UNIVERSIDADE FEDERAL DO PAMPA

RENATO DE SOUZA GARCIA

CODERBOT: UM ASSISTENTE VIRTUAL PARA APOIAR A APRENDIZAGEM DE PROGRAMAÇÃO POR MEIO DE *WORKED EXAMPLES*

Alegrete
2025

RENATO DE SOUZA GARCIA

CODERBOT: UM ASSISTENTE VIRTUAL PARA APOIAR A APRENDIZAGEM DE PROGRAMAÇÃO POR MEIO DE *WORKED EXAMPLES*

Projeto de Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Software como requisito parcial para a obtenção do título de Mestre em Engenharia de Software.

Orientador: Williamson Alison Freitas Silva
Coorientador: Pedro Henrique Dias Valle

Alegrete
2025

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

G216c Garcia, Renato
CoderBot: Um Assistente Virtual para Apoiar a Aprendizagem de Programação por meio de Worked Examples / Renato Garcia.
123 p.

Dissertação(Mestrado)-- Universidade Federal do Pampa, MESTRADO EM ENGENHARIA DE SOFTWARE, 2024.
"Orientação: Williamson Silva".

1. Assistente Virtual. 2. Ensino de Programação. 3. Aprendizagem Baseado em Exemplos. 4. Worked Examples. I. Título.

RENATO DE SOUZA GARCIA

CoderBot: um assistente virtual para apoiar a aprendizagem de programação por meio de worked examples

Dissertação apresentada ao Programa de Pós–Graduação em Engenharia de Software da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Mestre em Engenharia de Software.

Dissertação defendida e aprovada em: 20/12/2024.

Banca examinadora:

_____

Prof. Dr. Williamson Alison Freitas Silva

Orientador

UNIPAMPA

_____

Prof. Dr. Pedro Henrique Dias Vall

Coorientador

IME-USP

_____

Prof. Dr. Igor Fabio Steinmacher

Northern Arizona University(NAU)

Prof. Dr. Gabriel Alves de Albuquerque Júnior

UFRPE

Prof. Dr. Paulo Silas Severo de Souza

UNIPAMPA

Assinado eletronicamente por **WILLIAMSON ALISON FREITAS SILVA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 20/12/2024, às 17:28, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **PAULO SILAS SEVERO DE SOUZA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 20/12/2024, às 17:28, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **Igor Fabio Steinmacher**, **Usuário Externo**, em 20/12/2024, às 17:29, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **Gabriel Alves de Albuquerque Júnior**, **Usuário Externo**, em 20/12/2024, às 17:29, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **Pedro Henrique Dias Valle**, **Usuário Externo**, em 20/12/2024, às 17:30, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1621902** e o código CRC **ED763139**.

"O conhecimento não é uma propriedade,
mas um processo de descoberta contínua."
— Carl Sagan

AGRADECIMENTOS

Primeiramente à Deus pela vida, saúde, força e sabedoria concedidas ao longo desta caminhada.

À minha família, em especial, meus pais (Carmen e Sérgio), cujos quais me proporcionaram não apenas a oportunidade de seguir meus estudos, mas também o apoio incondicional, o suporte e o exemplo. À minha irmã e sobrinhas pelos momentos de descontração. Amo todos vocês.

Aos amigos, que mesmo diante de ausências e distâncias, sempre chamavam para uma conversa (voip), uma distração, e colaborando bons momentos de diversão. Agradeço aos meus colegas do programa de mestrado pelos auxílios, retiradas de dúvidas e demais colaborações.

Aos membros da banca de defesa, Prof. Dr. Igor Fabio Steinmacher, Prof. Dr. Gabriel Alves de Albuquerque Júnior, Prof. Dr. Paulo Silas Severo de Souza, pelas contribuições e apontamentos, cujos quais ajudaram a melhorar esse trabalho.

À Universidade Federal do Pampa (Unipampa) em especial ao Programa de Pós-Graduação em Engenharia de Software (PPGES) pela oportunidade.

Ao meu coorientador do mestrado Prof. Dr. Pedro Henrique Dias Valle, que colaborou durante a execução deste trabalho com correções, dicas, apontamentos, reuniões fora de hora e até mesmo conselhos diante do nervosismo de passar por uma banca de mestrado. Muito obrigado.

Por fim, ao meu orientador, Prof. Dr. Williamson Alison Freitas Silva, que desde o começo quando chamei para conversar sobre a possibilidade de orientação foi atencioso e prestativo. Mesmo diante de uma grande decisão durante o mestrado, demonstrou apoio e incentivo. E hoje eu vejo, que fiz a escolha certa em permanecer. Obrigado pela orientação, pelo conhecimento repassado, pelos puxões de orelha e pela paciência. Eu finalizo esse trabalho muito diferente de quando entrei, a aprendizagem proporcionada pelas reuniões e conversas foram de grande valia, e me tornaram uma pessoa e um professor melhor. Agradeço também ao incentivo e apoio em continuar essa jornada, muitas vezes sabemos o quão difícil é, mas é como o pensamento atribuído ao escritor Ernest Hemingway, que afirma que mais importante que a própria guerra é quem se encontra nas trincheiras ao seu lado, e mesmo quando eu estive receoso, você deu o suporte e direcionamento, me passando confiança sobre o processo. Muito obrigado, por todo o tempo em que se dedicou a orientar esse trabalho.

A todos que de alguma forma contribuíram para a realização desta pesquisa, seja com uma dica, um conselho, ou mesmo um incentivo.

RESUMO

O processo de ensino-aprendizagem em disciplinas de programação é complexo, tanto que os índices de aprovação nessas disciplinas são baixos. Este é um desafio que vem sendo constantemente relatado por docentes que ministram cursos de programação. Os conteúdos ensinados apresentam-se de difícil compreensão para os discentes, uma vez que estes necessitam desenvolver habilidades como abstração, resolução de problemas, raciocínio e pensamento lógico. Buscando uma aprendizagem mais eficaz, houve um aumento na adoção de Metodologias Ativas pelos docentes para melhorar o engajamento e a motivação dos discentes. Dentre estas, a Aprendizagem Baseada em Exemplos (ABE) é uma metodologia eficaz e eficiente para ensinar aos discentes novas habilidades de resolução de problemas. A ABE apoiada pela técnica de *Worked Examples* (WE) que fornece a apresentação de um problema, os passos para resolvê-lo e, por fim, uma solução final. Aliado a isso, as tecnologias educacionais têm evoluído a cada dia, sendo uma excelente forma de atrair e engajar estudantes. Uma dessas tecnologias emergentes são os *chatbots*, que são aplicações que simulam uma conversa por meio de linguagem natural e/ou dados pré-definidos. Nesse sentido, esta dissertação de mestrado apresenta o assistente virtual denominado CoderBot que possui o intuito de apoiar a aprendizagem de programação em cursos de graduação. Para atingir este objetivo o CoderBot aplica a ABE com apoio da técnica de WE utilizando exemplos corretos e incorretos. A fim de avaliar o CoderBot, realizou-se um estudo experimental visando avaliar a viabilidade de aplicação do CoderBot para analisar se o assiste virtual é viável e possui bom desempenho. Com base nos resultados do experimento realizado, percebeu-se que o CoderBot se apresentou como uma ferramenta eficiente e prática para apoiar o aprendizado de programação, promovendo compreensão, resolução de tarefas e um ambiente de uso confortável. Com boa usabilidade e impacto positivo na motivação e desempenho dos estudantes, a ferramenta facilita a aprendizagem, embora haja espaço para melhorias em aspectos como inovação e suporte a iniciantes.

Palavras-chave: Assistente Virtual; Ensino de Programação; Aprendizagem Baseado em Exemplos; *Worked Examples*.

ABSTRACT

The teaching-learning process in programming courses is so complex that the approval rates in these courses are low. This is a challenge frequently reported by educators teaching programming courses. The content taught is often difficult for students to comprehend, as it requires them to develop skills such as abstraction, problem-solving, reasoning, and logical thinking. To foster more effective learning, there has been an increase in the adoption of Active Methodologies by educators to improve student engagement and motivation. Among these methodologies, Example-Based Learning (EBL) is an effective and efficient approach to teaching students new problem-solving skills. EBL is supported by the Worked Examples (WE) technique, which presents a problem, the steps to solve it, and, finally, a complete solution.In addition, educational technologies have been evolving daily, serving as an excellent means of attracting and engaging students. One of these emerging technologies is chatbots, which are applications that simulate conversations through natural language and/or predefined data. In this context, this master's thesis presents a virtual assistant called CoderBot, designed to support programming learning in undergraduate courses. To achieve this goal, CoderBot applies EBL with the support of the WE technique, using both correct and incorrect examples. An experimental study was conducted to assess the feasibility and performance of CoderBot as a virtual assistant. Based on the experiment results, CoderBot proved to be an efficient and practical tool for supporting programming learning, promoting comprehension, task resolution, and a comfortable user environment. With good usability and a positive impact on students' motivation and performance, the tool facilitates learning. However, there is room for improvement in aspects such as innovation and support for beginners.

Keywords: Virtual Assistants; Teaching Programming; Example-Based Learning; Worked Examples.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ATQ	Attitude Questionnaire
ABE	Aprendizagem Baseada em Exemplos
BiLSTM-CRF	Bi-directional Long Short-term Memory Network e Conditional Random Field
CQA	Critérios da Avaliação da Qualidade
CE	Critérios de Exclusão
CI	Critérios de inclusão
DSR	Design Science Research
EBL	Example-Based Learning
FLQ	Flow Experience Questionnaire
FSS	Flow State Scale
GPT	Generative Pre-trained Transformer
HTML	Hypertext Markup Language
IES	Instituições de Ensino Superior
IFMS	Instituto Federal de Mato Grosso do Sul
IFPA	Instituto Federal do Pará
IDE	Integrated Development Environment
IA	Inteligência Artificial
IMI	Intrinsic Motivation Inventory
LLM	Large Language Models
PNL	Processamento de Linguagem Natural
MSL	Mapeamento Sistemático da Literatura
NLU	Natural Language Understanding
PHP	Hypertext Preprocessor
PLAQ	Perceived Learning Assessment Questionnaire

PICOC	População, Intervenção, Comparação, Resultado e Contexto
POO	Programação Orientada a Objetos
QP	Questão de Pesquisa
SBIE	Simpósio Brasileiro de Informática na Educação
SQL	Structured Query Language
SQs	Subquestões de Pesquisa
SUS	System Usability Scale
TAM	Modelo de Aceitação de Tecnologia
TCLE	Termo de Consentimento Livre e Esclarecido
UFAM	Universidade Federal do Amazonas
UNIPAMPA	Universidade Federal do Pampa
UEQ	User Experience Questionnaire
WE	Worked Examples

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta a introdução desta dissertação de mestrado. Além de contextualizar esta pesquisa, são apresentados os objetivos, a metodologia seguida e a organização deste trabalho.

1.1 Contexto

Os conteúdos de programação são complexos de ensinar e aprender (LUXTON-REILLY, 2016). Consequentemente, as taxas de aprovação nesses cursos são baixas (ALVES; REBOUÇAS; SCAICO, 2019), resultando em altas taxas de retenção que podem implicar diretamente na evasão dos alunos (PENNEY et al., 2023). Esse desafio tem sido constantemente relatado por diversos professores que ministram cursos na área de Computação (ROBINS, 2019). Isso pode estar relacionado ao fato de que o conteúdo ensinado é complexo para os alunos entenderem, uma vez que eles precisam desenvolver habilidades de abstração, resolução de problemas, raciocínio e pensamento lógico (TURPEN; DANCY; HENDERSON, 2016; LUXTON-REILLY et al., 2018). Em particular, ensinar programação pode ser desafiador porque os alunos muitas vezes lidam com conceitos abstratos e sintaxe da linguagem de programação, exigindo paciência e explicações claras (LÓPEZ-PERNAS et al., 2019). Além disso, as diversas origens e níveis de habilidade dos alunos podem complicar ainda mais o processo de ensino, pois alguns destes podem precisar de orientação personalizada (GUO, 2018), principalmente nos estágios iniciais do curso (MORAIS, 2022).

Manter os estudantes motivados e engajados pode ser uma tarefa difícil quando eles são confrontados com tarefas complexas de resolução de problemas (ROBINS, 2019). Além disso, a falta de contexto do mundo real pode tornar difícil para os alunos perceberem as aplicações práticas da programação, tornando importante que os professores forneçam exemplos práticos e relevantes para superar esta dificuldade (LUXTON-REILLY et al., 2018). Nessa perspectiva, os professores têm repensado a forma de ensinar os conteúdos para obter resultados de aprendizagem mais eficazes (CALDERON; SILVA; FEITOSA, 2021; GUERINO et al., 2021). Portanto, percebe-se um crescimento na adoção de Metodologias Ativas como alternativa pedagógica para auxiliar no engajamento e a motivação dos alunos (CALDERON; SILVA; FEITOSA, 2021). De forma complementar, os professores também têm utilizado tecnologias educacionais para apoiar o ensino

de programação, uma vez que elas promovem a colaboração, a cooperação e a interação entre os alunos (MAGEIRA et al., 2022; CALDERON; SILVA; FEITOSA, 2021). Portanto, tais tecnologias visam melhorar e otimizar a experiência de aprendizagem dos alunos (MALIK et al., 2022).

Nesse sentido, uma importante tecnologia educacional emergente a ser explorada são os *chatbots*, que são aplicações inteligentes que interagem com os usuários em diversos aspectos da vida diária por meio de linguagem natural e/ou dados predefinidos (MAGEIRA et al., 2022; DAHIYA, 2017). O *chatbot* processa a mensagem que o usuário recebe e retorna uma resposta correspondente à solicitação do usuário (DAHIYA, 2017). A utilização de ferramentas e recursos educacionais adicionais tem sido uma estratégia utilizada por docentes para propor um aprendizado mais dinâmico e envolvente além de oferecer um suporte eficaz e de alta disponibilidade (ALVES; NASCIMENTO; SOUSA, 2021). Isso acontece devido a esses sistemas ficarem à disposição dos estudantes praticamente o tempo todo, onde eles podem buscar respostas imediatas às suas dúvidas, ou mesmo, relembrar conceitos que foram ensinados em sala de aula (BII, 2013; OKONKWO; ADE-IBIJOLA, 2021). Outra vantagem do uso de *chatbots* é a sua possibilidade de personalização da forma de *feedback*, podendo incluir aspectos emocionais e motivacionais fornecendo uma maior interação entre o estudante e a ferramenta, além de, devido a essa disponibilidade, cada aluno pode estudar no seu tempo, de acordo com seu próprio ritmo (ROCA et al., 2024).

