sofrer caso o risco se concretize.

Impacto	Descrição
Nenhum	Não há impacto sobre o desenvolvimento do HuBot
Baixo	Pequeno impacto para o desenvolvimento
Médio	O impacto é notável e começa a gerar consequências para o progresso do desenvolvimento
Alto	O impacto compromete de maneira significativa o andamento do desenvolvimento
Altíssimo	O impacto é grave a ponto de inviabilizar o desenvolvimento, a menos que seja devidamente mitigado

Tabela 5 – Tabela de Impacto

5.1.6 Código-Fonte

O código-fonte do projeto está disponível publicamente em um repositório do *GitHub*. Para acessar, utilize o seguinte link: <<https://github.com/TCC-chat-requisitos/HuBot>>

Risco	Descrição	Impacto	Ação Preventiva	Ação Reativa
Falhas na Integração com a API do ChatGPT	Possíveis dificuldades técnicas ao integrar a API do ChatGPT ao chatbot desenvolvido no RASA.	Alto	Realizar testes durante o desenvolvimento da integração e manter contato com a documentação oficial da API.	Revisar a integração, identificar e corrigir os erros específicos, e realizar testes adicionais para garantir a estabilidade.
Problemas de Deploy	Dificuldades ao realizar o deploy do HuBot no ambiente de produção.	Médio	Documentar e testar o deploy em ambientes de simulação, utilizar ferramentas para monitorar o serviço em deploy.	Revisar e ajustar o processo de deploy, e solicitar ajuda, se disponível.
Dependência de Ferramentas Externas	Risco de indisponibilidade ou mudanças inesperadas nas ferramentas utilizadas, como RASA ou a API do ChatGPT.	Alto	Monitorar a estabilidade e as atualizações das ferramentas externas.	Adaptar o código para versões novas ou alternativas de ferramentas e buscar soluções alternativas caso ocorra uma falha crítica.
Qualidade das Histórias de Usuário	Risco de que as histórias geradas pelo chatbot não atendam às expectativas em termos de qualidade ou relevância.	Altíssimo	Desenvolver um prompt bem definido e realizar testes contínuos para refinar os critérios de avaliação das histórias.	Ajustar o prompt e a lógica de avaliação, revisando as histórias geradas manualmente, se necessário, para garantir a qualidade.
Falta de Sincronização entre os Desenvolvedores	Risco de comunicação inadequada ou falta de alinhamento entre os dois desenvolvedores do projeto.	Médio	Realizar reuniões regulares de alinhamento, utilizar ferramentas de gestão de projetos e documentar todas as decisões.	Reavaliar as tarefas e redistribuir responsabilidades conforme necessário para corrigir problemas de comunicação ou desentendimentos.
Falta de Documentação Completa	Risco de que a documentação do projeto não esteja completa ou clara o suficiente para orientação da equipe.	Médio	Manter a documentação atualizada durante todo o desenvolvimento, com revisões periódicas.	Realizar uma revisão completa da documentação, corrigindo lacunas.
Atrasos no Cronograma	Possibilidade de atrasos nas entregas devido à complexidade das tarefas ou problemas imprevistos.	Alto	Dividir o cronograma em entregas mensuráveis, com revisões regulares de progresso.	Repriorizar tarefas, ajustar o cronograma e, se necessário, solicitar uma extensão de prazo.
Incompatibilidade com o Telegram	Risco de incompatibilidade técnica ou limitações na integração do chatbot com o Telegram.	Baixo	Testar a integração com o Telegram desde as fases iniciais do desenvolvimento para identificar problemas precocemente.	Adaptar a implementação ou explorar soluções alternativas.
Sobrecarga de Trabalho	Risco de sobrecarga para os desenvolvedores, causando exaustão e impacto na qualidade do trabalho.	Alto	Definir um cronograma realista com pausas programadas e distribuir as tarefas de maneira equilibrada.	Reavaliar a carga de trabalho e redistribuir tarefas, priorizando as atividades mais críticas.

Tabela 6 – Tabela de Riscos e Medidas

5.1.7 Implementação do Chatbot

O *chatbot* foi implantado em um ambiente de produção utilizando uma máquina virtual na *DigitalOcean*. Abaixo estão detalhados os passos realizados para configurar o

ambiente e disponibilizar o sistema:

5.1.8 Configuração da Máquina Virtual

- Foi criada uma máquina virtual na *DigitalOcean* com o sistema operacional Ubuntu.
- Foram instalados os pacotes necessários, como *Python*, *pip* e *nginx*.
- Foi configurado um ambiente virtual *Python* (*venv*) para gerenciar as dependências do projeto.

5.1.9 Registro de Subdomínio e Configuração de Certificado SSL

- Foi registrado o subdomínio *hubot.mooo.com* no serviço *FreeDNS*.
- Foi realizada a configuração de um certificado TLS/SSL gratuito utilizando o *Certbot*, garantindo comunicação segura entre cliente e servidor.

5.1.10 Configuração do Servidor Nginx

- Foram realizadas as configurações do *Nginx* para atuar como *proxy* reverso, redirecionando as requisições HTTPS para o serviço do *chatbot*.
- As configurações incluíram a integração com o certificado SSL e o redirecionamento automático de HTTP para HTTPS.

5.1.11 Execução do Chatbot

- No ambiente virtual Python, foram instaladas as dependências do projeto com:

```
pip install -r requirements.txt
```

- A inicialização do servidor RASA foi realizada com os seguintes comandos:

`rasa run actions &`
`rasa run --enable-api --cors "*" &`

5.1.12 Integração com o Telegram

- Foi criado um bot no Telegram utilizando o *BotFather*, obtendo-se o token de acesso.
- Foi configurado o *webhook* do *Telegram* para redirecionar mensagens ao *chatbot* no domínio *hubot.mooo.com*.
- O nome de usuário do bot foi configurado como *historias_de_usuario_bot*.

5.1.13 HuBot em funcionamento

O *chatbot* foi projetado para interagir de forma contínua com o usuário, respondendo a perguntas e executando tarefas baseadas nas histórias de usuário implementadas. As imagens a seguir ilustram três possíveis fluxos contínuos de interação:

- Fluxo 1: Representado na Figura 2, este fluxo apresenta o início da interação com o *chatbot*, incluindo sua introdução e o menu de ações disponíveis.
- Fluxo 2: Este fluxo, ilustrado nas Figuras 3, 4 e 5, demonstra a criação de uma história de usuário (HU), seguida pela avaliação e sugestões de melhorias fornecidas pelo HuBot. Neste caso, o usuário decidiu aceitar todas as sugestões apresentadas.
- Fluxo 3: Semelhante ao fluxo anterior, mas com uma diferença significativa: aqui, o usuário opta por adotar apenas algumas das sugestões oferecidas pelo HuBot. Este fluxo é mostrado nas Figuras 6, 7 e 8.

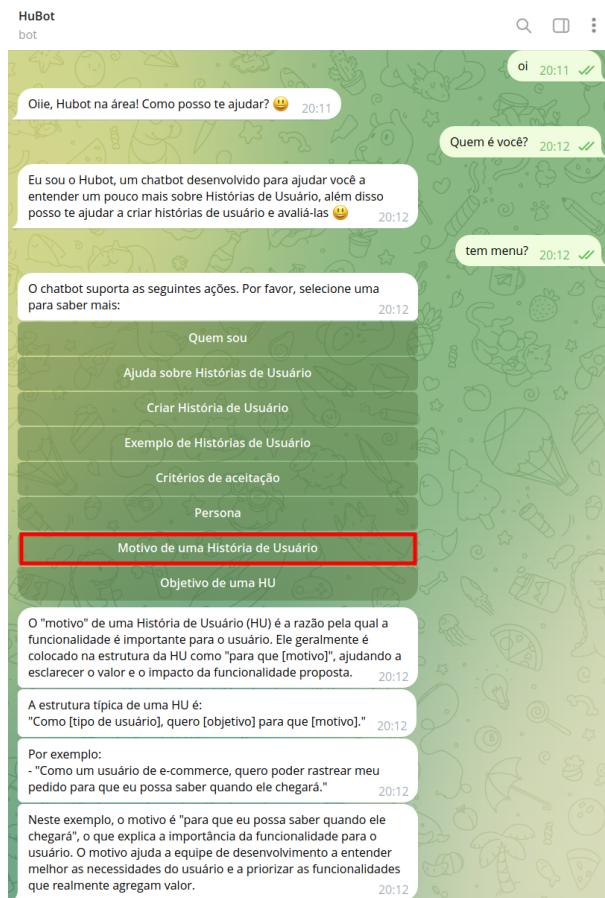


Figura 2 – Fluxo 1: Menu de ações do HuBot e seleção de opção.