Mesmo diante de tantos pontos convergentes é possível observar que os *chatbots* possuem limitações, no que tange seu uso de conjunto de dados, que nem sempre se apresentam atualizados, podendo conter imprecisões ou inconsistências (CORREIA et al., 2024). Alguns *chatbots* também possuem algumas incongruências em relação a formalidade excessiva no tratamento e/ou falta de didática no fornecimento das respostas (CHAVES, 2023). Além disso, muitas dessas ferramentas tem por objetivo apenas a entrega da solução focando em produtividade e não em colaborar com a aprendizagem por meio de orientações didáticas (PENNEY et al., 2023). Essa forma de entrega da solução sem as devidas explicações acabam por não estimular o pensamento computacional e até mesmo pode corroborar com uma desmotivação do estudante que não compreende a solução por inteiro devido a falta de conhecimento em elementos de sintaxe das linguagens de programação (EDWARDS et al., 2020). Diante dessas informações é possível observar a necessidade de explorar uso de *chatbots* no contexto educacional visando a construção de ferramentas de apoio a um aprendizado eficaz e envolvente (PENNEY et al., 2023).

Apesar do cenário acima, há evidencias na literatura de que os *chatbots* têm sido aplicados como agentes pedagógicos, especialmente no ensino de computação (como Teste de Software (PASCHOAL et al., 2022), Pensamento Computacional (ALMEIDA; ARAÚJO, 2021) e Programação (CARREIRA et al., 2022). Eles possibilitam que os alunos sejam mais ágeis no processo de aprendizagem, tirem dúvidas específicas (CLARIZIA et al., 2018; RUAN et al., 2019) e obtenham um *feedback* imediato e adequado sem uma forte dependência por parte dos professores (SMUTNY; SCHREIBEROVA, 2020), tornando o processo de aprendizagem mais inspirador e personalizado (CLARIZIA et al., 2018; RUAN et al., 2019). Uma percepção vantajosa da utilização de *chatbots* é a redução de obstáculos, melhoria do processo de aprendizagem, redução da resistência do aluno em relação à complexidade do conteúdo e, por fim, nota-se uma redução na necessidade de envolvimento do professor (LIDÉN; NILROS, 2020).

Diante das informações dispostas acima, este trabalho apresenta o CoderBot, um assistente virtual com aspecto pedagógico educacional para apoiar estudantes de graduação durante a aprendizagem de programação. O CoderBot é fundamentado na Aprendizagem Baseada em Exemplos (ABE), também conhecida como *Worked Examples* (WEs), e integra exemplos práticos de programação corretos e incorretos para potencializar a experiência de aprendizagem. Esta ferramenta tecnológica emerge facilitando novas abordagens de ensino e aprendizagem para docentes e estudantes iniciantes.

1.2 Objetivo geral

O objetivo principal desta dissertação de mestrado consiste em apoiar os estudantes de graduação na aprendizagem de conteúdos de programação por meio da disponibilização de um assistente virtual, denominado CoderBot, que utiliza da Abordagem Baseada em Exemplos como estratégia educacional.

1.3 Objetivos específicos

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

- Identificar, por meio de um mapeamento sistemático da literatura (MSL), o corpo de conhecimento existente sobre *chatbots* propostos e adotados na aprendizagem de programação em cursos de graduação, de forma a fundamentar o desenvolvimento

desta pesquisa;

- Desenvolver um *template* que sirva como guia para docentes na criação e no uso de exemplos aplicados ao ensino de programação, contribuindo para práticas pedagógicas mais eficazes;

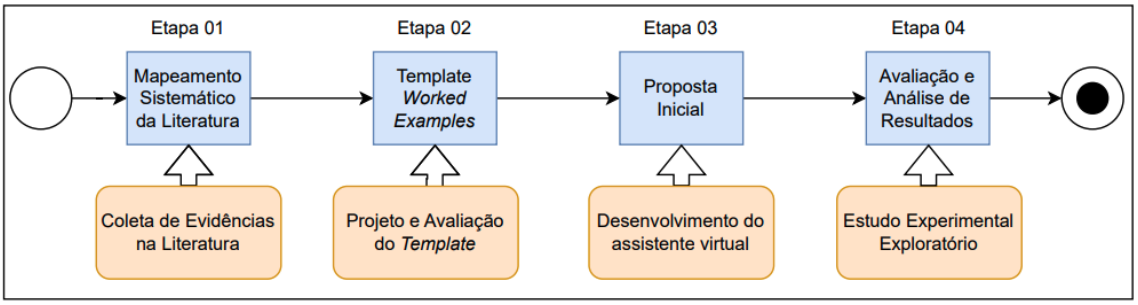
- Projetar e implementar um assistente virtual para uma plataforma *Web*, utilizando os exemplos estruturados com base no *template* proposto, para apoiar o aprendizado de programação;

- Avaliar a percepção dos estudantes em relação ao uso do CoderBot como ferramenta de apoio à aprendizagem de programação, utilizando um estudo experimental para validar sua eficácia.

1.4 Metodologia

Para alcançar o objetivo geral desta proposta de mestrado, seguiu-se as atividades apresentadas na Figura 1.

Figura 1 – Metodologia adotada na pesquisa.



Fonte: De autoria própria.

Diante disso, primeiramente, decidiu-se investigar estudos que abordassem a proposição e a adoção de *chatbots* ou assistentes virtuais no contexto de ensino de programação nos cursos de graduação (Etapa 01). A identificação desses estudos foi conduzida por meio de um abrangente Mapeamento Sistemático da Literatura (MSL), que possibilitou a extração de dados cruciais para uma compreensão mais aprofundada do tema em análise. Uma das lacunas identificadas foi a falta de um padrão para o uso de *worked examples* na apresentação de conteúdos voltados à aprendizagem de programação em *chatbots*. Como

resultado, percebeu-se que não há uma forma padronizada de elaborar os exemplos a serem apresentados.

Tal informação implicou na necessidade da realização de um outro estudo, visando o projeto e avaliação de um *template* para apoiar a construção de *worked examples* voltados para o ensino de programação (Etapa 02). Os dados e informações foram extraídos durante um experimento realizado com docentes atuantes em cursos de graduação na área de Computação e indicam os fatores que são importantes para construir um exemplo voltado ao ensino de programação.

A partir dos estudos realizados, a proposta inicial deste trabalho foi definida (Etapa 03), com o objetivo de desenvolver um assistente virtual que apoiasse a aprendizagem de programação para alunos de graduação. A abordagem aplicada baseia-se em exemplos, demonstrando situações elaboradas conforme o *template* desenvolvido no estudo 4.

O *template* foi analisado e adequado ao contexto da aplicação, sendo trabalhado em duas vertentes: a forma correta, na qual o aluno busca aprender ou relembrar como resolver um determinado problema, e a forma incorreta, na qual é apresentado um exemplo contendo erros, cabendo ao aluno a atividade de identificar o erro. Ambas as escolhas contam com um *feedback* da ferramenta, que explica detalhadamente como foi realizada a resolução do problema.

Em seguida, realizou-se a etapa de planejamento e execução do estudo experimental exploratório (Etapa 04), no qual o assistente virtual CoderBot foi aplicado em turmas de graduação de cursos de computação. O experimento incluiu formulários de consentimento dos participantes para a realização do experimento, um formulário para caracterização dos participantes e um documento-guia para o docente responsável pela turma. Após essa etapa, foi entregue aos estudantes uma lista de exercícios, que deveriam ser resolvidos exclusivamente com o apoio do CoderBot. Ao final da lista de exercícios, os estudantes receberam novos formulários para avaliar aspectos relacionados à percepção de aprendizagem, às emoções experimentadas durante a execução e, por fim, questionários destinados a avaliar a percepção dos estudantes em relação ao uso da tecnologia.

1.5 Organização do Trabalho

Esta dissertação está organizada em outros cinco capítulos, além deste primeiro capítulo que apresentou os aspectos de importância para construção deste trabalho. A organização do texto segue a estrutura abaixo:

- Capítulo 2 – Fundamentação Teórica: apresenta um embasamento teórico sobre os assuntos que fazem parte da concepção do assistente virtual bem como o *background* necessário em relação ao ensino de programação;
- Capítulo 3 – Mapeamento Sistemático da Literatura: desenvolvimento do mapeamento da literatura sobre os *chatbots* existentes que apoiam a aprendizagem de programação em cursos de graduação;
- Capítulo 4 – Apresentação do estudo que formulou a construção de um *template* projetado com o objetivo de auxiliar os docentes a padronizar o uso de WE no ensino de programação;
- Capítulo 5 – Proposta inicial do assistente virtual destinado a apoiar a aprendizagem de programação por meio de abordagem baseada em exemplos;
- Capítulo 6 – Conclusão: este capítulo contém a avaliação e análise do experimento, bem como conclusões e contribuições deste trabalho, além de indicar a continuação desta pesquisa através de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a formação do embasamento teórico visando definir aspectos inerentes a Ensino de Programação, Chatbots Educacionais, Abordagem Baseada em exemplos e *Worked Examples*.

2.1 *Ensino de Programação*

Ter conhecimento em programação não é algo que está restrito atualmente a cientistas e engenheiros. Com a globalização tecnológica, o desenvolvimento de pensamento computacional é de grande importância para a era atual, tornando assim, a programação útil a diversos tipos de profissionais (YANG; LIN, 2024).

Dohmke, Iansiti and Richards (2023) indicam uma onda crescente de digitalização em andamento, isso nos mais diversos tipos de empresas e industrias, o que implica na necessidade de cada vez mais programadores. Zhao et al. (2022) observam um aumento na quantidade de estudantes matriculados em cursos da área da computação nos últimos anos, entretanto muitos desses estudantes acabam reprovando, ou até mesmo evadindo do curso em seu primeiro ano ao enfrentar módulos de programação. Na literatura encontramos explicações sobre a ocorrência desse tipo de situação, tais como reportadas por Robins, Rountree and Rountree (2003), afirmando que aprender programação é difícil, tanto que cursos de programação costumam ter altas taxas de evasão, pois estudantes novatos possuem déficits em aspectos que são necessários para programação, como por exemplo, matemática básica, cálculos, ou mesmo noções de lógicas. Chao (2016) demonstra que, mesmo ao final desses cursos introdutórios, os alunos ainda apresentam dificuldades em decomposição de problemas, projeção e mesmo resolver seus problemas usando programação. Ramalingam, LaBelle and Wiedenbeck (2004) explicam dizendo que programadores precisam desenvolver habilidades de abstração, muito demandado pelos usos de estruturas lógicas.

Nesse sentido, alguns autores comentam que a iniciação em programação deve ocorrer desde cedo, por exemplo, no ensino fundamental, tendo em vista os benefícios e o pensamento crítico desenvolvidos (YANG; LIN, 2024). O pensamento computacional possui uma carga cognitiva de grande importância na resolução de problemas, o que tem chamado bastante atenção de pesquisadores e profissionais (LAI; WONG, 2022).

Diante do cenário descrito acima, nota-se que programação de computadores se

tornou uma habilidade importante para que os estudantes possam enfrentar os desafios do mundo contemporâneo por meio do uso de ferramentas computacionais (CHAO, 2016). Nota-se ainda que há um trabalho significativo de educadores e pesquisadores em educação para fomentar o ensino de programação de computadores e reconhecê-la como uma competência importante a ser desenvolvida (YANG; LIN, 2024).

Oroma, Wanga and Ngumbuke (2012), em seu estudo sobre as dificuldades em aprender a programar, mostram uma ausência de motivação, de certa forma, pela própria dificuldade da linguagem, bem como, de formas de aprendizagem que não favorecem habilidades cognitivas e metacognitivas, o que conclui por um esforço voltado a memorização de códigos e não a aprendizagem profunda. Em frente a essas informações é possível afirmar a necessidade de novas abordagens para contemplar as carências observadas na aprendizagem em programação.

O avanço em produção de tecnologia educacional trouxe ferramentas capazes de ampliar conteúdos e capturar informações relevantes sobre aspectos de aprendizagem (GONDA; CHU, 2019). Essa evolução busca por entender e propor experiência personalizada para apoiar os estudantes em suas necessidades particulares de aprendizagem (WANG et al., 2016). Rezende and Barros (2008) averiguaram a forma de navegação de estudantes de diferentes níveis de conhecimento, utilizando um software online, onde os autores denotaram ser um suporte a aprendizagem. Outro aspecto identificado foi a utilização de recursos interativos que possibilitam ao estudante realizar a escolha, sequência ou mesmo tempo dedicado para aprender um determinado conteúdo (REZENDE; BARROS, 2008).

2.2 *Chatbots Educacionais*

Uma tecnologia de destaque atualmente são os *chatbots*, que propõem assistentes virtuais para facilitar trabalhos e atividades cotidianas, oferecendo uma aplicação escalável, acessível e com grande potencial de personalização, contendo bastante potencial para apoiar aprendizagem na educação (WOLLNY et al., 2021). Os *chatbots* se tornam boas escolhas de ferramentas tecnológicas para apoiar a aprendizagem devido à sua simplicidade e personalização em relação a outras aplicações, além de sua intuitividade e interação baseada em comunicação visual (FADHIL; VILLAFIORITA, 2017).

Chatbots são ferramentas tecnológicas capazes de simular um diálogo de forma natural, por texto ou voz, e têm sido aplicados nos mais diversos domínios, como

Marketing, Atendimento ao Cliente, Suporte Técnico e, especialmente, na Educação (SMUTNY; SCHREIBEROVA, 2020). Hiremath *et al.* (2018) corroboram indicando que os *chatbots* são programas simples, onde todo o conhecimento já se encontra programado, bastando o uso de métodos como correspondência de padrões, processamento de linguagem natural e mineração de dados para efetuar a correspondência entre a entrada informada pelo usuário e o conhecimento já disponível na ferramenta.

Ondáš *et al.* (2019) indicam que um dos principais focos da aplicação de *chatbots* foi para tentar suprir a carga humana destinada em suporte de programas de computador, em que muitas soluções para problemas corriqueiros de software, podem ser definidas em um passo-a-passo onde o usuário pode seguir ou verificar as informações úteis, e caso o problema não seja resolvido ele será redirecionado a um auxílio humano. Há registros de *chatbots* sendo usados como complemento a uma aplicação maior, muitos deles servido de funções de confirmações de agendamento, efetuar reservas em estabelecimentos, ou até mesmo disseminar conteúdos (SMUTNY; SCHREIBEROVA, 2020).

Diante da perspectiva de ferramentas que prestam assistência ao usuário, é evidente a necessidade de expandir esse tipo de tecnologia em diversas áreas, entre elas, a Educação. Mondal *et al.* (2018) destacam que, frequentemente, membros de instituições educacionais precisam estabelecer contato para resolver procedimentos via *Web*, seja pela distância ou pela diferença de fuso horário entre estudante e instituição. Nesse contexto, nem sempre há alguém disponível para atendimento, o que já justifica a necessidade de suporte automatizado. Além disso, quando integrado ao ensino, o uso de *chatbots* possibilita uma maior interação entre discentes matriculados em cursos de graduação, proporcionando um *feedback* adequado para esclarecer dúvidas e auxiliar no aprendizado (PASCHOAL et al., 2022).

Na literatura é possível encontrar diversos casos de *chatbots* empregados como agentes pedagógicos. O estudo de Wollny et al. (2021) demonstra uma grande quantidade de *chatbots* destinados a aprendizagem de idiomas (mais de 50% dos *chatbots* encontrados), programação, matemática, habilidades de comunicação, tecnologias educacionais, leis, habilidades de escrita, psicologia e administração de computadores.

No ensino de Computação, há evidências do uso desses agentes em áreas como testes de software (PASCHOAL et al., 2022) e desenvolvimento do pensamento computacional (ALMEIDA; ARAÚJO, 2021). Esses agentes permitem que os estudantes acelerem seu processo de aprendizagem (CLARIZIA et al., 2018; RUAN et al., 2019), solucionem dúvidas pontuais e recebam *feedback* imediato, reduzindo a dependência de suporte

constante por parte dos docentes (SMUTNY; SCHREIBEROVA, 2020). Isso torna a experiência de aprendizagem mais motivadora e adaptada às necessidades individuais dos estudantes (CLARIZIA et al., 2018; RUAN et al., 2019).

2.3 *Worked Examples*

A Aprendizagem Baseada em Exemplos (ABE), também conhecido como *Worked Examples* (WE), está fundamentado na Teoria da Carga Cognitiva, que defende que o ensino é aprimorado quando não há uma sobrecarga de informações, permitindo que o foco seja mantido nos pontos-chave do problema (SWELLER; MERRIENBOER; PAAS, 1998). O uso de WE auxilia na redução da carga cognitiva, fornecendo uma estrutura a ser seguida para alcançar o resultado esperado, o que beneficia a compreensão e o entendimento do conteúdo em questão. Existem diversas abordagens para realizar demonstrações de WE no processo de aprendizagem, uma delas, ao invés de apresentar uma solução didática correta opta por utilização de exemplos errôneos objetivando que os estudantes encontrem e corrijam os erros (GROSSE; RENKL, 2007), ou mesmo para obter um *feedback* adicional (KOPP; STARK; FISCHER, 2008) visando estimular os estudantes a processar os exemplos mais profundamente.

Os *worked examples* são ferramentas úteis para discentes e até mesmo para aqueles que estão aprendendo de forma autodidata. Podem ser moldados para se adequarem a diversas realidades educacionais. O estudo dos *worked examples* é um tema relevante para compreender e aprimorar as práticas de ensino e aprendizagem. O uso de WE pode ser evidenciado em diversas áreas do conhecimento, como Matemática (MCGINN; LANGE; BOOTH, 2015), Física (CARTWRIGHT; MCMULLIN, 1984), Química (MONK, 2008), Computação (ABDUL-RAHMAN; du Boulay, 2014), entre outras áreas. A seguir vamos apresentar alguns trabalhos mais relevantes que utilizaram exemplos como modelo de aprendizagem para o ensino de programação.

Garces *et al.* (2023) realizaram um experimento para explorar o uso de exemplos práticos no ensino de programação. O experimento avaliou como a forma de aprendizagem pode afetar os estudantes no processo de aprendizagem, utilizando estratégias de *debugging* (ato de identificar, analisar e corrigir erros por meio de depuradores) e *self-explaining* (escrita de código claro, sem necessidade de muitos comentários externos). Os participantes eram de duas turmas de um curso de computação e foram divididos em dois grupos: um com estudantes que já possuíam experiência em programação e outro

sem experiência. Em cada turma foi aplicada uma abordagem para o uso dos exemplos. Embora as estratégias tenham sido úteis para o ensino de programação, os resultados não mostraram uma grande disparidade entre elas. No entanto, independentemente do estudante já possuir ou não experiência, a estratégia de *self-explaining* foi mais eficaz do que a de *debugging*.

Hosseini *et al.* (2020) compararam *worked examples* com métodos não interativos convencionais. Os autores conduziram um experimento controlado, dividindo os estudantes em dois grupos, um para cada método de exemplo. A avaliação consistiu em comparar resultados com base no engajamento, desempenho na resolução de problemas e aprendizado. Os resultados mostraram que os estudantes que tiveram contato com métodos iterativos tiveram um desempenho relativamente maior na resolução de problemas, porém levaram mais tempo para resolver os subproblemas de construção de programas. Quanto ao aprendizado, não houve diferença significativa entre os dois grupos.

Diante dos trabalhos mencionados acima, é possível identificar um interesse e verificar que estão ocorrendo pesquisas sobre utilização de WE para ensino e aprendizagem de programação. Esses estudos corroboram a observação de um campo ainda recente a ser explorado. O potencial relatado por esses estudos justifica a necessidade de novas pesquisas no que se refere ao ensino e aprendizagem por meio de aplicação de *worked examples*. Outro fator identificado neste estudo é que cada grupo de autores definiu suas próprias regras para construção um *worked examples*, o que evidencia um nicho a ser explorado em nossa pesquisa, ou seja, a proposição de um *template* que possa orientar os docentes na produção de *worked examples* para ensino de programação.

2.4 *Considerações Finais*

A formação do embasamento teórico, traz informações de grande importância para a construção deste trabalho. É notável a necessidade de melhoria em recursos tecnológicos para apoiar a aprendizagem de programação, os aspectos de dificuldade explicitam bem essa situação. O alinhamento entre uso de assistentes virtuais (*chatbots*) e metodologias ativas, como a aprendizagem baseada em exemplos, podem trazer grandes benefícios a aprendizagem.

3 MAPEAMENTO SISTEMÁTICO DA LITERATURA

Este capítulo apresenta um Mapeamento Sistemático da Literatura sobre *chatbots* que apoiam o processo de ensino e aprendizagem de programação. Os resultados deste estudo fornecem uma visão geral, abrangente e replicável do estado da arte, ajudando pesquisadores e profissionais a encontrar características, limitações e lacunas nas pesquisas atuais sobre o tema.

3.1 Introdução

Foi conduzido um estudo de Mapeamento Sistemático da Literatura (MSL) considerando as diretrizes fornecidas por Petersen, Vakkalanka and Kuzniarz (2015) e Kitchenham and Charters (2007) visando investigar os *chatbots* propostos na literatura que apoiam o aprendizado de programação em cursos de graduação. Os dados foram extraídos a partir do uso da ferramenta Parsifal[1], sendo as buscas realizadas em 17 de abril de 2023.

O MSL foi realizado em três etapas: planejamento, condução e relato dos resultados. As atividades relativas às etapas de planejamento e condução do estudo de MSL são descritas nas subseções a seguir, e a etapa de relatório é apresentada na Seção 3.3.

3.2 Planejamento do MSL

Essa seção apresenta a descrição das etapas referentes ao planejamento do MSL.

3.2.1 Questões de Pesquisa

A seguinte Questão de Pesquisa (QP) foi definida para alcançar o objetivo proposto:

QP: Como os *chatbots* têm sido adotados no ensino de programação em cursos de graduação?

[1] <https://parsif.al/>

O objetivo ao responder a esta QP foi identificar e analisar três aspectos distintos dos estudos selecionados:

- **Contexto:** Quais *chatbots* têm sido adotados para o ensino de programação?

- **Desenvolvimento:** Quais tecnologias são utilizadas no desenvolvimento dos *chatbots* propostos;

- **Estudos Empíricos:** Os pesquisadores realizaram algum estudo experimental com *chatbots*?

Com base nesses três aspectos, definiram-se Subquestões de Pesquisa (SQs) para responder a questões específicas de cada aspecto. A Tabela 1 apresenta as sub-questões de pesquisa que norteiam este MSL, assim como as motivações subjacentes para cada questão.

Tabela 1 – Subquestões de Pesquisa e Motivação.

Aspecto	Subquestões de Pesquisa	Motivação
Contexto	**SQ1.** Quais os *chatbots* existentes na literatura que apoiam na aprendizagem de programação?	Investigar os *chatbots* adotados por instrutores durante cursos de graduação em programação.
	SQ2. Qual o contexto/conteúdo ensinado por meio do chatbot?	Investigar o conteúdo ou tópico de programação apoiado pelo *chatbot* educacional.
	SQ3. Qual linguagem de programação foi empregada pelo *chatbot* para apoiar na aprendizagem?	Investigar qual linguagem de programação é mais empregada em chatbots educacionais para fins de ensino de programação.
Desenvolvimento	**SQ4.** Quais as ferramentas (linguagens e *frameworks*) utilizadas para desenvolver esses *chatbots*?	Explorar as ferramentas que os pesquisadores empregam no desenvolvimento de *chatbots*, incluindo linguagens de programação, frameworks e outras informações.
	SQ5. Como os *chatbots* educacionais interagem?	Para identificar como os *chatbots* interagem, por exemplo, Processamento de Linguagem Natural (PNL), Compreensão de Linguagem Natural (NLU) e outros.
	SQ6. Qual o tipo de licença do *chatbot*?	Para descobrir se os *chatbots* educacionais estão disponíveis para uso.
Estudos Empíricos	**SQ7.** O artigo apresenta uma avaliação empírica? Se sim, que tipo de estudo empírico foi realizado?	Determinar o principal tipo de pesquisa que pesquisadores e instrutores têm realizado em sala de aula com *chatbots* educacionais.
	SQ8. Qual período de aplicação do uso do *chatbot*?	Verificar se a aplicação do *chatbot* educativo com os alunos foi realizada de forma episódica ou longitudinal.
	SQ9. Quais métricas foram empregadas para avaliar o *chatbot*?	Encontrar as métricas utilizadas pelos pesquisadores para avaliar o desempenho de aprendizagem dos alunos usando *chatbots* educacionais.
	SQ10. Qual a modalidade de ensino em que foi aplicado?	Identificar as modalidades de ensino nas quais os *chatbots* educacionais têm sido utilizados para o ensino de programação.
	SQ11. Qual o perfil do estudante durante a aplicação do *chatbot* na aprendizagem?	Para obter *insights* sobre dados demográficos, experiência e contexto dos perfis dos participantes.

Fonte: De autoria própria.

3.2.2 Escopo da Pesquisa

As principais bibliotecas digitais utilizadas para conduzir este MSL foram: ACM Digital Library (ACM)[2], Engineering Village (Compendex)[3], Biblioteca Digital IEEE Xplore (IEEE)[4] e Scopus[5]. Estas bibliotecas foram escolhidas porque (1) permitem um bom funcionamento e abrangência dos motores de busca; (2) ACM também indexa algumas publicações da Springer Link, Science Direct e a maioria das publicações relacionadas ao ensino de Ciência da Computação; e (3) Scopus é o banco de dados mais extenso que indexa resumos e citações indexa publicações de diversas editoras conhecidas (por exemplo, Springer, Elsevier e Taylor & Francis) (KITCHENHAM; CHARTERS, 2007; NAKAMURA et al., 2022).

3.2.3 Idioma

O idioma selecionado para a realização das buscas foi o inglês, uma língua universal que permite aos pesquisadores conectar-se e colaborar com colegas de todo o mundo. A colaboração e a comunicação com outros pesquisadores são essenciais para o crescimento e o sucesso intelectual. Estudos em inglês facilitam o compartilhamento e a divulgação dos resultados de pesquisa, fornecendo informações detalhadas sobre métodos e resultados. Esses estudos permitem que pesquisadores repliquem e validem os resultados, beneficiando colegas na replicação ou ampliação de estudos de MSL.

3.2.4 Estratégia de Busca

O mecanismo de busca disponível em grande parte das bibliotecas digitais é baseado em expressões de busca textual. Assim, a definição da *string* de busca é essencial para a eficácia dessa etapa e para o MSL (STEINMACHER et al., 2015). Os termos de busca foram definidos com base no procedimento descrito por Keele et al. (2007), que sugere a definição de parâmetros para População, Intervenção, Comparação, Resultado e Contexto (PICOC). O PICOC foi desenvolvido para identificar palavras-chave e formular *strings*

[2]<http://dl.acm.org/>
[3]<https://www.elsevier.com/solutions/engineering-village>
[4]<https://www.ieee.org/>
[5]<https://www.scopus.com/search/>

de busca a partir da questão de pesquisa. Em nossa pesquisa, o PICOC foi aplicado da seguinte forma:

- **P***opulation*: estudos que apresentam o uso de *chatbots* educacionais para apoio ao aprendizado de programação em cursos de graduação;
- **I***ntervention*: *chatbots* educacionais empregados para ensinar ou aprender programação.
- **C***omparison*: não aplicável, pois o objetivo não é comparar *chatbots* educacionais, mas sim caracterizá-los.
- **O***utcome*: melhorar o aprendizado dos alunos por meio de *chatbots* educacionais.
- **C***ontext*: ensinar ou aprender programação em cursos de graduação.

A consulta foi iterada diversas vezes para garantir o uso de um conjunto abrangente de sinônimos, permitindo uma alta cobertura e, ao mesmo tempo, mantendo o número de documentos recuperados sob controle. Um processo de refinamento semelhante ao de Zhang, Kitchenham and Pfahl (2010) foi adotado para incluir novos termos de pesquisa de estudos previamente selecionados e verificar se os estudos de controle retornavam nas strings de consulta testadas. Após as iterações, os pesquisadores alcançaram uma cobertura satisfatória de estudos com a *string* de busca nas bibliotecas digitais.

A *string* de busca apresentada abaixo foi utilizada para pesquisar automaticamente em bibliotecas digitais, usando o operador booleano OR para combinar grafias alternativas e sinônimos. Simultaneamente, adotou-se o operador booleano AND para unir os três conceitos. Além disso, é importante destacar o uso do asterisco (“*”), que permite incluir qualquer variação da palavra no termo de pesquisa (por exemplo, o termo “agente conversaciona*” inclui “agente conversacional” e “agentes conversacionais”).

(“chatbot” **OR** “chatterbot” **OR** “artificial conversational entity” **OR** “chatbots” **OR** “mobile chatbots” **OR** “conversational agent*” **OR** “talkbot*” **OR** “talk bot*” **OR** “conversational interface” **OR** “conversational system” **OR** “dialogue system”) **AND** (“programming*” **OR** “program” **OR** “CS2” **OR** “CS1” **OR** “computer programming” **OR** “introductory computer” **OR** “introductory programming” **OR** “novice programming” **OR** “coding education” **OR** “introductory computer science”) **AND** (“student” **OR** “learning” **OR** “course” **OR** “teaching” **OR** “training”)

3.2.5 Critérios para Seleção dos Estudos

Independentemente do mecanismo utilizado para realizar a busca, os estudos foram selecionados de acordo com diversos critérios relevantes para a QP. Os critérios de inclusão e exclusão foram elaborados seguindo as diretrizes sugeridas por Kuhrmann, Fernández and Daneva (2017). Estabeleceram-se os seguintes critérios de inclusão (CI) dos estudos:

- **CI1**: Estudos que discutem aspectos do uso de *chatbots* para aprender programação em cursos de graduação.
- **CI2**: Estudos contendo experimentos com evidências do uso de *chatbots* para aprendizado de programação em cursos de graduação.
- **CI3**: Estudos apresentando *chatbot* como ferramenta de apoio ao aprendizado de programação em cursos de graduação.

Foram excluídos os estudos que atenderam a pelo menos um dos seguintes critérios de exclusão (EC):

- **EC1**: Estudos que apresentam *chatbots* educacionais, mas não no contexto do ensino de programação.
- **EC2**: Estudos duplicados (por exemplo, um estudo com um trabalho publicado em locais ou datas diferentes). Neste caso, será considerada apenas a versão mais completa e mais recente.
- **EC3**: Estudos que não estão escritos em língua inglesa.
- **EC4**: Os seguintes tipos de publicações: livros, teses de doutorado, dissertações de mestrado, patentes, tutoriais, propostas de *workshops* ou pôsteres.
- **EC5**: O texto completo do estudo não está disponível para download.