Fonte: o autor

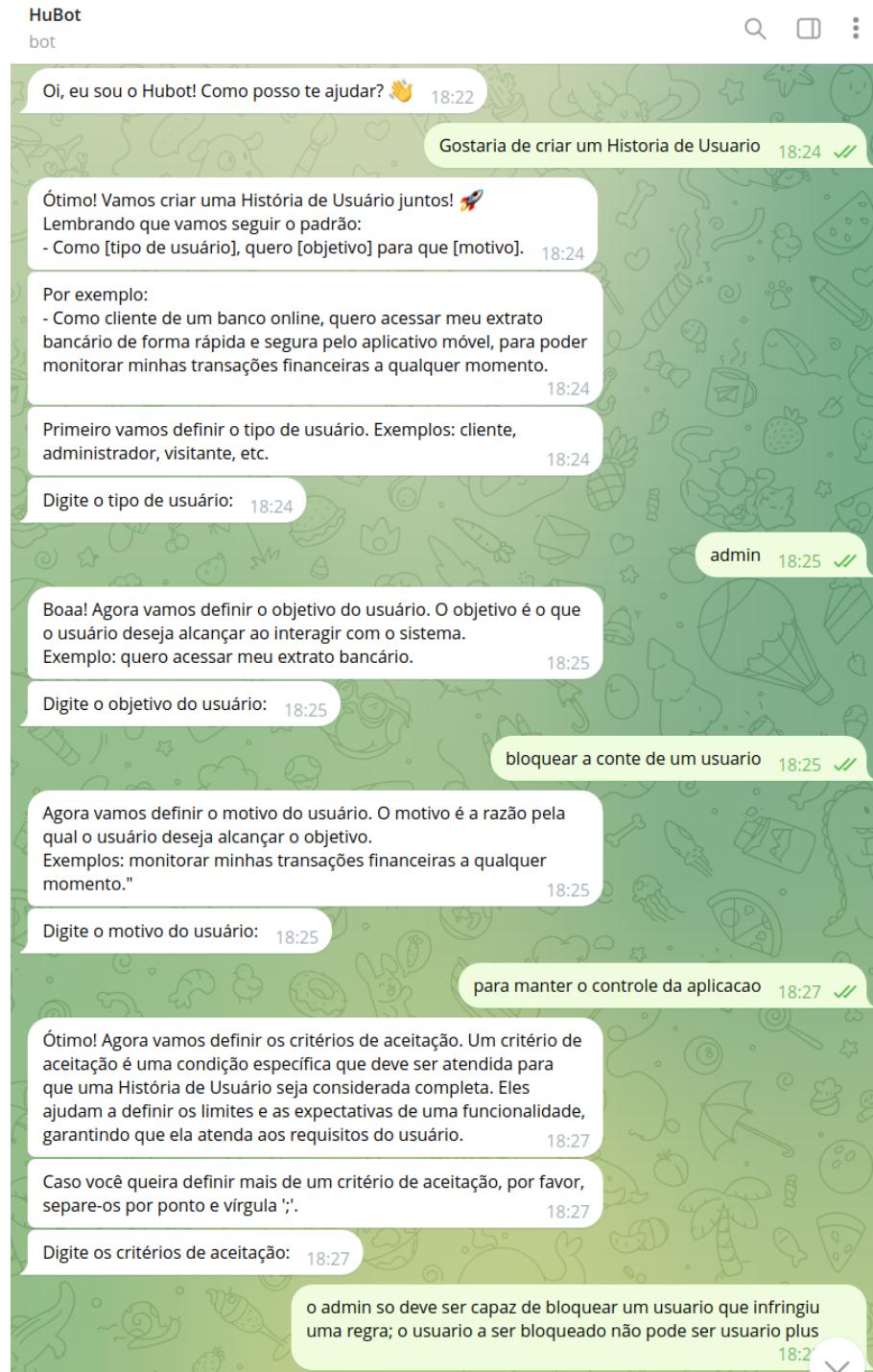


Figura 3 – Fluxo 1, pt. 1: Preenchimento do tipo de usuário, objetivo e motivo.

Fonte: o autor

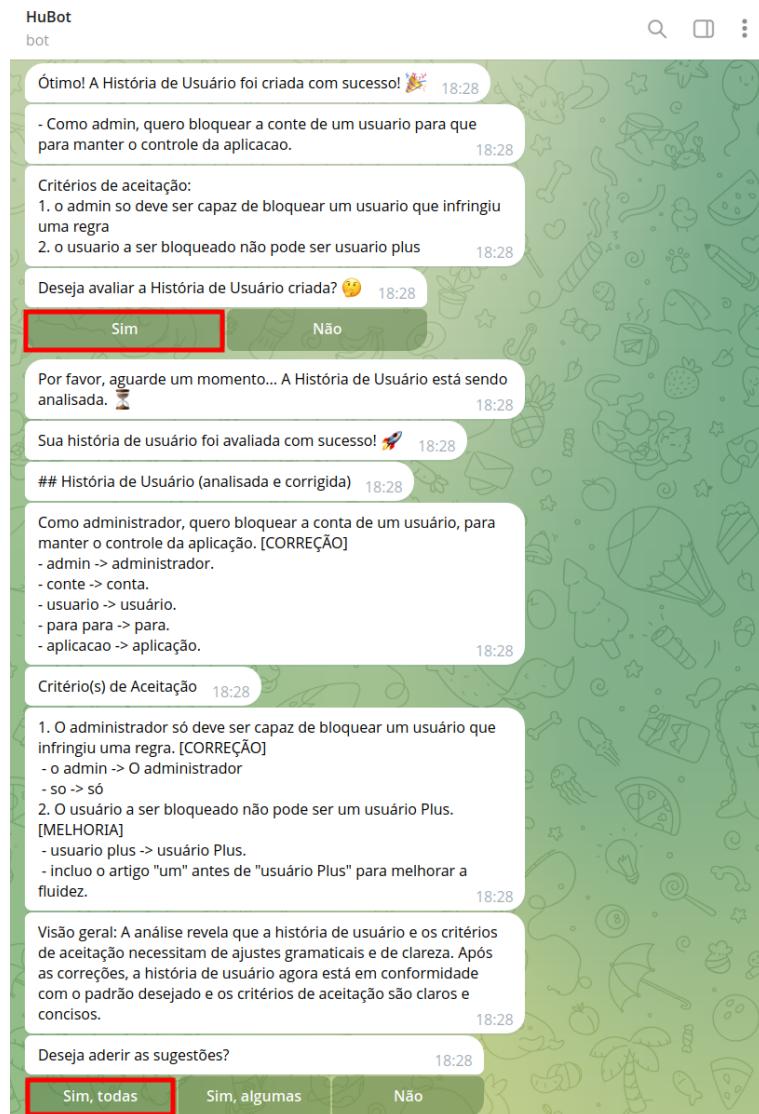


Figura 4 – Fluxo 2, pt. 2: Analise e sugestões para a HU criada.

Fonte: o autor

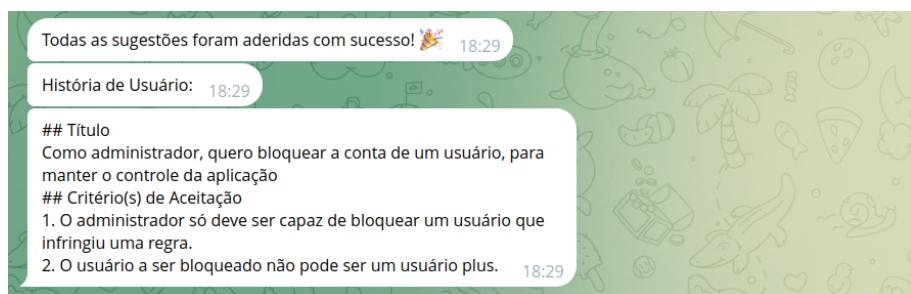


Figura 5 – Fluxo 2, pt. 3: Resultado final da HU, com melhorias aderidas.

Fonte: o autor

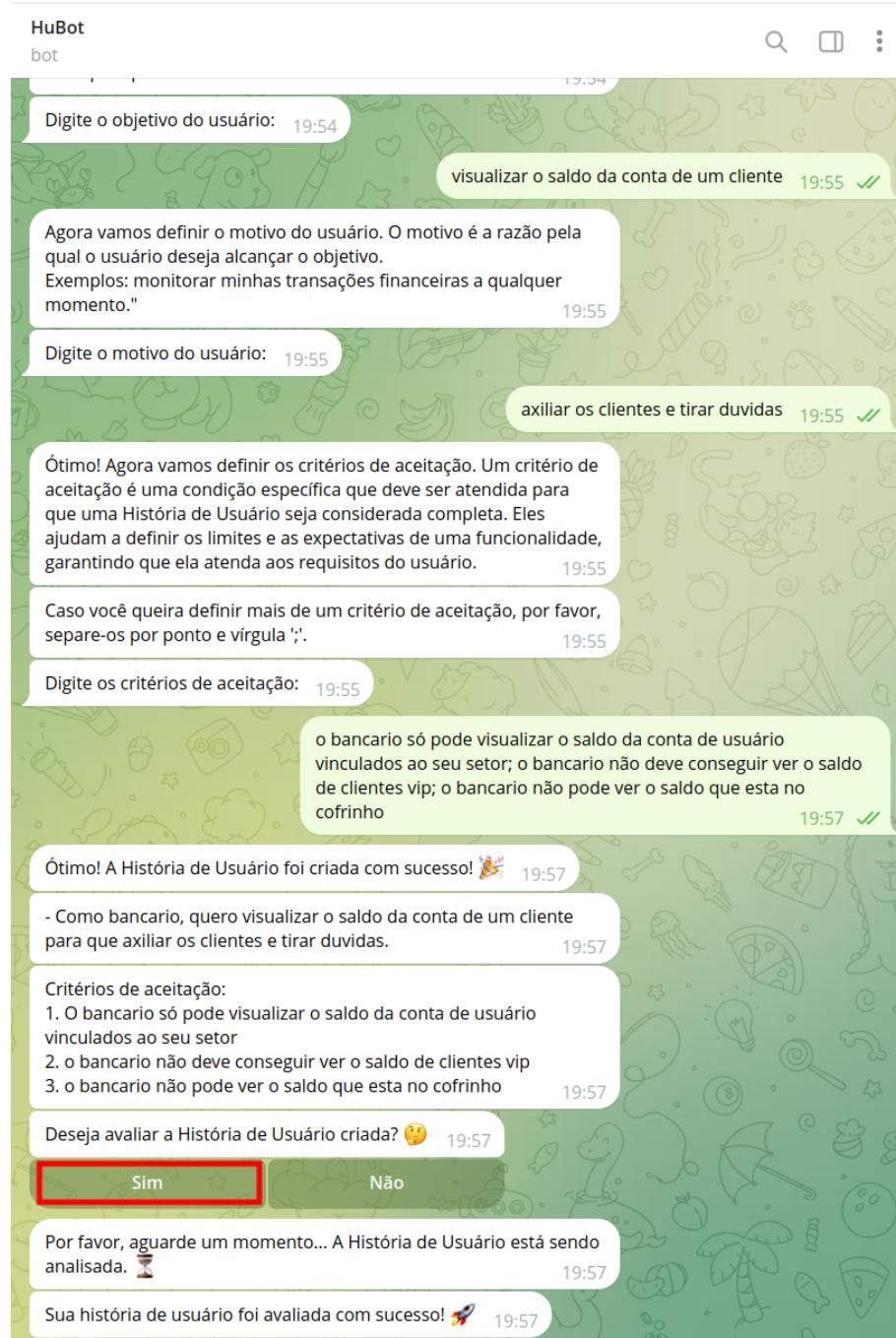


Figura 6 – Fluxo 3, pt. 1: Preenchimento do tipo de usuário, objetivo e motivo.