3.2.6 Processo de Seleção

A *string* de busca apresentada na Subseção 3.2.4 foi utilizada para recuperar estudos candidatos a serem analisados no MSL, em abril de 2023. Os estudos passaram

por duas etapas distintas de filtragem, em um processo de seleção iterativo e incremental. Dois pesquisadores conduziram a seleção nas duas etapas, discutindo os resultados e resolvendo eventuais conflitos para mitigar preconceitos pessoais. Reuniões específicas foram realizadas para minimizar divergências ou incertezas durante a seleção dos estudos.

Na etapa inicial, os estudos foram filtrados com base em seus títulos e resumos. Estudos cujos títulos e resumos não estavam relacionados a **chatbots** para o ensino de programação em cursos de graduação foram excluídos. Os critérios de inclusão e exclusão, descritos na Subseção 3.2.5, foram aplicados para apoiar a seleção do conjunto final de estudos analisados. A decisão de incluir, excluir ou classificar um estudo na primeira etapa foi conduzida e discutida por dois pesquisadores. Qualquer estudo sobre o qual não houvesse consenso, com base apenas em título e resumo, era mantido para a próxima etapa do processo de seleção.

Quando as informações do título e do resumo foram insuficientes para determinar a relevância de um estudo para a pesquisa, procedeu-se à segunda etapa, que envolveu a leitura completa dos estudos incluídos na primeira etapa. Assim, a segunda etapa teve como objetivo garantir uma análise mais precisa dos estudos. Nessa fase, os pesquisadores aplicaram os critérios de seleção para julgar se os artigos deveriam ou não ser incluídos. Os resultados foram revisados e discutidos, e quaisquer divergências entre os pesquisadores foram resolvidas.

3.2.7 Avaliação da qualidade do estudo

Nesta etapa, foi definido um conjunto de Critérios de Avaliação da Qualidade (CQA) para qualificar e classificar os estudos selecionados no MSL. Esses critérios quantificam a relevância de cada estudo primário e permitem a comparação entre eles (IUNG et al., 2020). Cada CQA recebe um dos três valores de resposta: "**N**ão" (0,0), "**P**arcialmente" (1,5) e "**S**im" (3,0). Consequentemente, cada estudo recebe uma pontuação final que varia de 0,0 (mínimo) a 9,0 (máximo). Para avaliar a qualidade do estudo, estabeleceram-se faixas de pontuação. Foram rotulados estudos com pontuação menor ou igual a 1,0 como "Fracos", aqueles entre 1,5 e 2,0 como "Razoável", aqueles acima de 2,5 e abaixo de 3,1 como "Bom", e aqueles com 3,5 ou superior como "Excelente". Os CQA foram definidos da seguinte forma:

- **QA1.** O estudo apresenta um *chatbot* aplicado em contexto de programação?

  - **Validação**: **S** - O estudo apresenta um *chatbot* que oferece apoio em contexto de programação; **P** - O estudo menciona um *chatbot* que dá apoio em contexto de programação; **N** - O estudo não apresenta ou menciona um *chatbot* que dê apoio em contexto de programação.

- **QA2.** A utilização do *chatbot* está descrita de forma clara e abrangente?

  - **Validação**: **S** - O estudo apresenta detalhadamente o design do *chatbot* e traça um fluxo de trabalho definido para atingir objetivos específicos; **P** - O estudo menciona brevemente certas atividades do *chatbot* sem fornecer maiores explicações ou detalhes; **N** - O estudo não apresenta nenhuma evidência de uso de *chatbot*.

- **QA3.** Existem evidências empíricas que apoiam o uso de *chatbots* para aprendizagem de programação?

  - **Validação**: **S** - O estudo demonstra a aplicação do *chatbot* educacional em um estudo empírico, como experimentos, estudos de caso ou outros métodos de pesquisa; **P** - O estudo menciona brevemente, mas carece de informações detalhadas; **N** - O estudo não apresenta nenhuma evidência empírica.

- **QA4.** Qual é a extensão das citações recebidas pelo estudo de outros autores?

  - **Validação**: **S**: Mais de dez autores citaram o estudo; **P**: Cinco a dez artigos citaram o estudo; **N**: Menos de cinco autores citaram o estudo.

3.2.8 Estratégia de Extração de Dados e Processo de Mapeamento

Os estudos selecionados na segunda etapa do processo de extração de dados foram submetidos a análises. A ferramenta Parsifal (Parsifal) foi utilizada para extrair dados do conjunto final de publicações, definindo campos de extração com base nas questões de pesquisa descritas na Tabela 1. Após uma leitura minuciosa, as informações dos estudos foram extraídas e analisadas para responder às QP deste MSL. Além disso, foram resumidas as possíveis limitações dos estudos revisados, que visavam relatar as tendências e desafios dos trabalhos existentes. Assim, todos os estudos foram verificados pelos outros dois autores para garantir a exatidão dos dados extraídos. Quaisquer conflitos nos dados extraídos foram discutidos e resolvidos.

3.2.9 Replicabilidade

Foram tomadas as medidas necessárias para garantir que outros pesquisadores possam facilmente replicar e ampliar a pesquisa. Para tal está sendo disponibilizado um pacote de replicação abrangente que inclui os resultados completos obtidos do MSL. O repositório completo está disponível em <https://figshare.com/articles/dataset/Pacote_de_Replica_o/27964311> (GARCIA, 2024).

3.3 Resultados

Os resultados deste MSL serão apresentados nas subseções seguintes.

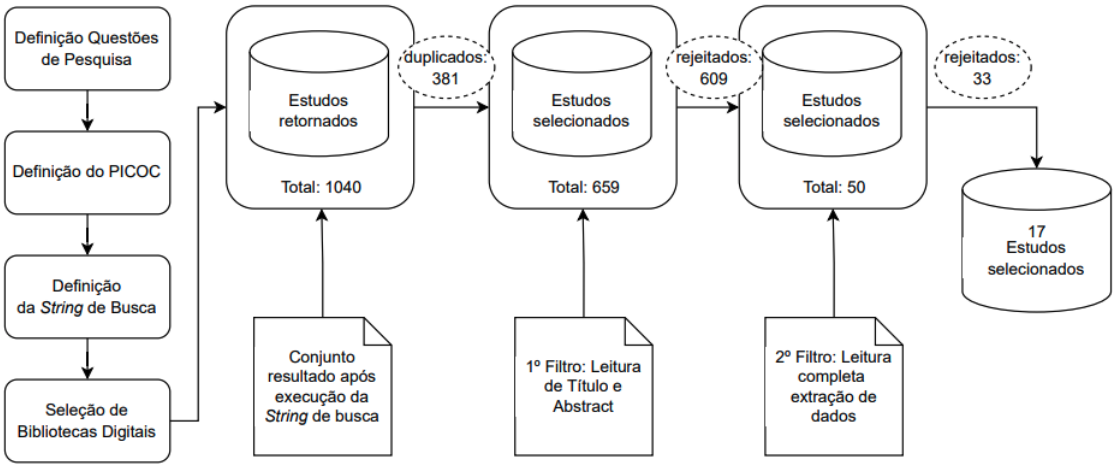
3.3.1 Estudos Primários Selecionados

A Figura 2 resume todo o processo de triagem de artigos. A consulta de pesquisa retornou 1.040 publicações: 61 na ACM, 264 no Engineering Village, 169 no IEEE e 546 no Scopus. Alguns artigos apareceram em várias bibliotecas digitais, mas consideramos cada artigo apenas uma vez nesse caso, de acordo com a ordem de busca (Biblioteca Digital ACM, seguida por Engineering Village, IEEE e Scopus). Todos os artigos potenciais foram acessados usando a conexão à Internet e a autenticação fornecida pela Universidade Federal do Pampa (UNIPAMPA). Quando um artigo potencial não estava disponível gratuitamente na rede da instituição, o autor do estudo foi contatado para solicitar uma pré-impressão.

Após a remoção das publicações duplicadas, o número de publicações selecionadas para o primeiro filtro foi de 659. Durante o primeiro filtro (leitura de títulos e resumos), excluímos 609 estudos que não atendiam aos critérios de inclusão e aprovamos 50 publicações para o segundo filtro. No segundo filtro, realizamos uma leitura minuciosa dos artigos selecionados no primeiro filtro e, de acordo com os critérios de inclusão e exclusão, incluímos ou excluímos os artigos. Ao final do processo, aceitamos e extraímos 17 publicações. Todos os artigos podem ser consultados na Tabela 2.

Realizamos a avaliação da qualidade com base nos estudos selecionados por meio de critérios de seleção (inclusão e exclusão). A Tabela 3 apresenta a lista dos 17 estudos selecionados. Além disso, apresentamos os resultados do índice de qualidade, onde cada

Figura 2 – Resultados dos filtros do mapeamento sistemático



Fonte: De autoria própria.

Tabela 2 – Estudos Selecionados.

ID	Título do Trabalho	Referência
S01	A cognitive assistant for learning java featuring social dialogue	Coronado et al. (2018)
S02	A Pilot Study Integrating an AI-Driven Chatbot in an Introductory Programming Course	Verleger and Pembridge (2018)
S03	Coached program planning: Dialogue-based support for novice program design	Lane and VanLehn (2003)
S04	Design Intelligent Educational Chatbot for Information Retrieval based on Integrated Knowledge Bases	Nguyen *et al.* (2022)
S05	Development of online learning media based on Telegram Chatbot (Case studies: Programming courses)	Ardimansyah and Widianto (2021)
S06	FritzBot: A Data-Driven Conversational Agent for Physical-Computing System Design	Chen *et al.* (2021)
S07	Integrating A Dialogue Tree Based Turkish Chatbot into an Open Source Python Coding Editor	Bilgin and Yavuz (2022)
S08	Lecturer's Apprentice: A Chatbot for Assisting Novice Programmers	Ismail and Ade-Ibijola (2019)
S09	Pyo, a Chatbot Assistant for Introductory Programming Students	Carreira *et al.* (2022)
S10	Python-bot: A chatbot for teaching python programming	Okonkwo and Ade-Ibijola (2021)
S11	Revision-Bot: A Chatbot for Studying Past Questions in Introductory Programming	Okonkwo and Ade-Ibijola (2022)
S12	Sara, the Lecturer: Improving Learning in Online Education with a Scaffolding-Based Conversational Agent	Winkler *et al.* (2020)
S13	Say hello to 'Coding Tutor'! Design and evaluation of a chatbot-based learning system supporting students to learn to program	Hobert (2019)
S14	The impact of chatbots using concept maps on correction outcomes–a case study of programming courses	Kuo and Chen(2022)
S15	TicTad: A chatterbot for learning visual C# programming based on expert system	Kasinathan *et al.* (2018)
S16	Using Learning Analytics to Explore Responses from Student Conversations with Chatbot for Education	Wan Hamzah *et al.* (2021)
S17	When to intervene: Toward a Markov Decision Process dialogue policy for computer science tutoring	Mitchell *et al.* (2013)

Fonte: De autoria própria.

estudo pode ser identificado por meio da coluna ID. As pontuações baseadas nos CQA são mostradas na terceira (QA1), quarta (QA2), quinta (QA3) e sexta (QA4) colunas. A pontuação final é apresentada na sétima coluna, denominada SC (Score). Destacamos que três pesquisadores avaliaram individualmente cada um dos 17 estudos de acordo com os

quatro CAQs apresentados na Subseção 3.2.7.

Tabela 3 – Estudos de resultados de controle de qualidade.

ID	Referências	Avaliação de Qualidade				
		QA1	QA2	QA3	QA4	SC
S01	Coronado et al. (2018)	S	S	S	S	**4.0**
S02	Verleger and Pembridge (2018)	S	S	S	S	**4.0**
S03	Lane and VanLehn (2003)	S	S	S	S	**4.0**
S04	Nguyen et al. (2022)	P	S	S	N	**2.5**
S05	Ardimansyah and Widianto (2021)	P	S	N	P	**2.0**
S06	Chen, Xu and Zhu (2021)	S	S	S	N	**3.0**
S07	Bilgin and Yavuz (2022)	S	S	N	N	**2.0**
S08	Ismail and Ade-Ibijola (2019)	S	S	S	S	**4.0**
S09	Carreira et al. (2022)	S	S	S	N	**3.0**
S10	Okonkwo and Ade-Ibijola (2020)	S	S	S	S	**4.0**
S11	Okonkwo and Ade-Ibijola (2022)	S	S	S	N	**3.0**
S12	Winkler et al. (2020)	S	S	S	S	**4.0**
S13	Hobert (2019)	S	S	S	S	**4.0**
S14	Kuo and Chen (2023)	S	S	S	N	**3.0**
S15	Kasinathan et al. (2018)	S	S	S	N	**3.0**
S16	Hamzah et al. (2021)	S	S	S	P	**3.5**
S17	Mitchell, Boyer and Lester (2013)	N	S	S	N	**2.0**

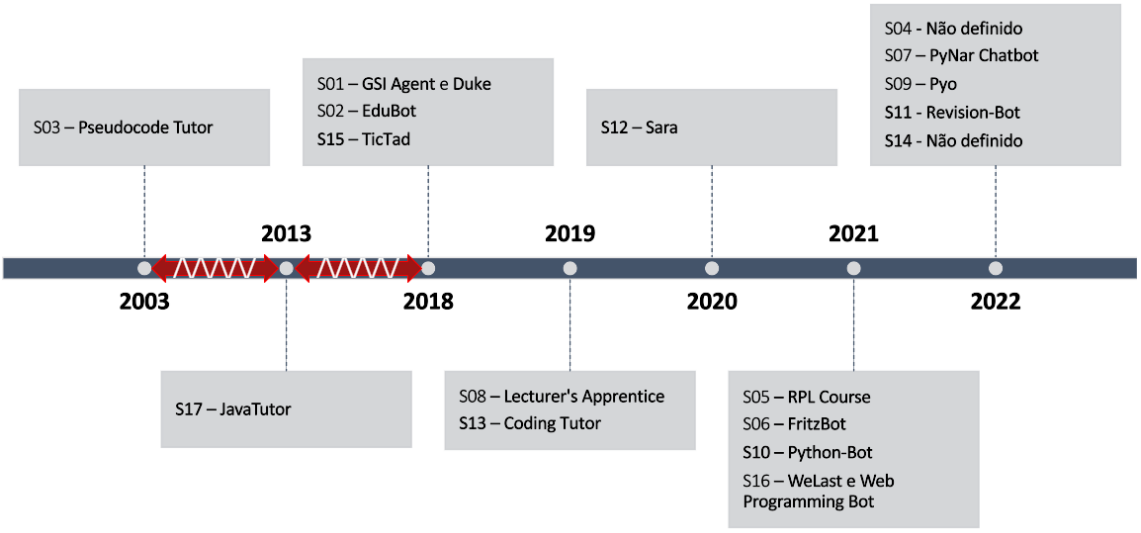
Fonte: De autoria própria.

3.3.2 Tendências de Publicação

Observamos que os estudos selecionados foram publicados entre 2003 e 2022. Em uma perspectiva temporal (Figura 3), notamos que, a partir de 2003, houve apenas uma publicação relevante para o nosso MSL, seguida por uma lacuna até 2013. Durante esse período, surgiu um outro estudo sobre o ensino de programação com *chatbots*. Após isso, identificamos mais um intervalo de cinco anos sem novas publicações alinhadas à nossa pesquisa. O crescimento em 2018 foi marcado por três publicações relevantes, seguido por uma queda em 2020 e um novo aumento em 2021 e 2022, com quatro e cinco publicações, respectivamente. Vale ressaltar que em 2017 houve a proposta de arquitetura de *transformes*, que revolucionou o processamento de linguagem natural, por meio do recebimento de uma frase ou texto que é quebrada e correlacionada em partes para construir a informação (BRAN; SCHWALLER, 2024), sendo muito útil para desenvolvimento de *chatbots* que trabalham com NLP. Além disso, em 2019 houve uma necessidade urgente de reestruturação do ensino e da aprendizagem, possivelmente ligada à pandemia da COVID-19. Isso resultou no desenvolvimento de novas ferramentas e abordagens,

entre as quais se destacam os *chatbots* ou assistentes virtuais, que se tornaram cada vez mais populares devido à sua acessibilidade a partir de locais remotos pela Internet. Diante disso, acreditamos que a comunidade tem explorado o campo dos *chatbots* aplicados ao ensino de programação.

Figura 3 – Tendências de Publicação.



Fonte: De autoria própria.

Classificamos e analisamos os estudos para avaliar sua distribuição por (i) tipo de publicação (ou seja, periódico, conferência ou e *workshop*) e (ii) locais de publicação específicos. Tabela 4) mostra a distribuição de acordo com o tipo de publicação. Os tipos de publicação mais frequentes são artigos de conferência e periódicos, cada um com oito estudos, seguidos por artigos de *workshops*, que apresentam apenas um estudo. Apesar de haver um estudo publicado em 2003, a quantidade de artigos em conferências e periódicos sugere a relevância contínua desse tópico.

Nas subseções a seguir apresentamos os resultados para cada subquestão.

3.3.3 SQ01 - Quais os *chatbots* existentes na literatura que apoiam na aprendizagem de programação?

Para responder à principal questão de pesquisa de nosso trabalho, analisamos e classificamos 17 estudos que evidenciam o uso de *chatbots* desenvolvidos para a aprendizagem de programação. A partir desses estudos, identificamos e caracterizamos 20

Tabela 4 – Locais de Publicação

Local de Publicação	Tipo	ID
International Journal of Human-Computer Studies (IJHCS)	Journal	**S1, S6**
International Journal of Computer Science (IJCS)	Journal	**S4,11**
Frontiers in Education Conference (FIE)	Conference	**S2**
Annual Technical Symposium on Computer Science Education (SIGCSE)	Conference	**S3**
Seminar on Advances in Mathematics, Science, and Engineering for Elementary Schools (SAMSES)	Workshop	**S5**
International Informatics and Software Engineering Conference (IISEC)	Conference	**S7**
International Multidisciplinary Information Technology and Engineering Conference (IMITEC)	Conference	**S8**
International Symposium on Computers in Education (SIIE)	Conference	**S9**
Engineering Letters (EL)	Journal	**S10**
Conference on Human Factors in Computing Systems (CHI)	Conference	**S12**
International Conference on Information Systems (ICIS)	Conference	**S13**
Education and Information Technologies (EAIT)	Journal	**S14**
Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)	Journal	**S15**
International Journal of Engineering Pedagogy (iJEP)	Journal	**S16**
International Conference on Artificial Intelligence in Education (AIEDCS)	Conference	**S17**

Fonte: De autoria própria.

chatbots.

Em uma perspectiva cronológica, o estudo **S3**, realizado em 2003, apresenta o Pseudocode Tutor, um *chatbot* dedicado ao suporte inicial à aprendizagem de programação por meio de pseudocódigo. Em seguida, o estudo **S17** demonstra o *chatbot* JavaTutor, que oferece uma tutoria em diálogo para o ensino de programação, com trocas de turnos irrestritas aplicadas na linguagem Java.

Em 2018, observou-se um aumento significativo na produção desses sistemas de conversação para o ensino de programação. O estudo **S1** apresentou dois *chatbots*: o GSI Agent, focado em auxiliar alunos na busca por informações do grupo de pesquisa, e o Duke, projetado para apoiar a aprendizagem de programação com a linguagem Java. O estudo **S2** introduz o agente Edubot, que fornece suporte à resolução de problemas utilizando a linguagem MATLAB. Finalizando os *chatbots* identificados no ano de 2018, temos o TicTad **(S15)**, que oferece um agente de conversação, em modo texto e voz, para apoiar o aprendizado baseado em cursos introdutórios de programação na linguagem C#.

Em 2019, surgiu o *chatbot* Lecturer's Apprentice **(S8)**, que se destaca por oferecer suporte à aprendizagem de programação introdutória baseada em pseudocódigo, além de auxílio em problemas emocionais que os alunos possam enfrentar, estejam eles relacionados ou não à fase de aprendizagem. O segundo *chatbot* identificado nesse ano em nosso estudo é o Coding Tutor **(S13)**, que fornece um tutor capaz de oferecer dicas e conteúdos adicionais para tarefas realizadas em Java. O Coding Tutor também é capaz de responder a questões abertas e analisar os códigos enviados pelos alunos, fornecendo *feedback*.

No ano 2020, destaca-se o Sara **(S12)**, um *chatbot* cujo objetivo é oferecer suporte

aos alunos durante as aulas *online*, proporcionando ajuda por voz e texto quando necessário. Sara é acionado ao pausar o vídeo de aprendizagem, fornecendo informações sobre os conceitos introdutórios de programação em linguagem Python.

Alguns anos à frente, em 2021, observamos um número significativo de cinco agentes conversacionais em nosso MSL. O estudo **S5** apresenta o RPL Course, voltado a auxiliar alunos na aprendizagem de programação básica, embora não especifique uma linguagem de programação. Em seguida, o estudo **S6** introduz o *chatbot* FritzBot, que tem por objetivo apoiar a modelagem e codificação de sistemas de computação física na linguagem Arduino. O Python-Bot, mencionado no estudo **S10**, fornece suporte à aprendizagem de programação em Python, oferecendo apontamentos e a possibilidade de agendar reuniões com tutores ou professores. O estudo **S16** descreve duas versões do mesmo *chatbot*: o WeLast (plataforma Android) e o Web Programming (bot via Telegram), ambos com o propósito de apoiar a aprendizagem em três componentes de programação web: *Hypertext Preprocessor* (PHP), *Database and Structured Query Language* (SQL) e *Hypertext Markup Language* (HTML).

No último ano retornado pelo MSL, em nossa análise, 2022, o estudo **S7** relata o PyNar, um *chatbot* integrado ao editor de códigos Pynar, projetado para apoiar alunos que desejam aprender Python de forma autônoma. Apresentado pelo estudo **S9** como uma ferramenta de auxílio para estudantes interessados em aprender a programar em Python, o PyNar abrange temas fundamentais de programação. Outro *chatbot* relevante é o Revision-Bot **(S11)**, desenvolvido para ajudar alunos na prática de questões de exames passados em um curso de programação em Python, com o objetivo de aprimorar o desempenho dos estudantes.

O estudo **S4** descreve um *chatbot* que oferece suporte à aprendizagem de conceitos de programação em linguagem C++ e princípios de programação orientada a objetos. Em seguida, o estudo **S14** explora um *chatbot* voltado ao apoio na aprendizagem de estruturas de dados, sem foco específico em uma linguagem de programação. Os estudos **S4** e **S14** não especificam o nome dos *chatbots* descritos.

A partir dos estudos analisados em nosso MSL, constatamos a presença de diversos *chatbots* aplicados à aprendizagem de programação, cada um com suas particularidades e abordagens específicas.

3.3.4 SQ02 - Qual o contexto/conteúdo ensinado por meio do chatbot?

Para responder à questão de pesquisa, nosso estudo identificou uma categorização em sete grupos para organizar os 19 *chatbots* descritos nos estudos analisados. Esses grupos foram definidos como:

- **Ensino de programação apoiado por linguagem de programação**: onde o *chatbot* apoiava a aprendizagem de conceitos de programação usando uma linguagem de programação;
- **Conceitos de Programação**: voltado a ensinar conceitos de programação, sem abordar diretamente uma linguagem de programação;
- **Paradigma de Orientação a Objetos**: destinado a apoiar a aprendizagem do paradigma;
- **Estrutura de Dados**: difundindo conceitos, teorias e forma de funcionamento de algoritmos que contemplam estruturas de dados;
- **Programação Web**: cujo *chatbot* apoiou a aprendizagem do desenvolvimento de aplicações para web;
- **Computação física** contemplando *chatbot* que auxilia na escrita de programas para prototipação de hardware, mais especificamente com Arduino;
- **Outros**: categoria que contempla outros assuntos encontrados nos *chatbots* retornados pelo MSL, no caso, Informações específicas sobre um grupo de pesquisa e Apoio Emocional.

A maioria, correspondendo a 63,16% dos *chatbots* identificados nos estudos, aborda o contexto ou conteúdo de apoio à aprendizagem de programação por meio da adoção de uma **Linguagem de Programação**, ou seja, ensina conceitos, como por exemplo, repetições adotando uma linguagem de programação, que são eles, **(S1, S2, S3, S4, S7, S8, S9, S10, S11, S12, S13, S15, S17)**. O *chatbot* apresentado no estudo **S5** foi classificado na categoria de **Conceitos de Programação**, com foco no ensino de conceitos iniciais de programação, como por exemplo definir codificação, isso usando apenas do idioma, no caso, língua da indonésia, sem apoio de linguagem de programação ou mesmo pseudocódigo correspondendo a 5,26%.

Além disso, observou-se que cinco *chatbots* foram aplicados em áreas que normalmente são ensinadas após o discente já possuir algum conhecimento em programação. Esses grupos incluem: **Paradigma de Orientação a Objetos (S4)**, com 5,26%,, onde são trabalhados os conceitos e codificação aplicados em temas, como classes, objetos, herança, etc, **Estruturas de Dados (S14)** com 5,26%, contemplando estruturas, como listas, pilhas, filas, **Programação Web (S16)** com 10,53%, este contendo dois *chatbots* que ensinam o usuário o desenvolvimento de sistemas para web e por fim o *chatbot* apresentado na categoria **Computação Física (S6)**, com com 5,26%, ensinando a construir protótipos de hardware e lógica básica para programação com Arduino.

Também foi identificado um percentual de 10,53% de *chatbots* que abordam conteúdos alheios à programação, como **Informações sobre o grupo de pesquisa (S1)** e **Apoio Emocional (S8)**.

A tabela 5 demonstra a separação detalhada conforme as linguagens abordadas por cada *chatbot* e seu estudo referente. Existe a replicação de *chatbot* em outra categoria, quando o mesmo se enquadra em mais de uma categoria, como por exemplo **S4**, ou o estudo contemplou mais de um *chatbot*, onde cada qual se encontra em uma categoria, como por exemplo, **S16**.

É importante destacar que o estudo **S4** abordou os temas "Introdução à Programação" e "Programação Orientada a Objetos" (POO), classificado-o em duas categorias

3.3.5 SQ03 - Qual linguagem de programação foi empregada pelo *chatbot* para apoiar na aprendizagem?

O levantamento revelou um conjunto de linguagens de programação abordadas pelos *chatbots*. Python se destaca com o maior número de *chatbots* relacionados, totalizando cinco, indicando uma ascensão significativa dessa linguagem no contexto educacional. Outras linguagens de programação incluem a linguagem de programação MATLAB (1), Java (3), Pseudocódigo (2), C++ (1), C# (1), Arduino *Language* (1) e uma categoria em que a linguagem utilizada no *chatbot* não foi especificada (1). Ao examinar as linguagens identificadas em nosso MSL, observamos uma crescente oferta de *chatbots* voltados ao apoio na aprendizagem de programação em linguagens específicas. Além disso, encontramos propostas direcionadas ao uso de pseudocódigo e a conteúdos textuais com definições de conceitos, conforme apresentado na Tabela 5. Notamos também a presença de *chatbots* voltados para assuntos avançados, como orientação a objetos e estruturas de

dados, frequentemente abordados em cursos mais avançados de programação.

Tabela 5 – Conteúdos Abordados

Conteúdo abordado pelos *chatbots*	**Linguagem de Programação**	***Chatbot***
Ensino de programação apoiado por linguagem de programação	C#	TicTad (**S15**)
	C++	não definido(**S4**)
	Java	Duke (**S1**)
		Coding Tutor (**S13**)
		JavaTutor (**S17**)
	MATLAB	LanguageEduBot (**S2**)
	Pseudocódigo	Pseudocode Tutor(**S3**)
		Lecturer's Apprentice(**S8**)
	Python	PyNar(**S7**)
		Pyo(**S9**)
		Python-Bot(**S10**)
		Revision-Bot(**S11**)
		Sara(**S12**)
Conceitos de Programação	Teoria e Definições (sem adoção de linguagem de programação)	RPL Course(**S5**)
Paradigma de Orientação a Objetos	C++	não definido(**S4**)
Estrutura de Dados	Ensino de algoritmos que trabalham com estrutura de dados (sem adoção de linguagem de programação)	não definido (**S14**)
Programação Web	PHP, SQL e HTML	WeLast(**S16**)
		Web Programming Bot(**S16**)
Computação Física	Arduino Language	FritzBot(**S6**)
Outros	Informações página grupo de pesquisa	GSI Agent(**S1**)
	Emocional	Lecturer's Apprentice(**S8**)

Fonte: De autoria própria.

3.3.6 SQ04 - Quais as ferramentas (linguagens e *frameworks*) utilizadas para desenvolver esses *chatbots*?

O MSL revelou uma diversidade significativa de tecnologias utilizadas na produção dos *chatbots*, com algumas notáveis exceções: os artigos **S3**, **S14** e **S17** não mencionam as tecnologias empregadas na construção dos *chatbots* apresentados.

No estudo **S1**, a construção do *chatbot* envolveu a utilização da ferramenta de raspagem Scrapy para coletar informações do site Java Vademecum, desenvolvido pelo professor José A. Mañas. Além disso, o ChatScript foi utilizado na construção do *chatbot*, em conjunto com modelos de ontologia Dublin Core e SKOS.

As tecnologias aplicadas no *back-end* do EduBot **(S2)** incluem o IMS-Global LTI Integration Standard para interação com o Canvas LMS e o Microsoft QnA Maker para o processamento de IA. O *front-end* utiliza uma interface em PHP para gerenciar o fluxo de trabalho na ferramenta.