Fonte: o autor

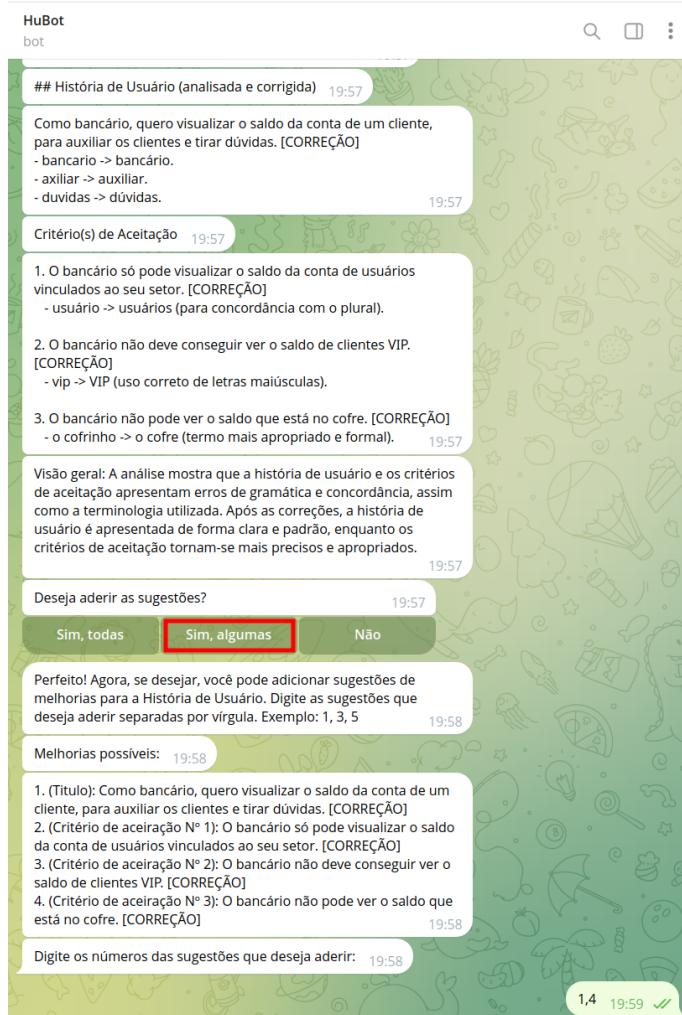


Figura 7 – Fluxo 3, pt. 2: Analise e sugestões para a HU criada.

Fonte: o autor

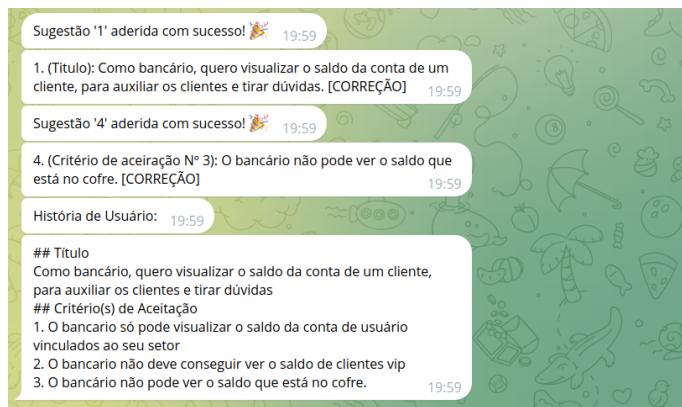


Figura 8 – Fluxo 3, pt. 3: Resultado final da HU, com algumas melhorias aderidas.

Fonte: o autor

6 Validação e Resultados

6.1 Método de Avaliação do HuBot

Para realizar a avaliação do HuBot por usuários reais foi criado um formulário utilizando o *Google Forms*. Esse formulário incluiu seções que abordaram consentimento de participação, confirmação de uso do HuBot, informações sobre os participantes, sua familiaridade com *chatbots* e a sua experiência ao utilizar o HuBot.

O formulário continha diferentes tipos de perguntas. Inicialmente, um termo de consentimento e uma confirmação do uso do HuBot eram apresentados. Caso a resposta fosse “não” para qualquer uma dessas perguntas, o participante seria automaticamente direcionado ao encerramento do formulário. As demais perguntas foram divididas entre perguntas abertas, onde os participantes podiam detalhar suas opiniões e sugestões, e perguntas estruturadas em uma escala *Likert* de cinco pontos (JOSHI et al., 2015), nas quais era necessário selecionar uma dentre cinco opções disponíveis.

A divulgação do formulário foi realizada exclusivamente online, sendo compartilhado em grupos de mensagens da faculdade e com colegas por meio de redes sociais, e-mail e aplicativos de comunicação, como o *Telegram*.

Este capítulo apresenta os resultados obtidos a partir das respostas ao formulário aplicado aos 23 participantes que testaram o HuBot, contendo as perguntas e respostas obtidas.

6.2 Formulário e Resultados

6.2.1 Confirmação de Participação e Uso do Chatbot

A primeira pergunta do formulário verificou se os participantes concordavam em participar do estudo. Todas as respostas foram positivas, conforme mostrado na Figura 9.

Sobre a confirmação de uso do chatbot, 91,3% confirmaram a interação com o HuBot, enquanto 8,7% não confirmaram. Isso pode ser visualizado na Figura 10.

A maioria significativa dos participantes interagiu com o HuBot. O pequeno grupo que não interagiu não influenciou nos resultados subsequentes, pois o formulário se encerrava caso a resposta desta pergunta fosse negativa.

Você concorda com as condições de participação?

23 respostas

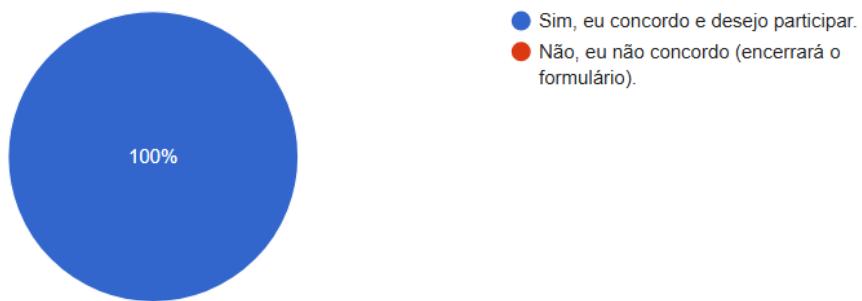


Figura 9 – Distribuição das respostas sobre a concordância com a participação no estudo.

Fonte: o autor

Você confirma que utilizou o chatbot @historias_de_usuario_bot no Telegram?

23 respostas

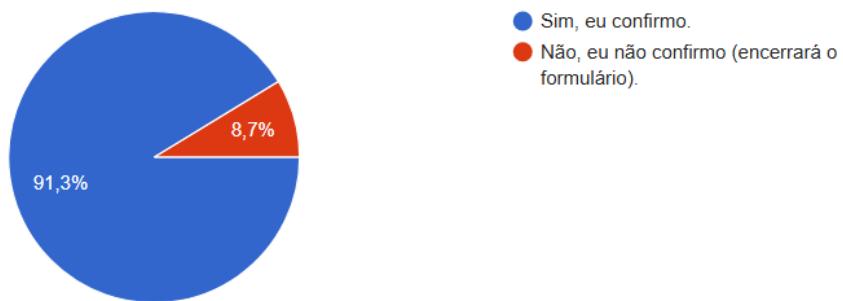


Figura 10 – Confirmação de uso do chatbot pelos participantes.

Fonte: o autor

6.2.2 Qual é a sua familiaridade com chatbots?

Os participantes foram questionados sobre sua familiaridade com *chatbots*. A Figura 11 apresenta a distribuição das respostas.

6.2.3 Qual é o seu nível de conhecimento sobre histórias de usuário?

Os participantes foram questionados sobre o nível de conhecimento que possuem sobre histórias de usuário. A Figura 12 apresenta as respostas.

Os dados indicam que a maioria dos participantes possui algum nível de conhecimento sobre histórias de usuário, sendo que 38,1% relatam saber o básico e conseguir criar histórias simples, enquanto 28,6% têm experiência intermediária e 19% possuem muita

Qual é a sua familiaridade com chatbots?

21 respostas

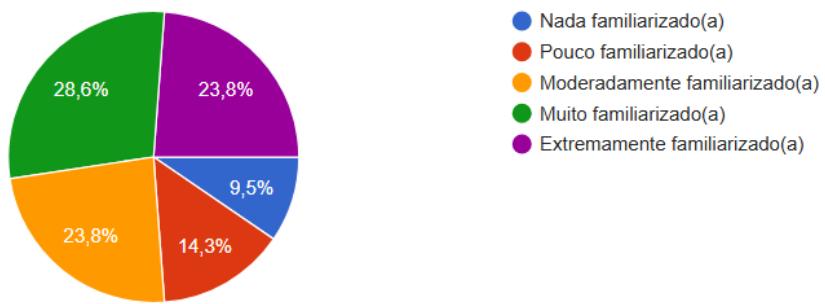


Figura 11 – Distribuição da familiaridade com chatbots.

Fonte: o autor

Qual é o seu nível de conhecimento sobre histórias de usuário?

21 respostas

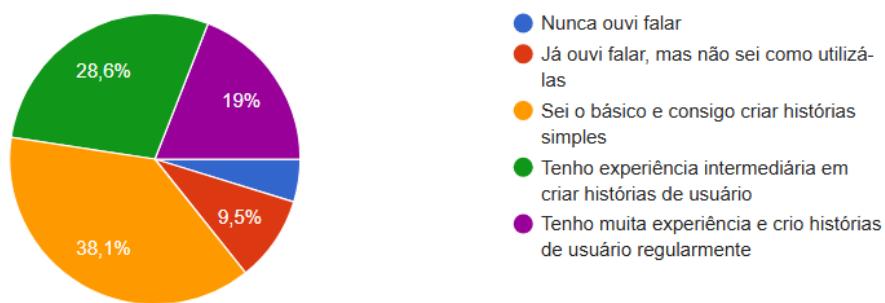


Figura 12 – Nível de conhecimento dos participantes sobre histórias de usuário.

Fonte: o autor

experiência e criam histórias regularmente.

Por outro lado, uma parcela menor dos participantes relatou nunca ter ouvido falar sobre histórias de usuário (4,8%) ou apenas ter ouvido falar sem saber utilizá-las (9,5%). Esse resultado reflete a possibilidade de utilizar o HuBot como uma ferramenta para avaliar sua utilidade entre aqueles que possuem pouco ou nenhum conhecimento na área.

6.2.4 Como você avalia sua experiência em desenvolvimento de software?

Os participantes também foram questionados sobre como avaliam sua experiência em desenvolvimento de software. A Figura 13 apresenta os resultados.

Como você avalia sua experiência em desenvolvimento de software?

21 respostas

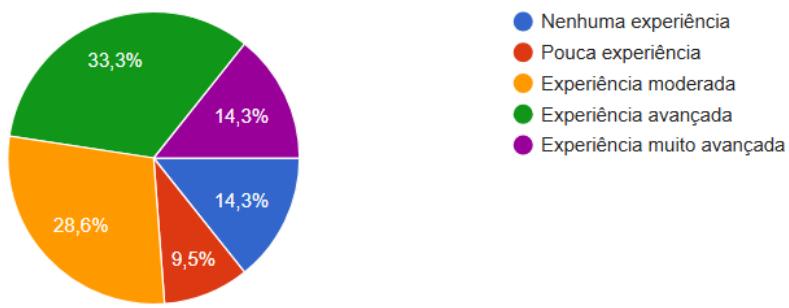


Figura 13 – Nível de experiência dos participantes em desenvolvimento de software.

Fonte: o autor

Os resultados mostram que a maioria dos participantes possui algum nível de experiência, com 33,3% relatando experiência avançada e 28,6% experiência moderada. Além disso, 14,3% possuíram experiência muito avançada.

Por outro lado, 14,3% dos participantes relataram não possuir nenhuma experiência em desenvolvimento de software, enquanto 9,5% indicaram ter pouca experiência. Esses dados destacam uma diversidade de níveis de conhecimento entre os participantes, o que permite explorar a eficácia do HuBot tanto para iniciantes quanto para usuários mais experientes.

6.2.5 Papel ou Função no Desenvolvimento de Software

Os participantes com experiência em desenvolvimento de software foram questionados sobre qual papel ou função desempenham geralmente. A Figura 14 apresenta a distribuição das respostas.

Os resultados indicam uma predominância de participantes que se identificaram como desenvolvedores, representando 33,3% das respostas. Outros papéis mencionados incluem Desenvolvedor FrontEnd (8,3%), Desenvolvedor Java (8,3%), Desenvolvedor de software (8,3%), Desenvolvedor e planejador (8,3%), Gerente de projetos (8,3%), além de combinações como Desenvolvedor e Gerente de projetos (8,3%). Um total de 16,7% dos participantes também mencionaram simplesmente “desenvolvedor”.

É importante destacar que a falta de padronização nas respostas fornecidas pelos participantes resultou em uma dispersão maior entre os papéis relatados.

Caso você tenha experiência em desenvolvimento de software, qual papel ou função você desempenha geralmente (por exemplo, desenvolvedor, analista de requisitos, gerente de projeto, etc.)?

12 respostas

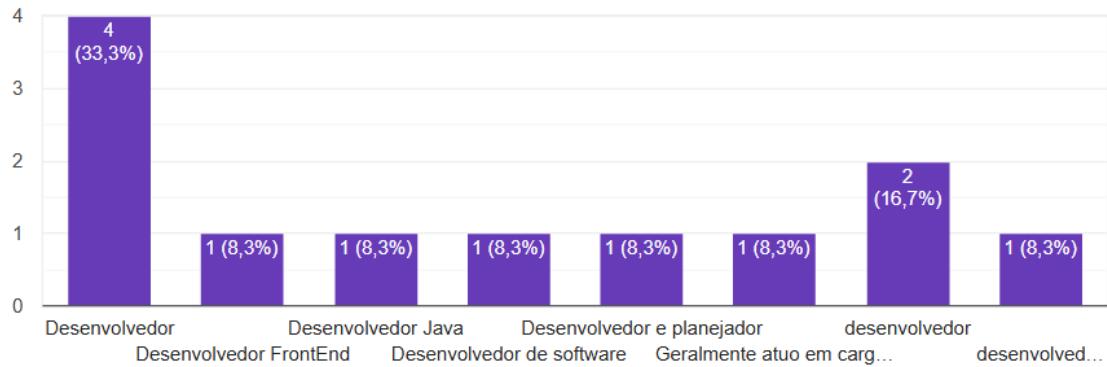


Figura 14 – Papel ou função desempenhada por participantes com experiência em desenvolvimento de software.

Fonte: o autor

6.2.6 Quão amigável achou a interação inicial com o HuBot?

Os participantes foram questionados sobre a amigabilidade da interação inicial com o HuBot. A Figura 15 apresenta as respostas obtidas.

Quão amigável achou a interação inicial com o HuBot?

21 respostas

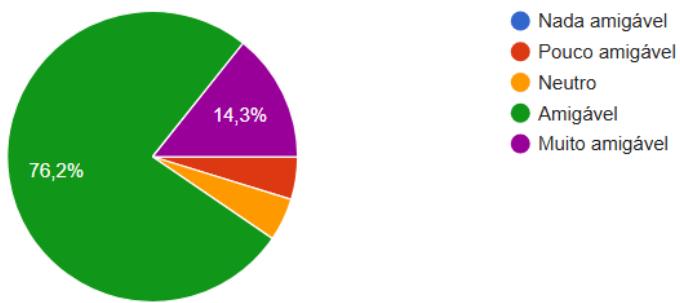


Figura 15 – Amigabilidade da interação inicial com o HuBot.

Fonte: o autor

Os resultados indicam que a maioria dos participantes considerou a interação com o HuBot como “amigável”, representando 76,2% das respostas. Outros 14,3% dos participantes classificaram a interação como “muito amigável”, enquanto 4,8% deram respostas

neutras e 4,8% consideraram a interação “pouco amigável”. Nenhum participante relatou uma interação “nada amigável”.

Durante a divulgação do formulário, a resposta “Pouco amigável” foi uma das primeiras respostas, após recebê-la realizamos uma atualização na forma da interação inicial do HuBot, isso pode ter influenciado as respostas subsequentes em algo mais positivo.

6.2.7 Quão fácil foi entender o propósito do HuBot?

Os participantes foram questionados sobre a facilidade de entender o propósito do HuBot. A Figura 16 apresenta a distribuição das respostas.

Quão fácil foi entender o propósito do HuBot?

21 respostas

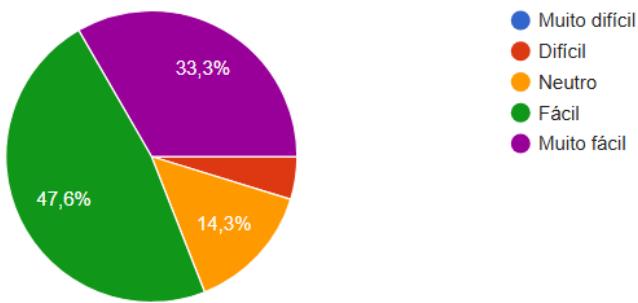


Figura 16 – Facilidade de entendimento do propósito do HuBot.

Fonte: o autor

Os resultados indicam que a maioria dos participantes considerou fácil entender o propósito do HuBot, com 47,6% das respostas nessa categoria. Além disso, 33,3% dos participantes acharam o propósito “muito fácil” de entender. Por outro lado, 14,3% dos participantes tiveram uma percepção neutra sobre a questão, enquanto 4,8% acharam o propósito “difícil” e nenhum participante relatou achar “muito difícil”.

As respostas “neutras” e “difícil” no início da divulgação do formulário também fez com que reformulássemos como o propósito era apresentado, o que pode ter sido refletido nas respostas “amigável” em sua grande maioria após essa atualização, indicando que o HuBot teve um bom nível de clareza quanto ao seu propósito.

6.2.8 Sugestões para Melhorar a Interação Inicial

Os participantes foram questionados se tinham alguma sugestão para melhorar a interação inicial com o HuBot. No total, 8 respostas foram coletadas:

- Não.
- Colocar um contexto no chatbot; se já existir, melhorar ele.
- Colocar sugestões de conversa e de escrita.
- Listar para o usuário, logo após a primeira interação, o que o bot pode fazer.
- Numerar as opções 1/2/3.
- No início, depois do comando `/start`, ele apenas apresenta uma mensagem curta questionando se pode ajudar em algo. Seria interessante se ele se apresentasse com uma explicação do seu propósito, algo que explique o seu objetivo. Dado que não conheço nada do projeto, ao interagir inicialmente com o chatbot, fiquei confuso sobre qual seria o próximo passo.
- Acho que está no caminho; ele dá boas perguntas, mas nem todo chatbot responde tudo de acordo.
- Na minha opinião, seria interessante melhorar a identação do texto inicial, adicionando mais emojis, para tornar mais legal a experiência do usuário. Isso se aplica às opções do menu também, como, por exemplo, um emoji para personas. Seria bom ter uma divisória entre as opções do menu, pois, quando clico em sucessivas opções, fica muito poluído de texto, dificultando a identificação das respostas.

Essas sugestões evidenciam algumas possíveis melhorias para a interação inicial do HuBot, incluindo a necessidade de contextualização mais clara, explicações introdutórias sobre o propósito do HuBot e a inclusão de elementos visuais, como emojis e divisórias, para tornar a experiência mais organizada e intuitiva. A sugestão que aponta a fala do HuBot após o comando `/start` foi acatada durante a divulgação do formulário.

6.2.9 Clareza no fluxo de criação de histórias de usuário

Os participantes foram questionados sobre a clareza do fluxo de criação de histórias de usuário no HuBot. A Figura 17 apresenta a distribuição das respostas.

Os resultados indicam que 57,1% dos participantes consideraram o fluxo “claro”, enquanto 28,6% o classificaram como “muito claro”. Outros 9,5% mantiveram uma percepção neutra, e 4,8% avaliaram o fluxo como “pouco claro”. Não houve respostas indicando que o fluxo fosse “nada claro”.

Analizando as respostas individualmente, os usuários que responderam como “pouco claro” possuem pouca experiência em desenvolvimento de software e em criação de histórias de usuário, sugerindo um espaço para melhorias visando o público que ainda não possui muito conhecimento na área.

O fluxo de criação de histórias de usuário foi claro?