O estudo **S4** empregou o PyVi, uma biblioteca para processamento de texto em vietnamita, para realizar a tokenização; o Word2VecVN, para classificar os tokens em

palavras do idioma vietnamita; e o Keras, um *framework* de *deep learning*, que foi utilizado para construir o modelo de aprendizado profundo do *chatbot*. O Rasa NLU foi aplicado para a classificação de intenções, incluindo o modelo vietnamita. Além disso, o FPT.AI Conversation, uma plataforma para criação e desenvolvimento de *chatbots* com suporte a múltiplas línguas, também foi utilizado. Adicionalmente, foi desenvolvida uma arquitetura neural baseada em Bidirectional LSTM para montar sequências relacionadas às entradas fornecidas.

O *chatbot* RPL Course, relatado no estudo **S5**, utilizou a plataforma Telegram e a linguagem de programação JavaScript. Ele foi desenvolvido com dois designs de sistema: um para o *bot* e outro para o banco de dados, implementado com *Apps Scripts*.

Para o *chatbot* FritzBot **(S6)**, destinado à computação física, foi empregada a estrutura de uma rede neural BiLSTM-CRF (*bi-directional Long Short-term Memory Network e Conditional Random Field*) como plug-in para a aplicação web Fritzing. A interface *front-end* foi desenvolvida em C++ com o *framework* Qt, comunicando-se com o sistema *back-end* em Python via protocolo TCP/IP.

No estudo **S7**, a construção do *chatbot* foi realizada utilizando a linguagem de programação Python, as bibliotecas Turkish NLP para o reconhecimento de sentenças e o Qt GUI para a construção da interface gráfica.

No estudo **S8**, o *chatbot* Lecturer's Apprentice utiliza a ferramenta IBM Watson, um serviço baseado em nuvem para responder a questões em linguagem natural básica. Além disso, o Android Studio, em conjunto com a linguagem de programação Java, foi empregado para a criação das classes manipuladoras do *chatbot*, enquanto a comunicação com as interfaces foi realizada por meio de XML.

O *chatbot* Pyo, do estudo **S9**, foi codificado em Python e desenvolvido utilizando o *framework* Rasa, que usa *machine learning* para facilitar a construção de *chatbots* complexos e sua integração com websites.

Os *chatbots* Python-bot (**S10**) e Revision-bot (**S11**) utilizaram a plataforma de desenvolvimento de *chatbots* SnatchBot como unidade de interface, empregada para processar as mensagens dos usuários e acessar um banco de mensagens com respostas pré-definidas. É importante destacar que o SnatchBot oferece suporte ao processamento de linguagem natural.

O *chatbot* Sara, descrito no estudo **S12**, foi desenvolvido utilizando a linguagem JavaScript e a ferramenta HTML5 Web Speech API, que incorpora dados de voz em aplicativos web. Adicionalmente, o *framework* NLP.js foi empregado para transcrever as

respostas dos alunos em texto e aplicar modelos de processamento de linguagem natural (NLP) para classificação de respostas.

As tecnologias utilizadas no *chatbot* Coding Tutor (**S13**) foram baseadas em uma API de serviços web RESTful, e a lógica de processamento foi implementada em JavaScript, utilizando a biblioteca de processamento de linguagem natural NLP.js. A interface foi desenvolvida com HTML, CSS3 e JavaScript, empregando a biblioteca Bootstrap 4 para garantir componentes responsivos.

No estudo **S15**, o *chatbot* TicTad não detalha as tecnologias utilizadas, mas informa o uso da linguagem de programação C# e da ferramenta de construção de bots Verbot 5, que combina processamento de linguagem natural com inteligência artificial.

Por fim, o estudo **S16**, os *chatbots* WeLast (para Android) e Web Programming Bot (no Telegram) foram desenvolvidos com o uso de Android Studio, Dart, Flutter e Dialogflow. O Android Studio serviu como ambiente de desenvolvimento integrado para aplicativos Android, enquanto Dart e Flutter foram empregados na construção das interfaces de usuário e lógica do aplicativo. O Dialogflow, uma plataforma de NLP da Google, foi utilizado para o processamento de linguagem natural dos *chatbots*.

3.3.7 SQ05 - Como os *chatbots* educacionais interagem com os estudantes?

De acordo com Hien et al. (2018) os*chatbots* podem ser classificados em três categorias principais de acordo com sua forma de interação: *Pattern-based model*, onde o *chatbot* combinam mensagens de entrada com o padrão pergunta-resposta para devolver uma resposta, o *Retrieval-based model* consulta e analisa os recursos disponíveis usando APIs para encontrar a resposta adequada, e por fim, *Generative model* que é o mais inteligente entre os três modelos, pois gera respostas com base nas entradas atuais e anteriores do usuário.

A forma de interação baseada em *Pattern-based model* enquadra os *chatbots*relatados nos estudos **S1**, **S3**, **S5**, **S14**, **S15**, **S16**, enquanto *Retrieval-based model* trouxe os *chatbots* dos estudos **S2**, **S8**, **S10**, **S11**, **S13**, **S17**, e do tipo *Generative model* os estudos **S4**, **S6**, **S7**, **S9**, **S12**.

3.3.8 SQ06 - Qual o tipo de licença do *chatbot*?

Em nosso MSL, a descrição sobre a forma de distribuição ou licença dos *chatbots*, conforme indicado pelos estudos, demonstrou que dos 17 estudos identificados, apenas **S11 e S13** informaram ser de domínio gratuito, enquanto os demais, devido à ausência dessa informação, foram classificados como "não relatado". A falta de clareza nesse aspecto destaca a necessidade de maior transparência por parte dos autores ao compartilhar detalhes sobre a distribuição e o licenciamento dos *chatbots* desenvolvidos em seus estudos.

3.3.9 SQ07 - O artigo apresenta uma avaliação empírica? Se sim, que tipo de estudo empírico foi realizado?

Em nosso conjunto de resultados, foi possível verificar que 15 dos 17 estudos realizaram algum tipo de experimento, indicando uma porcentagem de 88,24%, enquanto os dois restantes (**S5** e **S7**) não apresentam nenhuma evidência empírica, totalizando 11,76%. Dentre as evidências, podemos relatar Experimento Controlado, Estudo Experimental, Survey, Estudo de Caso e Experimento de Pesquisa de Campo. A relação entre os experimentos realizados e os *chatbots* pode ser observada na Tabela 6. É importante ressaltar que foi encontrado um estudo em que não foi realizado nenhum tipo de experimento (**S7**) e um em que não há muitas definições e formalidades sobre como foi conduzido o experimento (**S5**).

Tabela 6 – Experimentos por Estudo

Tipo de Experimento	**ID Estudo**
Experimento Controlado	S1, S3, S4, S6, S12, S14
Estudo Piloto	S2
Não definido	S5
Não realizado	S7
Survey	S8, S10, S11
Estudo experimental (Pesquisa Exploratória)	S9
Estudo de caso	S13, S15, S16, S17

Fonte: De autoria própria.

Como representante dos estudos que possuem evidência empírica no formato de experimento controlado, podemos destacar o estudo **S14**. Nesse estudo, a aplicação do

experimento foi conduzida dividindo os participantes em um grupo experimental e um grupo controle, cada um com 30 alunos. As etapas envolveram assistir a um vídeo tutorial, seguido por um questionário e uma etapa de correção em caso de não conseguir responder à pergunta do questionário. A diferença entre o grupo experimental e o controle estava na forma das correções. No grupo experimental, o *chatbot* explicou passo a passo os conceitos das unidades, enquanto no grupo de controle, as correções foram realizadas com mapas conceituais e descrições de texto. Todas as questões foram elaboradas utilizando uma escala Likert de cinco pontos para avaliação, enquanto questões sobre carga cognitiva foram avaliadas com uma escala Likert de sete pontos. Também foi conduzida uma entrevista presencial com oito alunos do grupo experimental, com foco no método de correção, no sistema de aprendizagem e em outros aspectos relevantes.

Na categoria de Estudo Experimental (Pesquisa Exploratória), sendo o único representante, **S9** apresentou o objetivo de identificar as funcionalidades que um *chatbot* para apoio à aprendizagem de programação deve possuir, bem como de avaliar a interação dos usuários com esse *chatbot*. O experimento foi conduzido com alunos que tinham acesso de alta disponibilidade a materiais de vídeo gravados sobre o conteúdo de programação. A etapa prática de programação foi realizada em uma plataforma que incorporou o *chatbot* Pyo. Essa plataforma também disponibilizou materiais instrucionais, exercícios de programação e um editor de código.

O estudo **S8** conduziu um *survey* online para avaliar a percepção dos alunos de uma universidade, matriculados no curso de Sistemas de Informação, sobre a programação introdutória (problemas), a possível utilidade do *chatbot* e para qual finalidade ele seria mais adequado.

A aplicação de **S2** foi realizada por meio de um estudo piloto, onde os participantes eram alunos do curso de introdução à computação para engenheiros, com foco na resolução de problemas por meio de programação em MATLAB. A análise foi realizada verificando a interação dos usuários com o EduBot ao longo do curso. Os pesquisadores apresentaram o *chatbot* no início das aulas e periodicamente relembravam os alunos sobre como utilizá-lo como uma opção para obter assistência com o curso.

3.3.10 SQ08 - Qual período de aplicação do uso do *chatbot*?

Os períodos de avaliação, ou utilização dos *chatbots* em experimentos, identificados em nosso MSL indicam uma grande variedade, como exemplificado no estudo **S9**, que

teve uma duração de dois meses, totalizando 40 horas. Este estudo focou em participantes com experiência mínima em programação.

No experimento conduzido em **S14**, os participantes eram alunos matriculados em cursos de programação, e a aplicação do *chatbot* teve uma duração de 4 semanas. Durante esse período, o tempo gasto por unidade era de aproximadamente 50 minutos. O experimento consistiu em 7 vídeos tutoriais, divididos em 7 unidades, e os alunos completaram de 1 a 2 unidades por semana.

No estudo **S16**, a aplicação do *chatbot* ocorreu durante 90 dias e envolveu 47 estudantes de uma universidade pública. A variação nos períodos de avaliação pode estar relacionada à diversidade de participantes nos experimentos, influenciando diretamente na duração da aplicação do *chatbot*, especialmente em casos nos quais os participantes possuem menos experiência e exigem um tempo mais extenso de interação com o sistema.

3.3.11 SQ09 - Quais métricas foram empregadas para avaliar o *chatbot*?

Para responder a SQ09, conduzimos uma extração de métricas que estão sendo utilizadas nos estudos para validar os *chatbots* apresentados pelo conjunto resultado de nosso MSL. Com base nessa informação, agrupamos as métricas que possuíam características semelhantes para denominar categorias, como pode ser observada na Tabela 7.

Tabela 7 – Métricas identificadas.

Métricas	**ID**
Interação	S1, S3, S9, S2, S16
Usabilidade	S6, S10, S11, S13
Engajamento	S1, S2
Acurácia - Chatbot	S2, S4, S10
Desempenho Aluno	S3, S6, S17
Carga de Trabalho Percebida	S6
Percepção sobre Programação	S8, S10, S11, S14, S15
Satisfação	S9, S10, S11, S13, S15
Retenção Conhecimento	S12, S14
Capacidade de Transferência	S12
Motivação de Aprendizagem	S14

Fonte: De autoria própria.

A primeira categoria que apresentamos é denominada **Interação**, sendo responsável por mapear estudos que analisaram informações referentes à forma de interação dos estudantes com os *chatbots*. Entre eles, destaca-se o estudo **S1**, que utilizou *logs* para ex-

trair essa métrica. Por outro lado, o estudo **S9** mediu o número de interações por semana e a quantidade de alunos ativos, exemplificando diferentes abordagens na avaliação da interação.A seguir, temos a categoria **Usabilidade**, que avalia aspectos referentes à facilidade de uso do *chatbot* (**S10**, **S6** e **S13**) e quão amigável é a interface, conforme exposto no estudo **S11**.

A categoria **Engajamento** aborda a continuidade e a frequência com que o estudante interage com o *chatbot*, demonstrando seu engajamento com o software. O estudo **S1** analisou como o uso de diálogo social nos sistemas de Perguntas e Respostas pode aumentar a satisfação dos usuários, enquanto o estudo **S2** relatou as percepções dos estudantes sobre como acessavam e mantinham o uso do *chatbot*, observando que alguns abandonavam quando não encontravam uma resposta imediata, recorrendo a outras fontes.

No que diz respeito à **Acurácia**, os estudos **S2**, **S4** e **S10** visaram avaliar o desempenho do *chatbot* em fornecer respostas assertivas em relação à entrada do usuário.

Relativamente poucos experimentos relataram a análise de métricas referentes ao **Desempenho do Aluno**. Nessa categoria, os estudos **S3**, **S6** e **S17** mediram o desempenho apresentado pelos alunos. O estudo **S3** analisou atividades de edição e compilação, o **S6** mediu o tempo e a taxa de conclusão da tarefa, e o **S17** aplicou pré e pós-testes de conhecimento do conteúdo apresentado pelo *chatbot*.

Carga de Trabalho Percebida é uma categoria na qual os estudos **S6** e **S14** mensuram, respectivamente, a carga de trabalho e a demanda mental dos participantes do experimento.

Outro ponto analisado pelos estudos foi a **Percepção sobre Programação**, que indica o interesse em compreender o nível de conhecimento dos participantes do experimento e os principais problemas enfrentados (**S8, S9, S10, S11, S14, S15**).

Nos estudos **S9, S10, S11, S13 e S15**, foram coletados dados para avaliar a **satisfação** dos usuários em relação à utilização do *chatbot*.

Em particular, o estudo **S12** destacou duas métricas que se diferenciaram das demais: **Retenção de Conhecimento** e **Habilidade de Transferência**. Para mensurar a primeira métrica, utilizaram-se questões de múltipla escolha, enquanto, para a segunda, aplicaram-se questões abertas.

Por fim, indicamos a última categoria de métricas, na qual o estudo **S14** revelou, por meio de seus experimentos, a extração da métrica de motivação para a aprendizagem. Esse estudo observou como diferentes métodos de correção influenciaram a motivação

dos alunos na assimilação de conceitos de programação.

3.3.12 SQ10 - Qual a modalidade de ensino em que foi aplicado?

As modalidades de ensino em que os *chatbots* foram aplicados foram classificadas em três categorias: presencial, a distância e não definida. Os resultados encontrados no MSL indicam que, na categoria **presencial**, onde o experimento ocorreu no local das aulas, estão os estudos **S2, S3, S4, S6, S12, S13, S14, S16 e S17**. Em seguida, os estudos nos quais os experimentos foram realizados **a distância**, por métodos que disponibilizaram a ferramenta de forma *online*, são **S10 e S11**. Por fim, o restante dos estudos, **S1, S5, S7, S8, S9 e S15**, não apresenta a descrição da modalidade empregada e, portanto, foi classificado como **não definida**.

3.3.13 SQ11 - Qual o perfil do estudante durante a aplicação do *chatbot* na aprendizagem?

De acordo com nossa questão de pesquisa principal, foram selecionadas apenas publicações que envolveram o uso de *chatbots* para a aprendizagem de programação no ensino superior. Isso implica que os participantes dos experimentos estão diretamente correlacionados com algum tipo de graduação. Dentre eles, podemos destacar os participantes relatados no estudo **S1**, que eram estudantes universitários com idades entre 18 e 25 anos, sendo a maioria com menos de 22 anos. Todos os participantes possuíam experiência em programação Java e conhecimento do domínio do estudo.