21 respostas

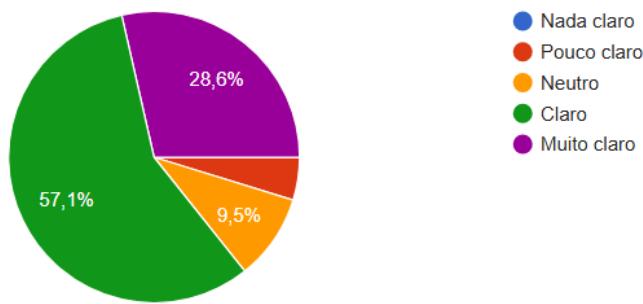


Figura 17 – Clareza no fluxo de criação de histórias de usuário no HuBot.

Fonte: o autor

6.2.10 Sugestões e Experiências com o Fluxo de Criação de Histórias de Usuário

Os participantes foram questionados se tinham sugestões de melhoria para o fluxo de criação de histórias de usuário ou se gostariam de compartilhar suas experiências. No total, 6 respostas foram registradas, conforme descrito a seguir:

- Não é uma sugestão, mas estou apontando o que achei mais legal do chat: a forma como o chat formula exemplos de histórias de usuário, trazendo o significado, exemplo e uma explicação do exemplo.
- No exemplo, o objetivo começava com “quero”. Inseri começando assim também, mas, no final, saiu duplicado “quero quero”.
- A lista de critérios de aceitação escreve o primeiro item com a primeira letra maiúscula, porém os demais itens são escritos começando com letra minúscula, o que quebra o padrão de visualização. Mas, ao meu ver, o fluxo de criação está coerente com o processo que geralmente é realizado.
- Acredito que, se o bot fosse mais direcionado, seria de mais fácil entendimento para o usuário. Com isso quero dizer que, se o bot listasse alternativas para o usuário escolher em cada tópico, não haveria necessidade de correções posteriormente e seria algo mais direcional, não abrindo espaços para bugs de IA.
- Tentar mostrar um caminho ou um exemplo já demonstrando como é uma história.
- Achei legal a ideia.

Essas respostas ressaltaram tanto pontos positivos quanto oportunidades de melhoria no fluxo de criação de histórias de usuário. Aspectos como padronização de texto, maior direcionamento nas escolhas e exemplos mais detalhados foram citados como caminhos para aprimorar a experiência do usuário.

6.2.11 Satisfação com o Fluxo de Definição das Partes que Compõem a História de Usuário

Os participantes foram questionados sobre o nível de satisfação com o fluxo de definição das partes que compõem a história de usuário (tipo de usuário, objetivo, motivo e critérios de aceitação). A Figura 18 apresenta a distribuição das respostas.

O quanto satisfeito você está com o fluxo de definição das partes que compõem a história de usuário?

(Definição de tipo de usuário, objetivo, motivo e critérios de aceitação)

21 respostas

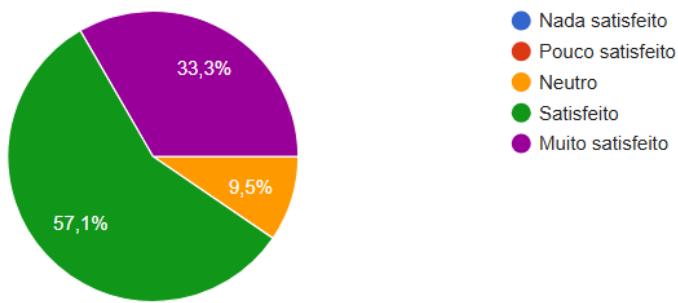


Figura 18 – Nível de satisfação com o fluxo de definição das partes que compõem a história de usuário.

Fonte: o autor

Os resultados mostram que 57,1% dos participantes estão “satisfeitos” com o fluxo, enquanto 33,3% relataram estar “muito satisfeitos”. Outros 9,5% mantiveram uma percepção neutra, e não houve respostas indicando insatisfação.

Isso indica que o fluxo atende bem às expectativas da maioria dos usuários, não recebendo nenhuma resposta negativa.

6.2.12 Avaliação das Sugestões de Melhorias para as Histórias de Usuário

Os participantes foram questionados sobre como avaliam as sugestões de melhorias para as histórias de usuário levantadas pelo HuBot. A Figura 19 apresenta a distribuição das respostas.

Como você avalia as sugestões de melhorias para as histórias de usuário levantadas pelo HuBot?

21 respostas

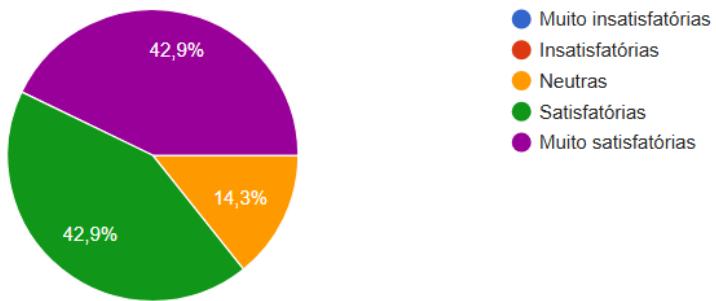


Figura 19 – Avaliação das sugestões de melhorias para as histórias de usuário levantadas pelo HuBot.

Fonte: o autor

Os resultados indicam que 42,9% dos participantes consideraram as sugestões “satisfatórias”, enquanto outros 42,9% avaliaram como “muito satisfatórias”. Um total de 14,3% dos respondentes manteve uma percepção neutra, e não houve respostas indicando insatisfação.

Esses dados sugerem que as sugestões de melhorias fornecidas pelo HuBot foram amplamente bem recebidas pelos usuários.

6.2.13 Clareza nas Explicações Fornecidas pelo HuBot

Os participantes foram questionados sobre a clareza das explicações fornecidas pelo HuBot, como as relacionadas ao conceito de história de usuário e seu objetivo. A Figura 20 apresenta a distribuição das respostas.

Os resultados mostram que 47,6% dos participantes consideraram as explicações “claras”, enquanto 42,9% avaliaram como “muito claras”. Outros 4,8% tiveram uma percepção neutra, e 4,8% avaliaram as explicações como “pouco claras”. Não houve respostas indicando que as explicações foram “nada claras”.

Esses dados indicam que as explicações fornecidas pelo HuBot são em sua maioria compreendidas pelos usuários, com uma aceitação positiva predominante e apenas um pequeno percentual sugerindo a necessidade de ajustes para melhorar a clareza. Ao analisar as respostas individualmente foi constatado que as respostas “pouco claras” vieram de usuário que possuem nenhuma ou pouca experiência em desenvolvimento de software e criação de histórias de usuário, reforçando a dificuldade no entendimento do HuBot para esse público.

As explicações fornecidas pelo HuBot foram claras (por exemplo, sobre o que é uma história de usuário ou o objetivo)?

21 respostas

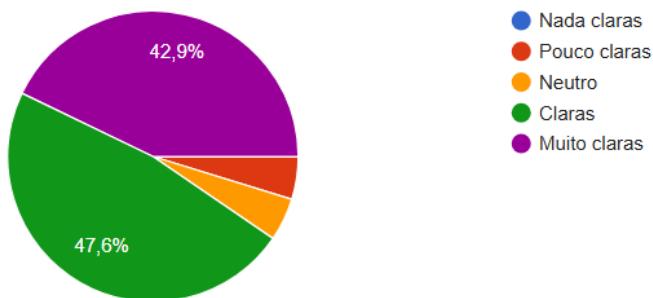


Figura 20 – Avaliação da clareza das explicações fornecidas pelo HuBot.

Fonte: o autor

6.2.14 Principais Pontos Positivos do HuBot

Os participantes foram questionados sobre os principais pontos positivos do HuBot. Abaixo, as respostas obtidas:

- Avaliação das histórias de usuário.
- Explicações claras e objetivas sobre histórias de usuário, auxiliando na sua criação.
- Sugestões claras e úteis em situações hipotéticas.
- Correção de erros gramaticais para aprimorar a escrita.
- Exemplos de histórias de usuário e interação, com sugestões para que o usuário faça mais perguntas.
- Guiar o usuário de forma eficiente durante o processo de criação e oferecer uma avaliação final da história criada.
- Rapidez, fluidez e interatividade.
- Facilidade de acesso e interface intuitiva.
- Inovação e explicações detalhadas sobre os componentes de uma história de usuário.

Esses pontos destacam como o HuBot tem sido útil e eficaz para os usuários. A facilidade de interação, a clareza nas explicações e a capacidade de guiar o usuário durante o processo de criação das histórias de usuário. Além disso, a rapidez e a correção de erros de escrita também foram pontos positivos mencionados.

6.2.15 Dificuldades e Problemas Durante a Interação

Os participantes foram questionados sobre dificuldades ou problemas encontrados durante a interação com o HuBot. As dificuldades mencionadas foram:

- Na criação da história de usuário, palavras repetidas apareceram na versão final.
- Não foi possível refazer uma etapa sem voltar ao menu, o que fez o usuário escrever informações aleatórias para continuar o processo.
- Falta de contexto em determinados momentos, o que causou confusão quanto ao progresso da interação.
- Duplicação do verbo “quero” no objetivo da história de usuário.
- A ausência de numeração nas opções, o que dificultou a agilidade na escolha.
- Ao tentar avaliar a história, houve uma falha na resposta, com a mensagem de análise sendo repetida três vezes sem retorno.
- Dificuldade em sair da criação de uma história de usuário, sem uma opção clara para finalizar ou reiniciar o processo.
- Excesso de texto sem divisão clara, fazendo o usuário ter que ler longas respostas para identificar a informação correta, o que tornou o processo mais demorado.

Esses pontos mostram algumas das dificuldades que os usuários enfrentaram, principalmente relacionadas à falta de flexibilidade no fluxo de interação e ao excesso de texto sem organização visual clara. Porém muitos usuários não relataram problemas.

6.2.16 Sugestões de Melhorias para o HuBot

Os participantes foram convidados a sugerir melhorias para o HuBot. As sugestões foram:

- O bot deve responder com algo como “Não entendi” ao receber algo fora do escopo.
- Melhorar a estrutura do texto para um visual mais agradável, sem alterar o conteúdo.
- Reconhecer o comando “menu” e interromper a criação da história de usuário, retornando ao menu de opções.
- Melhorar o contexto e histórico do chatbot para uma interação mais fluida.
- Numerar as opções para facilitar e agilizar a escolha.