No estudo **S2**, os estudantes cursavam a disciplina de Introdução à Computação para engenheiros, ministrada por dois instrutores em uma instituição exclusiva de STEM+Business. Encontramos também, no estudo **S6**, 24 participantes (todos estudantes de design de uma universidade local, sendo 12 homens e 12 mulheres, com idade média de 22,1 anos). Continuando, o estudo **S10** aplicou o experimento em alunos do primeiro ano do curso de Sistemas de Informação da Universidade de Johannesburg, na África do Sul, incluindo aproximadamente 160 estudantes de primeiro ano que estavam cursando a disciplina introdutória de programação em Python, bem como aqueles que já haviam concluído o curso anteriormente.

O estudo **S12** escolheu um total de 182 participantes para o experimento, sendo

estudantes de graduação e de pós-graduação, dos quais 74 eram mulheres e 108 homens, com idades entre 18 e 35 anos, recrutados em uma universidade europeia de negócios.

O levantamento de informações sobre o perfil dos estudantes permite observar que a aplicação de *chatbots* no apoio à aprendizagem de programação abrange outros cursos além dos específicos de programação, como cursos para engenheiros, para designers e, até mesmo, para outros públicos de graduação. Essa abordagem busca validar a eficácia da ferramenta com participantes sem contato direto com programação.

3.4 Considerações Finais sobre o capítulo

O MSL investigou como esses *chatbots* têm sido desenvolvidos e aplicados para apoiar o ensino e a aprendizagem de programação, trazendo informações relevantes sobre os sistemas criados até o momento.

Os resultados encontrados no MSL indicam que os *chatbots* tem sido desenvolvidos para apoiar o ensino/aprendizagem de programação, entretanto é possível perceber que não existe uma padronização tecnológica para ferramentas de construção desses *chatbots*. A maior parte dos *chatbots* foram desenvolvidos para apoiar linguagem Python, talvez por sua alta popularidade, mas ainda assim existem outras opções de *chatbots* que abordam outras linguagens de programação.

A maioria dos *chatbots* abordou o ensino de programação contemplando conceitos como variáveis, constantes e laços de repetição, elementos que se concentram nas disciplinas iniciais dos cursos de programação. Em menor quantidade, encontram-se estudos que exploram áreas conceituais classificadas como mais avançadas, uma vez que demandam conhecimento prévio em programação, como Estruturas de Dados, Paradigma de Programação Orientada a Objetos, Programação para Web e Computação Física. Esses *chatbots* tiveram como objetivo apoiar o aprendizado de programação, oferecendo recursos valiosos aos estudantes, o que confirma a versatilidade e o potencial do uso dessa ferramenta no auxílio à aprendizagem de programação.

Os *chatbots* que trabalhavam apenas com conceitos, ou uso de pseudocódigos, foram uma minoria, o que pode indicar uma tendência a adoção direta de uma linguagem de programação.

Muitos dos *chatbots* encontrados necessitam de mais experimentos ou uma melhor especificação, para que suas avaliações possam fornecer investigações profundas e trazer mais confiabilidade quanto sua colaboração no ensino/aprendizagem de progra-

mação. Talvez estudos comparativos entre essas ferramentas possam trazer informações valiosas para a área de pesquisa.

Outro aspecto identificado, que levantou questionamentos, foi a falta de informação sobre a licença da maioria dos *chatbots*, o que pode indicar restrições de uso. Isso impacta diretamente novas pesquisas, pois não se sabe até onde é possível utilizar, modificar ou mesmo produzir novos conteúdos com base nesses *chatbots*. Torna-se necessário consultar os pesquisadores responsáveis para compreender sob qual licença seus softwares estão disponibilizados.

Por fim, e muito provavelmente o aspecto mais relevante identificado por este MSL, destaca-se a ausência de um padrão para a construção dos exemplos de programação. Cada *chatbot* analisado apresenta seus exemplos ou respostas às entradas de forma distinta, sem qualquer informação sobre um estudo ou levantamento prévio que identificasse os aspectos necessários para a construção de exemplos capazes de apoiar o estudante em sua aprendizagem.

Os elementos identificados nesses *chatbots* analisados no MSL serviram como base para a construção do *template* de *worked examples* e da ferramenta educacional, o assistente virtual CoderBot. Além disso, as formas de avaliação apresentadas fundamentaram o estudo de validação da ferramenta.

4 TEMPLATE DE *WORKED EXAMPLES* PARA O ENSINO DE PROGRAMAÇÃO

Este capítulo apresenta um *template* projetado com o objetivo de auxiliar os docentes a padronizar o uso de WE no ensino de programação. Além disso, também descreve os resultados de um estudo exploratório, realizado com docentes que usaram o *template* e que forneceram *insights* sobre a viabilidade e a eficácia do uso do *template*.

4.1 Introdução

Apesar da popularidade dos WE em outras áreas de conhecimento, há uma escassez de estudos que abordam sua prática na área de Computação, especialmente no ensino de programação. *Worked Examples* podem ser facilmente adaptados para esse contexto, já que a programação pode ser dividida em linhas de código sucessivas, representando as etapas de processamento da solução, que culminam em um código mais complexo para resolver um problema.

Aprender a programar envolve adquirir uma estratégia para resolver problemas, e os exemplos trabalhados têm se mostrado úteis nesse processo (MERRIENBOER, 1990; HOSSEINI et al., 2020). No entanto, muitas vezes, os docentes limitam-se a apresentar exemplos com conceitos básicos e, gradualmente, abordam tópicos mais complexos. Sem o uso de esquemas ou estruturas adequadas que permitam aos estudantes explorar todo o seu potencial de aprendizagem, eles podem enfrentar dificuldades na compreensão de novos conteúdos apresentados em sala de aula.

Segundo Atkinson *et al.* (2000), o design e a estrutura dos WEs são importantes ao discutir a eficácia desses recursos em sala de aula. Logo, é necessário considerar o *design* e a estrutura dos materiais de aprendizagem, a fim de otimizar a compreensão e o aprendizado dos estudantes.

4.2 Metodologia para o desenvolvimento e avaliação do *template*

Para alcançar o objetivo da construção do *template*, fizemos uso da metodologia *Design Science Research* (DSR) (HEVNER, 2007). A DSR é uma metodologia que apoia o processo de criação, desenvolvimento e avaliação de artefatos que visam solucionar

problemas práticos de interesse da comunidade. A DSR é composta por três ciclos inter-relacionados: o Ciclo de Relevância, o Ciclo de Design e o Ciclo de Rigor.

O **Ciclo de Relevância** consiste na definição do problema a ser pesquisado. A motivação desta pesquisa está relacionada à percepção de que a aplicação de WE está limitada, resultando em uma aprendizagem ineficaz. Ao fornecer um *template* estruturado e padronizado, os docentes terão um guia claro sobre como utilizar WE em sala de aula. Isso ajudará os docentes a criar exemplos relevantes, abordando problemas específicos de programação, destacando as etapas de solução e fornecendo aos estudantes uma base sólida para a compreender conceitos e adquirir habilidades cognitivas.

O **Ciclo de *Design*** consiste no desenvolvimento e avaliação do artefato para o problema. Neste trabalho, o artefato é o *template* de *worked examples*. Para o desenvolvimento do *template*, realizamos uma pesquisa em busca de evidências na literatura de artigos publicados que já usaram WE, e tentamos identificar características comuns de como os WE podem ser aplicados. Como ponto de partida, decidimos utilizar o modelo proposto por Tonhão *et al.* (2021). A proposta dos autores apresenta os principais elementos de um exemplo trabalhado e informações consideradas importantes para o seu entendimento, além de informações de cunho pedagógico que são importantes para auxiliar os docentes na adoção desses exemplos. Além disso, encontramos outras evidências de trabalhos na literatura que incorporamos também na primeira versão do *template*. Vale ressaltar que não identificamos nenhum outro trabalho que estruture e forneça guidelines de como utilizar *worked examples* para o ensino de programação.

A Tabela 8 apresenta a versão inicial, bem como o mapeamento das informações coletadas de cada trabalho que identificamos. O *template* está dividido em três partes: "Dados Gerais", "Dados sobre o *Worked Example*" e "Aplicação do *Worked Example* (Corretos e Errôneos)".

Na **Parte 01 (Dados Gerais)** do *template*, é o local em que o docente fornecerá informações dos dados gerais sobre o curso e a origem do exemplo trabalhado. Nos Dados Gerais sobre o Curso, devem ser adicionadas informações como o nome da disciplina e o conteúdo que será ensinado, o tópico da disciplina, o subtópico da disciplina (conteúdos específicos, se houver) e, por fim, o conhecimento prévio necessário para os estudantes compreenderem o exemplo trabalhado. Nos Dados do Local do exemplo trabalhado, deve-se informar o local de onde exemplo foi retirado, por exemplo, de slides de aula, de repositórios ou de projetos abertos (GitHub).

Na **Parte 2 (Contexto do Exemplo Trabalhado)** devem ser fornecidas informa-

Tabela 8 – *Proposta de Template* para uso de WE para o ensino de programação.

Parte 01 - Dados Gerais	**Dados Gerais Sobre o Curso**	
	Título da disciplina	[Inserir título da disciplina] (TONHãO; COLANZI; STEINMACHER, 2021)
	Tópico da disciplina	[Inserir tópico da disciplina] (TONHãO; COLANZI; STEINMACHER, 2021)
	Subtópicos	[Inserir subtópicos da disciplina que serão abordados na atividade] (TONHãO; COLANZI; STEINMACHER, 2021)
	Conhecimento prévio	[Inserir conhecimentos prévios necessários para a compreensão da atividade] (TONHãO; COLANZI; STEINMACHER, 2021)
	Dados do Local do *Worked Examples*	
	Local onde foi retirado	[Inserir de onde o material para o *worked example* foi retirado] (TONHãO; COLANZI; STEINMACHER, 2021)
Parte 02 - Contexto	**Dados do *Worked Example***	
	Descrição do problema	[Inserir descrição da atividade a ser realizada, incluindo o problema a ser resolvido]
	Resultado	[inserir resultado para o problema abordado]
	Material complementar	[Inserir material de apoio para resolução do problema]
Parte 3.1 - *Worked Example* Correto	***Worked Example* Correto**	
	Reflexivo	[Inserir texto explicativo sobre o problema com worked example, o texto deve ajudar o aluno a refletir sobre o problema] (ABDUL-RAHMAN; du Boulay, 2014) (MCGINN; LANGE; BOOTH, 2015) (RODIAWATI; RETNOWATI, 2019) (HUANG et al., 2015)
	Níveis de dificuldade	[Inserir níveis de dificuldade da atividade reflexiva, como fácil, médio ou difícil] (HUANG et al., 2015)
	Etapas Corretas	[Inserir passo-a-passo para conclusão do exercício] (RODIAWATI; RETNOWATI, 2019) (RETNOWATI; MARISSA, 2018) (MCGINN; LANGE; BOOTH, 2015)
	Teste	[Inserir dados de testes para verificar se o resultado obtido é o mesmo que o esperado] (ABDUL-RAHMAN; du Boulay, 2014)
Parte 3.2 - *Worked Example* Errôneo	***Worked Example* Errôneo**	
	Reflexivo	[Inserir texto explicativo sobre o problema com worked example, o texto deve ajudar o aluno a refletir sobre o problema] (ABDUL-RAHMAN; du Boulay, 2014) (MCGINN; LANGE; BOOTH, 2015) (RODIAWATI; RETNOWATI, 2019) (HUANG et al., 2015)
	Níveis de dificuldade	[Inserir níveis de dificuldade da atividade reflexiva, como fácil, médio ou difícil] (HUANG et al., 2015)
	Etapas Errôneas	[Inserir passo-a-passo para conclusão do exercício] (CHEN; MITROVIC; MATHEWS, 2016) (HUANG et al., 2015)
	Você consegue identificar o erro? Indique a linha onde este ocorre	[Inserir uma explicação sobre porque o erro ocorre e em que parte do código/passo-a-passo está] (GARCES et al., 2023)
	Teste	[Inserir dados de testes para verificar se o resultado obtido é o mesmo que o esperado] (ABDUL-RAHMAN; du Boulay, 2014)

Fonte: De autoria própria.

ções que expliquem o contexto do exemplo para o discente, incluindo a descrição do problema, a demonstração de resultado com dados de entrada e saída esperados, e material complementar para auxiliar no aprendizado do estudante. É importante que o material complementar contenha recursos adicionais, como vídeos, sites ou documentação.

Em seguida, temos a **Parte 3 (*Worked Example* Correto)**, que demonstra a seção em que a solução do problema é apresentada por meio do WE ao estudante. Existem quatro itens que devem ser preenchidos. O primeiro é o "Reflexivo", no qual o docente deve fornecer um detalhamento sobre a lógica que o estudante deve seguir para compreender e resolver o problema. Isso permite que o estudante estimule a utilizar sua capacidade cognitiva e conecte o novo conhecimento com experiências anteriores de aprendizagem. Isso destaca a importância de ativar e integrar conhecimentos prévios para uma compreensão mais profunda e efetiva do WE (ABDUL-RAHMAN; du Boulay, 2014). A partir dessas informações, o estudante poderá criar etapas para solucionar o problema, formando assim

mapas mentais. O segundo item está relacionado ao nível de dificuldade do exemplo, no qual o docente deve classificar qual a dificuldade do exemplo. O terceiro item é responsável por identificar as Etapas Corretas da Solução. Neste item devem ser indicados os passos necessários para chegar a uma solução correta do problema, complementando o conteúdo conteúdo apresentado no item "Reflexivo". Por fim, o último item é responsável pelos "Testes", no qual o docente deve apresentar os possíveis dados de entradas para o problema, bem como as saídas esperadas. Dessa forma, o estudante pode receber um *feedback* sobre a sua solução.

A **Parte 4 (*Worked Example* Errôneo)** possui objetivos semelhantes aos apresentados na Parte 3. Contudo, há uma diferença, pois a Etapa 4 deve conter um ou mais passos incorretos. A quantidade de erros no código deve ser determinada pelo docente, podendo ser erros sintáticos ou semânticos. O estudante terá acesso aos itens "Reflexivos" e as "Etapas Errôneas", nos quais poderá realizar uma análise comparativa e identificar o erro. Em seguida (Identificação do Erro), o estudante deverá identificar, compreender e relatar os erros encontrados. Assim, o estudante estará aprendendo e expandindo seus conhecimentos, além de obter informações sobre como evitar os erros em atividades futuras (CHEN; MITROVIC; MATHEWS, 2016). Após o estudante identificar e relatar o erro, o docente deverá explicar detalhadamente a causa e, em seguida, apresentar a proposta de solução correta do WE. O último campo é o mesmo que se repete em relação à etapa 4, no qual são realizados os testes com as entradas e as suas possíveis saídas.