- Preparar o bot para lidar com mensagens variadas, oferecendo uma resposta adequada mesmo quando o usuário se desvia um pouco do seu objetivo principal.
- Reduzir a quantidade de texto nas respostas.
- Fornecer uma apresentação inicial mais clara, para evitar confusão, especialmente para usuários que fazem perguntas informais, como abreviações.
- Apresentar um exemplo de criação de história de usuário de forma contínua, sem ser por etapas, para facilitar o entendimento.
- Melhorar a divisão entre respostas quando o usuário escolhe múltiplas opções no menu e considerar a inserção de emojis ou imagens para tornar a interface mais atraente e intuitiva.
- Incluir um exemplo prático mais completo na definição dos critérios de aceitação.

Essas sugestões indicam que os usuários encontraram áreas que poderiam ser aprimoradas para tornar a experiência mais intuitiva. Melhorias na clareza da apresentação, na interação com comandos e no contexto fornecido poderiam ajudar a atender melhor às necessidades dos usuários.

6.2.17 Avaliação Geral do HuBot

Os participantes foram questionados sobre a sua experiência geral com o HuBot. A Figura 21 apresenta a distribuição das respostas.

Como você avalia sua experiência geral com o HuBot?

21 respostas

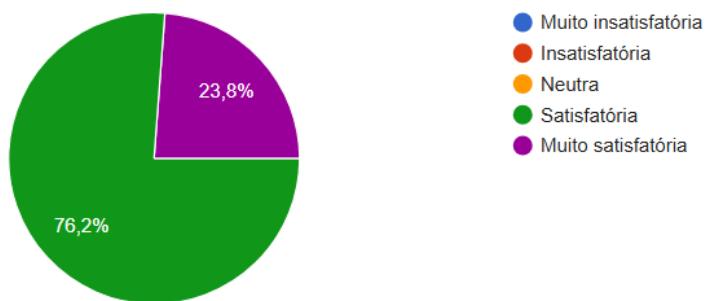


Figura 21 – Avaliação geral da experiência com o HuBot.

Fonte: o autor

Os resultados mostram que 76,2% dos participantes avaliaram a experiência como “satisfatória”, enquanto 23,8% consideraram a experiência “muito satisfatória”. Não houve

respostas indicando que a experiência foi “insatisfatória”, “muito insatisfatória” ou “neutras”.

Esses dados indicam que a experiência geral com o HuBot foi amplamente positiva, com uma maioria significativa dos participantes avaliando o sistema de forma satisfatória ou muito satisfatória.

6.2.18 Utilidade do HuBot para a Criação de Histórias de Usuário

Quando questionados sobre a utilidade do HuBot para pessoas que desejam criar histórias de usuário, os participantes mostraram uma percepção positiva. A Figura 22 apresenta a distribuição das respostas.

Você acredita que o HuBot poderia ser útil para pessoas que desejam criar histórias de usuário?

21 respostas

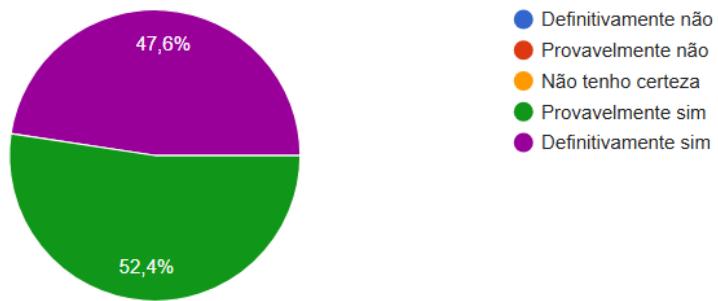


Figura 22 – Avaliação da utilidade do HuBot na criação de histórias de usuário.

Fonte: o autor

Os resultados indicam que 52,4% dos participantes acreditam que o HuBot seria “provavelmente útil” para a criação de histórias de usuário, enquanto 47,6% estão convencidos de que ele seria “definitivamente útil”. Nenhuma resposta indicou que o HuBot seria “provavelmente não útil” ou “definitivamente não útil”, refletindo uma percepção muito positiva sobre sua aplicabilidade.

Esses dados sugerem que o HuBot tem grande potencial para auxiliar na criação de histórias de usuário, sendo visto como uma ferramenta útil por quase todos os participantes.

7 Conclusão

Este trabalho abordou o desenvolvimento do HuBot, um chatbot projetado para auxiliar na criação e avaliação de histórias de usuário. A experimentação foi realizada utilizando a abordagem de pesquisa-ação, com a aplicação de um questionário para coletar opiniões e identificar possíveis melhorias no HuBot.

7.1 Objetivos Alcançados

A pesquisa atingiu com sucesso o seu principal objetivo [1.2.1](#): desenvolver um *chatbot* capaz de auxiliar os usuários no levantamento e avaliação de histórias de usuário.

Além disso, os objetivos específicos [1.2.2](#), que são desdobramentos do objetivo geral, foram alcançados:

- Pesquisa das principais formas de se escrever uma história de usuário e da criação de *prompts*.
- Desenvolvimento de um *prompt* eficaz, afim de obter sugestões de melhorias para as HUs através de um LLM.
- Validação da eficácia do *prompt* através de testes manuais, até obter análises e melhorias de HUs de forma estruturada.
- Fornecimento de informações básicas sobre histórias de usuário.

7.2 Competências do HuBot

O HuBot cumpre com o sistema proposto e o objetivo principal desta pesquisa, possibilitando a criação e avaliação de histórias de usuário (HUs) por meio de interações conversacionais naturais. A ferramenta também permite responder a perguntas sobre conceitos fundamentais relacionados a HUs, critérios de aceitação e personas, embora essa funcionalidade ainda possua espaço para melhorias, já que usuários sem conhecimento prévio em histórias de usuário não apresentaram uma avaliação positiva nesse ponto.

De maneira geral, os usuários demonstraram satisfação com a experiência proporcionada pelo HuBot. Na pergunta de avaliação global da ferramenta, todos os participantes responderam que consideraram sua experiência como “satisfatória” ou “muito satisfatória”. Esse retorno evidencia que o HuBot conseguiu atender às expectativas dos usuários e se mostrou intuitivo e funcional para a maioria dos cenários testados.

Além disso, a principal funcionalidade da ferramenta, a avaliação de histórias de usuário, também foi bem recebida. Apesar de 14,3% dos usuários terem classificado sua experiência como “neutra”, o restante avaliou a funcionalidade como “satisfatória” ou “muito satisfatória”, mostrando que a ferramenta é eficaz em seu propósito central. Esses resultados indicam que o HuBot é uma solução promissora para auxiliar no desenvolvimento ágil, especialmente para quem já possui algum conhecimento sobre histórias de usuário.

7.3 Fragilidades do HuBot

Ainda que os objetivos gerais e os propósitos dos HuBot terem sido alcançados, ainda há um amplo espaço para melhorias, como ocorre em todo e qualquer software. Exemplos de fragilidades foram evidenciados em diversos comentários dos usuários, colecionados por meio do formulário.

Essas fragilidades ficam ainda mais perceptíveis entre os usuários com pouca experiência ou familiaridade com chatbots. Entre as melhorias e sugestões feitas pelos usuários, podemos agrupar nas seguintes categorias:

- **Clareza e contexto:** Alguns usuários relataram dificuldade em compreender o propósito do HuBot e navegar pelo fluxo de criação das HUs. Sugerindo assim, a necessidade de instruções mais claras e um contexto melhor apresentado.
- **Interatividade e flexibilidade:** Foram mencionadas limitações na navegação, como a impossibilidade de refazer certas etapas durante os fluxos ou a falta de opções para finalizar ou reiniciar uma etapa. Essa rigidez pode acabar deixando o usuário insatisfeito e sem alternativas claras sobre como proceder.
- **Acessibilidade e usabilidade:** Também foram reportadas reclamações sobre mensagens repetitivas e o excesso de informações transmitidas em poucas mensagens, que, por vezes, podem acabar tornando a interação cansativa. Além disso, foi sugerida a inclusão de mais emojis para tornar a comunicação mais leve e fluida.

Outras fragilidades podem ser identificadas no HuBot, mas as mencionadas acima foram as que obtiveram mais destaque.

7.4 Trabalhos Futuros

Com a conclusão deste trabalho, oportunidades de aprimoramento do HuBot foram identificadas. Nesta seção, destacam-se algumas possibilidades para trabalhos futuros,

que podem contribuir para a evolução da ferramenta e sua aplicabilidade em diferentes contextos:

- **Correções e melhorias:** Corrigir e buscar minimizar as fragilidade presentes no HuBot, como as destacadas no tópico 7.3.
- **Expansão de funcionalidades:** Implementar novas funcionalidades, como a geração automática de critérios de aceitação detalhados a partir de histórias de usuário.
- **Integração com outras plataformas:** Ampliar a integração do HuBot para suportar mais plataformas de comunicação.
- **Suporte a múltiplos idiomas:** Desenvolver a capacidade do HuBot de operar em diferentes idiomas além do português, como inglês e espanhol, para ampliar sua acessibilidade.
- **Engajamento e feedback:** Adicionar mensagens para capturar o *feedback* dos usuários ao final de cada fluxo, para que o *chatbot* esteja em constante evolução.
- **Ferramentas de avaliação alternativas :** Explorar o uso de outras ferramentas baseadas em LLM para avaliar histórias de usuário, como o *DeepSeek*, a fim de comparar os resultados gerados com os obtidos por *LLMs* da *OpenAI* e investigar possíveis diferenças na qualidade das respostas.

Referências

- 2001 Python Software Foundation. *Python Geral*. 2025. Accessed: 2025-01-12. Disponível em: <<https://docs.python.org/pt-br/dev/faq/general.html>>. Citado na página 43.
- ABDELLATIF, A. *Towards Understanding and Improving the Value of Chatbots in Software Engineering*. Tese (Doutorado) — Concordia University, 2021. Disponível em: <https://spectrum.library.concordia.ca/id/eprint/990118/1/Abdellatif_PhD_F2022.pdf>. Citado na página 27.
- ADAMOPOULOU, E.; MOUSSIADES, L. An overview of chatbot technology. *Springer, Cham*, 2020. Disponível em: <https://link.springer.com/chapter/10.1007/978-3-030-49186-4_31?ref=blog.min.io>. Citado na página 27.
- ALDAVE, A. et al. Leveraging creativity in requirements elicitation within agile software development: A systematic literature review. *Journal of Systems and Software*, Elsevier, v. 157, p. 110396, 2019. Citado na página 31.
- Amazon Web Services, Inc. *O que é Python?* 2025. Accessed: 2025-01-12. Disponível em: <<https://aws.amazon.com/pt/what-is/python/>>. Citado na página 43.
- ARVIDSSON, S.; AXELL, J. Prompt engineering guidelines for llms in requirements engineering. 2023. Disponível em: <<https://gupea.ub.gu.se/handle/2077/77967>>. Citado na página 29.
- BANO, M. et al. Teaching requirements elicitation interviews: an empirical study of learning from mistakes. *Requirements Engineering*, Springer, v. 24, p. 259–289, 2019. Citado na página 32.
- BECK, K. Embracing change with extreme programming. *Computer*, v. 32, n. 10, p. 70–77, 1999. Disponível em: <<https://ieeexplore.ieee.org/document/796139>>. Citado na página 33.
- BOCKLISCH, T. et al. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*, 2017. Citado na página 28.
- CAHN, J. *Architecture, Design, & Development*. Tese (Senior Thesis) — University of Pennsylvania, 2017. Citado na página 27.
- CONNECTING Brazilian Families with Emergency Government Assistance. 2021. Disponível em: <<https://rasa.community/showcase/picpay/>>. Citado na página 29.
- CUNNINGHAM, S. et al. A review of chatbots in education: Practical steps forward. *Australasian Association for Engineering Education*, 2019. Disponível em: <<https://eprints.qut.edu.au/134323/>>. Citado na página 27.
- DIALOGFLOW. 2024. Disponível em: <<https://cloud.google.com/dialogflow>>. Citado na página 29.

DWITAMA, F.; RUSLI, A. User stories collection via interactive chatbot to support requirements gathering. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, v. 18, n. 2, p. 890–898, 2020. Disponível em: <<http://telkomnika.uad.ac.id/index.php/TELKOMNIKA/article/view/14866>>. Citado na página 36.

FALBO, R. de A. Engenharia de requisitos. 2017. Citado 2 vezes nas páginas 30 e 31.

FRANCO, E. F. *Um modelo de gerenciamento de projetos baseado nas metodologias ágeis de desenvolvimento de software e nos princípios da produção enxuta*. Tese (Doutorado) — Universidade de São Paulo, 2007. Disponível em: <<https://www.teses.usp.br/teses/disponiveis/3/3141/tde-09012008-155823/en.php>>. Citado na página 34.

GIT - Reference. 2024. Disponível em: <<https://git-scm.com/docs>>. Citado na página 43.

GITHUB Docs. 2024. Disponível em: <<https://docs.github.com/pt/get-started/start-your-journey/about-github-and-git>>. Citado na página 43.

HERDIKA, H. R.; BUDIARDJO, E. K. Variability and commonality requirement specification on agile software development: Scrum, xp, lean, and kanban. In: *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*. [s.n.], 2020. p. 323–329. Disponível em: <<https://ieeexplore.ieee.org/abstract/document/9274564>>. Citado na página 34.

HOFMANN, H.; LEHNER, F. Requirements engineering as a success factor in software projects. *IEEE Software*, v. 18, n. 4, p. 58–66, 2001. Citado na página 39.

HÖRNEMALM, A. *ChatGPT as a Software Development Tool: The Future of Development*. 2023. Disponível em: <<https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1768339&dswid=1869>>. Citado na página 27.

JOSHI, A. et al. Likert scale: Explored and explained. *British journal of applied science & technology*, v. 7, n. 4, p. 396–403, 2015. Citado na página 67.

KALYAN, K. A survey of gpt-3 family large language models including chatgpt and gpt-4. *Natural Language Processing Journal*, 2024. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2949719123000456>>. Citado na página 28.

KHANNA, A. et al. A study of today's a.i. through chatbots and rediscovery of machine intelligence. *International Journal of u- and e- Service, Science and Technology*, 2015. Disponível em: <https://article.nadiapub.com/IJUNESST/vol8_no7/28.pdf>. Citado na página 27.

KUMAR, R.; ALI, M. M. A review on chatbot design and implementation techniques. *Int. J. Eng. Technol.*, v. 7, n. 11, p. 2791–2800, 2020. Disponível em: <https://www.researchgate.net/profile/Ramakrishna-Kumar-2/publication/348569890_A_Review_on_Chatbot_Design_and_Implementation_Techniques/links/60058174a6fdccdc8611978/A-Review-on-Chatbot-Design-and-Implementation-Techniques.pdf>. Citado na página 28.

- LAM, L. K.; HURTADO, C. A. L.; PORTILLO, L. W. Framework for automating requirement elicitation using a chatbot. In: *2022 IEEE Engineering International Research Conference (EIRCON)*. [S.l.: s.n.], 2022. p. 1–4. Citado na página 37.
- LIBARDI, P. L.; BARBOSA, V. Métodos ágeis. *Monografia (Graduação em ciência da computação)*, 2010. Disponível em: <<https://www.cursosavante.com.br/cursos/curso533/conteudo7485.pdf>>. Citado 2 vezes nas páginas 34 e 35.
- LUCASSEN FABIANO DALPIAZ, J. M. E. M. v. d. S. B. G. The use and effectiveness of user stories in practice. In: DANEVA, M.; PASTOR, O. (Ed.). *Requirements Engineering: Foundation for Software Quality*. Cham: Springer International Publishing, 2016. p. 205–222. Citado na página 21.
- LUCASSEN, G. et al. Improving user story practice with the grimm method: A multiple case study in the software industry. In: GRUNBACHER, P.; PERINI, A. (Ed.). *Requirements Engineering: Foundation for Software Quality*. Cham: Springer International Publishing, 2017. p. 235–252. ISBN 978-3-319-54045-0. Citado 2 vezes nas páginas 21 e 22.
- LUCASSEN, G. et al. Improving agile requirements: the quality user story framework and tool. *Requirements engineering*, Springer, v. 21, p. 383–403, 2016. Citado na página 39.
- LUCIA, A. D.; QUSEF, A. et al. Requirements engineering in agile software development. *Journal of emerging technologies in web intelligence*, Citeseer, v. 2, n. 3, p. 212–220, 2010. Citado na página 33.
- MANIFESTO Ágil. 2001. Disponível em: <<https://agilemanifesto.org/iso/ptbr/manifesto.html>>. Acesso em: March 24, 2024. Citado na página 32.
- MARCONI, M. d. A.; LAKATOS, E. M. *Fundamentos de metodologia científica*. 5. ed. São Paulo: Atlas, 2003. Disponível em: <https://edisciplinas.usp.br/pluginfile.php/7237618/mod_resource/content/1/Marina%20Marconi%20Eva%20Lakatos_Fundamentos%20de%20metodologia%20cient%C3%ADfica.pdf>. Citado na página 24.
- MENZINSKY, A. et al. Historias de usuario. *Ingeniería de requisitos ágil*, 2018. Citado 2 vezes nas páginas 21 e 32.
- Nginx, Inc. *Nginx*. 2024. Accessed: 2024-12-03. Disponível em: <<https://nginx.org/en/>>. Citado na página 44.
- O'HEOCHA, C.; CONBOY, K. The role of the user story agile practice in innovation. In: ABRAHAMSSON, P.; OZA, N. (Ed.). *Lean Enterprise Software and Systems*. Berlin: Springer Berlin Heidelberg, 2010. p. 20–30. Citado 2 vezes nas páginas 22 e 35.
- OLIVEIRA, R. R. A técnica de priorização moscow. *Management Plaza International—The Management Certification Company. PRINCE2*, 2014. Citado na página 57.
- OpenAI. *Introduction - OpenAI API*. 2024. Accessed: 2024-05-29. Disponível em: <<https://platform.openai.com/docs/introduction>>. Citado na página 41.
- OPENAI. *Prompt engineering*. 2024. Disponível em: <<https://platform.openai.com/docs/guides/prompt-engineering>>. Citado 2 vezes nas páginas 30 e 45.

PACHECO, C.; GARCÍA, I.; REYES, M. Requirements elicitation techniques: a systematic literature review based on the maturity of the techniques. *IET Software*, Wiley Online Library, v. 12, n. 4, p. 365–378, 2018. Citado na página 31.

PAREDES, C. M. G.; MACHUCA, C.; CLAUDIO, Y. M. S. Chatgpt api: Brief overview and integration in software development. *International Journal of Engineering Insights*, v. 1, n. 1, p. 25–29, 2023. Disponível em: <<https://doi.org/10.61961/injei.v1i1.7>>. Citado na página 41.

PAVEL, I. *Comparing chatbot frameworks: A study of Rasa and Botkit*. Dissertação (Mestrado), 2021. Citado na página 29.

POHL, K. *Requirements engineering: An overview*. [S.l.]: Citeseer, 1996. Citado na página 30.

POKHAREL, P.; VAIDYA, P. A study of user story in practice. In: *2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI)*. [s.n.], 2020. p. 1–5. Disponível em: <<https://ieeexplore.ieee.org/document/9325670>>. Citado na página 35.

POPPENDIECK, M. Lean software development. In: *29th International Conference on Software Engineering (ICSE'07 Companion)*. [s.n.], 2007. p. 165–166. Disponível em: <<https://ieeexplore.ieee.org/document/4222727/authors#authors>>. Citado na página 33.

POPPENDIECK, M.; CUSUMANO, M. A. Lean software development: A tutorial. *IEEE Software*, v. 29, n. 5, p. 26–32, 2012. Disponível em: <<https://ieeexplore-ieee-org.ez54.periodicos.capes.gov.br/document/6226341>>. Citado na página 33.

QUINN, M. P. *Qualitative research & evaluation methods*. [S.l.]: Sage Publication Ltd., London, 2002. Citado na página 23.

RAHARJANA, I. K.; SIAHAAN, D.; FATICHAH, C. User stories and natural language processing: A systematic literature review. *IEEE access*, IEEE, v. 9, p. 53811–53826, 2021. Disponível em: <<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9393933>>. Citado na página 35.

RASA. 2024. Disponível em: <<https://rasa.com/>>. Citado na página 28.

Rasa Technologies GmbH. *Rasa Open Source*. 2025. Accessed: 2025-01-12. Disponível em: <<https://rasa.com/docs/rasa/>>. Citado na página 44.

RIETZ, T.; MAEDCHE, A. Ladderbot: A requirements self-elicitation system. In: *IEEE 2019 IEEE 27th International Requirements Engineering Conference (RE)*. [S.l.], 2019. p. 357–362. Citado na página 37.

RODRIGUEZ, A. D.; DEARSTYNE, K. R.; CLELAND-HUANG, J. Prompts matter: Insights and strategies for prompt engineering in automated software traceability. In: *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*. [S.l.: s.n.], 2023. p. 455–464. Citado na página 30.

- SANTANA, T. S.; KUDO, T. N.; BULCAO-NETO, R. F. Um relato de experiência sobre o uso de histórias de usuário e critérios de aceitação no ensino de requisitos de software. In: SBC. *Anais do XXVIII Workshop de Informática na Escola*. [S.l.], 2022. p. 200–210. Citado na página 21.
- SAVANUR, A. et al. Application of chatbot for consumer perspective using artificial intelligence. In: *2021 6th International Conference on Communication and Electronics Systems (ICCES)*. [s.n.], 2021. p. 1479–1483. Disponível em: <<https://ieeexplore.ieee.org/ez54.periodicos.capes.gov.br/document/9488990>>. Citado na página 27.
- SEDANO, T.; RALPH, P.; PÉRAIRE, C. The product backlog. In: *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. [s.n.], 2019. p. 200–211. Disponível em: <<https://ieeexplore.ieee.org/document/8812076>>. Citado na página 34.
- SHARMA, V.; GOYAL, M.; MALIK, D. An intelligent behaviour shown by chatbot system. *International Journal of New Technology and Research*, v. 3, n. 4, 4 2017. Citado na página 27.
- SILVA, R.; BARRETO, F. Introdução a engenharia de requisitos. 08 2011. Citado na página 31.
- SOARES, H. F. et al. Investigating the link between user stories and documentation debt on software projects. In: *2015 12th International Conference on Information Technology - New Generations*. [s.n.], 2015. p. 385–390. Disponível em: <<https://ieeexplore.ieee.org/document/7113503>>. Citado na página 35.
- SOARES, M. dos S. Metodologias ágeis extreme programming e scrum para o desenvolvimento de software. *Revista Eletrônica de Sistemas de Informação*, v. 3, n. 1, 2004. Disponível em: <<https://www.periodicosibepes.org.br/index.php/reinfo/article/view/146#citations>>. Citado na página 33.
- SPINELLIS, D. Git. *IEEE Software*, v. 29, n. 3, p. 100–101, 2012. Citado na página 42.
- SURANA, C. S. R. K. et al. Intelligent chatbot for requirements elicitation and classification. In: *2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*. [S.l.: s.n.], 2019. p. 866–870. Citado na página 36.
- WAILTHARE, S. et al. Artificial intelligence based chat-bot. *Artificial Intelligence*, v. 5, n. 03, p. 2305–2306, 2018. Citado na página 28.
- WAUTELET, Y. et al. Unifying and extending user story models. In: SPRINGER. *Advanced Information Systems Engineering: 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings* 26. 2014. p. 211–225. Disponível em: <https://link.springer.com/chapter/10.1007/978-3-319-07881-6_15>. Citado na página 35.
- WEI, J. et al. *Emergent Abilities of Large Language Models*. 2022. Citado na página 28.
- WEIZENBAUM, J. Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 9, n. 1, p. 36–45, jan 1966. ISSN 0001-0782. Disponível em: <<https://doi.org/10.1145/365153.365168>>. Citado na página 27.

WHITE, J. et al. *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*. 2023. Disponível em: <<https://arxiv.org/abs/2302.11382v1>>. Citado na página 29.

ZACHARIAS, I. C. S. et al. User stories method and assistive technology product development: A new approach to requirements elicitation. In: CAMBRIDGE UNIVERSITY PRESS. *Proceedings of the Design Society: International Conference on Engineering Design*. [S.l.], 2019. v. 1, n. 1, p. 3781–3790. Citado na página 39.

ZHAO, H. et al. Explainability for large language models: A survey. *ACM Trans. Intell. Syst. Technol.*, Association for Computing Machinery, New York, NY, USA, v. 15, n. 2, 2024. ISSN 2157-6904. Disponível em: <<https://doi.org/10.1145/3639372>>. Citado na página 28.

ZHAO, W. X. et al. *A Survey of Large Language Models*. 2023. Disponível em: <<https://arxiv.org/abs/2303.18223>>. Citado na página 21.

Apêndices

APÊNDICE A – Primeiro tentativa de criação de prompt para avaliação de histórias de usuário

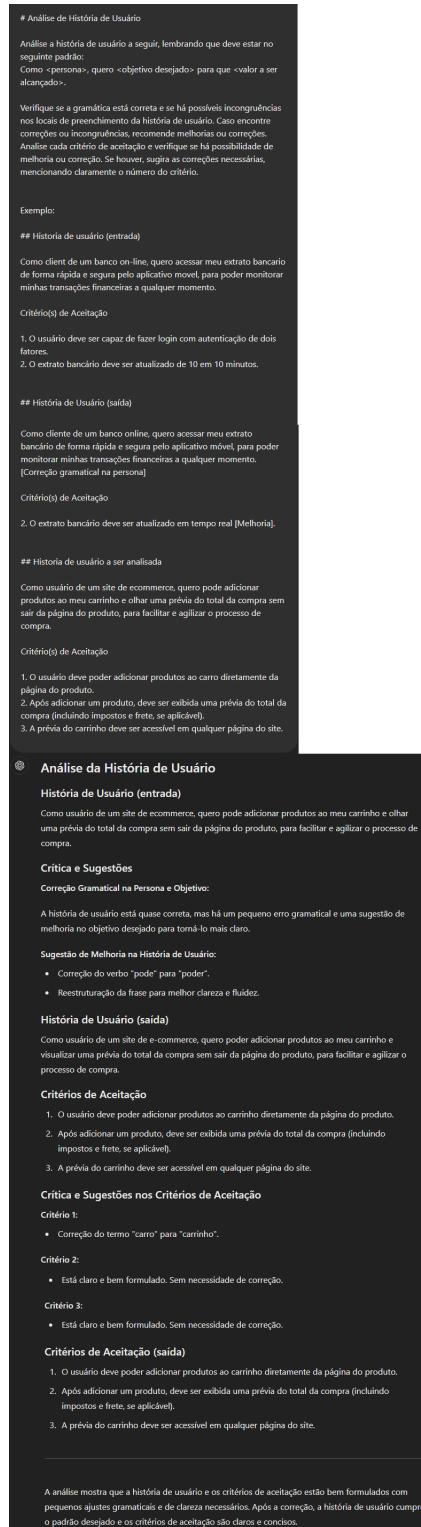


Figura 23 – Primeira tentativa de prompt para avaliação de histórias de usuário

Fonte: o autor

APÊNDICE B – Segunda tentativa de criação de prompt para avaliação de histórias de usuário

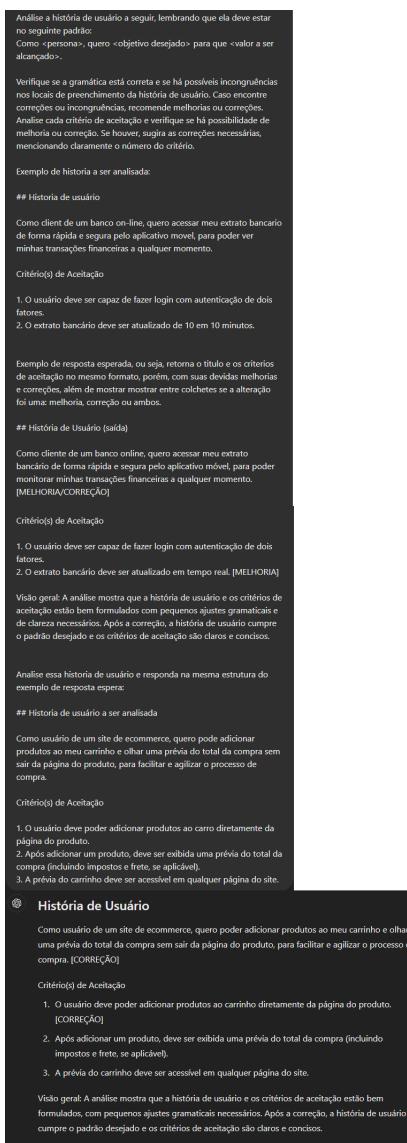


Figura 24 – Segunda tentativa de prompt para avaliação de histórias de usuário

Fonte: o autor

APÊNDICE C – Terceira tentativa de criação de prompt para avaliação de histórias de usuário

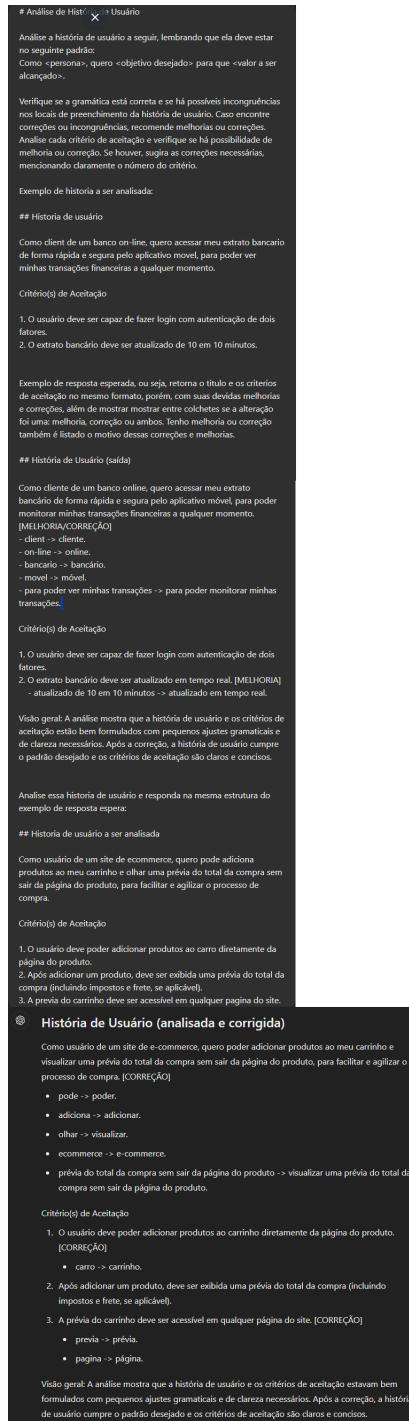


Figura 25 – Terceira tentativa de prompt para avaliação de histórias de usuário

Fonte: o autor