Por fim, o **Ciclo de Rigor** é a etapa na qual se obtém a contribuição esperada, com base nas teorias que direcionaram a pesquisa para alcançar o artefato proposto, como a Teoria da Carga Cognitiva (SWELLER; MERRIENBOER; PAAS, 1998), DSR (HEVNER, 2007), *Worked Examples* (GROSSE; RENKL, 2007; MCLAREN et al., 2016). A contribuição oferecida é denotada como a criação de um *template* estruturado e padronizado que visa auxiliar os docentes na produção de WE no ensino de disciplinas de programação.

4.3 Estudo Exploratório

Na metodologia DSR, é esperado que em cada ciclo de *Design* sejam realizadas avaliações no artefato proposto (HEVNER, 2007). Com base nessa premissa, esta seção apresenta o planejamento e a execução de um estudo exploratório conduzido para examinar a viabilidade de uso e a aceitação do *template* proposto, a partir da perspectiva de docentes.

4.3.1 Planejamento

Para facilitar a execução do estudo foram utilizadas as ferramentas disponibilizadas pelo *Google Workspace*. Os instrumentos elaborados compreenderam: (i) um termo de consentimento, que garantiu a confidencialidade dos dados fornecidos e o anonimato dos participantes; (ii) questionários de caracterização de perfil, para obter informações sobre os conhecimentos relacionados a exemplos e as características dos docentes; (iii) documentos contendo o roteiro do estudo e as instruções necessárias para a realização da avaliação; (iv) uma apresentação sobre *worked examples* e o *template* proposto.

4.3.2 Participantes

A análise realizada é de natureza exploratória e qualitativa, portanto, a representatividade da amostra é mais importante do que a quantidade. Dado esse contexto, quatro docentes do curso de Engenharia de Software, da Universidade Federal do Pampa (Unipampa), foram convidados para participar do estudo (D1, D2, D3 e D4). O convite foi realizado por conveniência. Todos os docentes tinham mais 10 de anos de experiência com o ensino em programação, sendo dois do gênero masculino e dois do gênero feminino. Três deles possuem doutorado e um possui mestrado, todos na área de Computação. Outro ponto a ressaltar é que os docentes atuam no ensino de disciplinas como Algoritmos e Programação, Estruturas de Dados, Programação Orientada a Objetos, entre outras. Nenhum dos docentes utilizou qualquer tipo de *template* para dar aula utilizando exemplos. Contudo, dois desses docentes (D1 e D2) comentaram que sempre utilizam exemplos em suas aulas, enquanto os outros dois (D3 e D4) utilizam com uma menor frequência.

4.3.3 Execução

A execução do estudo foi conduzida presencialmente com os docentes. Antes da avaliação, decidimos realizar um estudo piloto afim de verificar se o estudo alcançaria seus objetivos. Os resultados do piloto foram satisfatórios e não foi necessário aprimorar o roteiro do estudo.

Cada docente foi convidado por e-mail, que descrevia o objetivo do estudo e algumas orientações norteadoras. Caso aceitassem, era feito o agendamento de uma data para

a condução e, em seguida, era enviado um e-mail com o roteiro de preparação do estudo. Nesse documento, estavam disponíveis o formulário *online* do termo de consentimento e o formulário de caracterização, no qual os docentes deveriam responder a perguntas relacionadas à sua experiência de ensino.

A participação no estudo foi voluntária, e todos os docentes assinaram o termo de consentimento e aceitaram participar do estudo e fornecer os resultados para análise. No roteiro, os docentes foram orientados a assistir a uma breve apresentação sobre *worked examples* e, em seguida, sobre a proposta do *template*. Foi solicitado também aos docentes que utilizassem e criassem WE a partir do *template* disponibilizado antes da entrevista.

Na data agendada, foi conduzida a avaliação com os docentes, na qual foi solicitado que respondessem um questionário com o objetivo de coletar suas experiências e impressões sobre o *template*. O questionário foi elaborado com base nos indicadores TAM (VENKATESH; DAVIS, 2000), que são: **utilidade percebida**, define o grau em que o docente acredita que o uso do ***template*** pode melhorar seu desempenho no trabalho; **facilidade de uso percebida**, define o grau em que o docente acredita que a adoção do *template* pode ser feita sem muito esforço; e **intenção de uso percebida**: que define o grau em que o docente acredita que poderá utilizar o *template* no futuro. Cada item foi respondido em uma escala de cinco pontos, variando de "Discordo Fortemente" a "Concordo Fortemente", com opção neutra. Os itens do questionário podem ser vistos na Tabela 9. Após a a conclusão do questionário, realizamos uma entrevista de acompanhamento, com duração média de 20 minutos, na qual os docentes puderam explicar mais livremente suas percepções sobre o *template*.

Tabela 9 – Afirmativas do TAM.

Utilidade Percebida	
UP1	O template de worked examples melhora o meu desempenho contribuindo no planejamento das aulas de programação.
UP2	O template de worked examples melhora a minha produtividade no ensino de programação.
UP3	O template de worked examples facilita o trabalho dos docentes ao ensinar programação aos estudantes.
UP4	Eu acho o template de worked examples útil para auxiliar os docentes no ensino de programação em sala de aula.
UP5	Usando o template de worked examples para apoiar o ensino de programação, eu seria capaz de ensinar os conteúdos mais rapidamente.
Facilidade Percebida	
FUP1	O template de worked examples foi claro e fácil de entender para mim.
FUP2	Usar o template de worked examples não demandou muito esforço mental.
FUP3	Eu acho que o template de worked examples é fácil de usar.
FUP4	Eu acho fácil lembrar como executar as tarefas usando o template de worked examples.
Intenção de Uso	
IU1	Assumindo que eu tenha acesso ao template de worked examples, eu pretendo usá-lo futuramente para me apoiar no ensino de programação.
IU2	Dado que eu tenha acesso ao template de worked examples, eu prevejo que eu o usaria futuramente no ensino de programação.

Fonte: De autoria própria.

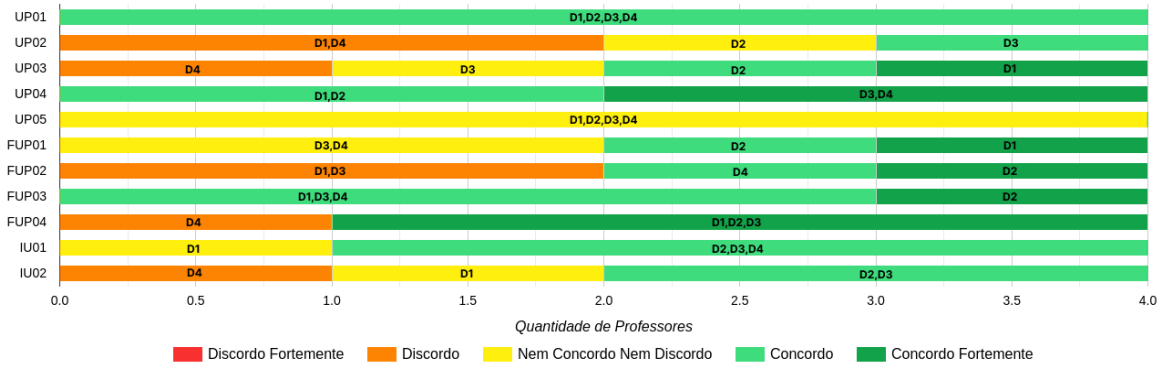
4.4 Resultados

Essa seção apresenta os resultados quantitativos e qualitativos obtidos durante a avaliação do *template* pelos participantes.

4.4.1 Percepções dos docentes sobre o *template*

A Figura 4 ilustra os resultados da análise quantitativa, obtidos através das respostas dos docentes que participaram avaliando o *template* proposto utilizando o método TAM.

Figura 4 – Avaliação do *template* pelos docentes



Fonte: De autoria própria.

Em relação à **Utilidade Percebida** do *template*, observamos que todos os docentes concordaram sobre a melhora no planejamento das aulas (UP1). No item UP05, todos os docentes permaneceram neutros, indicando que o uso do *template* pode ou não ter impacto na celeridade no processo de ensino do conteúdo. No item UP02, os docentes D1 e D4 não acreditam que o *template* pode causar uma melhoria da produtividade no ensino de programação. Com relação à **Facilidade de Uso Percebida**, observamos divergências de opiniões. No item FUP02, metade dos docentes (D1 e D3) discordaram de que há pouca demanda mental para usar o *template*. Por outro lado, no item FUP04, a maioria dos docentes (três de quatro) considerou fácil memorizar a organização do *template*. Essas divergências podem estar relacionadas à complexidade das disciplinas envolvidas, as quais podem demandar uma carga mental mais alta para uso do *template*.

Quanto à **Intenção de Uso Futuro Percebida**, notamos que houve uma tendência moderada de adesão ao *template* proposto. Em relação ao IU01, três docentes demonstra-

ram interesse em utilizar o artefato em suas práticas acadêmicas. Entretanto, em relação ao IU02, apenas dois docentes concordaram plenamente sobre a previsão de usar o *template*, caso este estivesse disponível. Essa diferença entre as declarações dos participantes e suas intenções reais pode ser atribuída a diversos fatores, como apercepção individual sobre a eficácia e utilidade do *template* dentro de suas próprias competências de ensino.

4.4.2 Resultados Qualitativos

A partir das entrevistas realizadas, realizamos uma análise qualitativa dos dados e identificamos pontos a partir de duas principais categorias.

A primeira categoria está relacionada às **instruções disponíveis**, em que os docentes comentaram sobre a FALTA DE CLAREZA E INSTRUÇÕES DO TEMPLATE. D1 apontou a dificuldade de visualizar a apresentação do código a linha a linha no *template*, mencionando que isso seria um problema caso os docentes desejasse fazer uso de fluxogramas: "*pressupõe que estou usando uma notação que vai de linha a linha(...) Se fosse um fluxograma, não tem linha, isso poderia ser um problema.*" D1 também sugeriu uma melhor estruturação dos testes, tornando-os mais objetivos e claros para os estudantes: *"no teste, eu os esperaria em uma perspectiva de Computação,(...) com uma entrada e uma saída esperada(...) acho que vai ficar mais objetivo."* Complementarmente, D3 destacou a falta de clareza sobre o conceito de teste adotado no *template*: *"pra mim, não ficou claro o que é o teste no template."* As observações dos docentes evidenciam necessidade de fornecer instruções detalhadas no *template*, explicando o conceito de caso de teste usado e orientando os docentes sobre como inserir os valores indicados para obter a saída esperada.

Notamos ainda que houve uma dificuldade com relação ao LAYOUT E A ORGANIZAÇÃO DOS ELEMENTOS DO TEMPLATE. O docente D4 acredita que é necessário ter uma melhor divisão dos elementos e sugere, *"talvez centralizar, ou destacar mais, ou até dividir."* Ele complementou dizendo que seria necessário realizar melhorias nos campos de preenchimento, tornando mais claro os pontos a serem preenchidos: "*mais a questão de layout que fiquei dúvida, preencho aqui ou aqui? O visual vai ajudar se ficar mais claro.*" Por outro lado, D3 acredita não ser necessário alterar a organização do *template*, pois acredita que " *a estrutura dele [template] está adequada, eu não tiraria nenhuma parte dele*."

Complementarmente, D2 expressou confusão ao não compreender se *"a proposta*

de solução incorreta é do mesmo algoritmo." Esse docente também apontou a redundância do item 'nível de dificuldade' que aparece em dois pontos do *template* e sugeriu que *"poderia estar apenas em cima, assim não repete embaixo."* D3 sugeriu que talvez *"não precisaria ter nível de dificuldade, pois se eu for aplicar em uma turma, não sei se isso é relevante para aparecer agora, e ainda aparece duas vezes."* Como pode ser observado, os docentes sugerem a remoção da duplicação do 'nível de dificuldade' e a centralização dele em um único lugar do *template*, deixando mais claro apenas em relação ao problema. Outro ponto levantado foi a apresentação ou não desse item considerando o contexto de uma turma em que o *template* possa ser aplicado.

Os docentes também forneceram algumas sugestões, que foram agrupadas na categoria **"sugestões de melhoria."** A primeira sugestão diz respeito à ADEQUAÇÃO DA NOMENCLATURA E TERMINOLOGIA usada no *template*. D1 sugeriu uma definição mais clara para a palavra 'erro', pois a definição dessa palavra pode variar dentro da Computação: *"tinha pensado na nomenclatura que está usada (erro), porque dependendo da área da Computação, a questão da nomenclatura de 'falha', 'erro' ou 'defeito' muda um pouco."*

D1 relatou que uma possível melhoria seria fazer CORRELAÇÃO DE DISCIPLINAS E PRÉ-REQUISITOS, uma vez que dependendo do curso de graduação, o nome da disciplina pode ser diferente: *"poderia ter sinônimos dessas disciplinas ou possíveis outros nomes pra elas. Aí você teria tanto conhecimento prévio, como coisas que eles deveriam saber pra fazer esse exemplo, e coisas que eles deveriam saber fazer depois de estudar esse exemplo."* Dessa forma, ao fornecer essa correlação, o *template* pode se adequar melhor aos diferentes contextos de ensino e garantir que os estudantes tenham a base necessária para aproveitar ao máximo o exemplo proposto.

Uma terceira sugestão está relacionada à CITAÇÃO E AUTORIA DOS WORKED EXAMPLES. D3 acredita ser importante mencionar a autoria: *"fiquei pensando se autoria não seria interessante, acho que uma autoria seria uma diferença aqui [no template]."* Tal ponto é interessante a ser discutido, até por questão de credibilidade, aumentando assim a confiabilidade do WE disponibilizado aos estudantes. Além disso, D1 sugeriu a adição de CAMPOS OPCIONAIS E INSTRUÇÕES GERAIS: *"faltou adicionar o que é opcional e o que não é, talvez, até instruções de como preencher os worked examples em uma visão geral. Os campos opcionais são marcados de algum jeito, ou, então, tá escrito quando é opcional(...), mas preferencialmente coloque os dois."* D1 ainda sugeriu a INCLUSÃO DE MÚLTIPLOS TESTES OU CASOS DE TESTE, ou seja, adicionar mais testes, ou um teste

que seria opcional para o caso de um erro.

Evidenciando a necessidade de execução dos códigos, D2 sugeriu a adoção de MÚLTIPLAS ABORDAGENS E PROPOSTAS DE SOLUÇÃO. O docente também sugeriu a inclusão de um campo mais amplo para adicionar o código, semelhante a um compilador online, o que permitiria realizar testes rápidos e diretos no código: *"talvez fosse interessante ter um espaço dedicado ao código, parecido com aqueles exemplos que, às vezes, apresentamos no Online GDB (compilador online)."* D3 endossa, indicando que seria útil uma ferramenta para auxiliar no processo de criação dos WE: *"seria útil, se houvesse uma ferramenta para me ajudar a preencher esse campo."* Outro ponto destacado por D4 é que os problemas computacionais podem ter diferentes formas de resolução, portanto, seria interessante possuir mais de um campo para adicionar uma solução diferente: *"existem problemas que podem ter mais de uma solução, talvez, colocar mais de um tipo de solução do problema(...) acho que deveria ter opção de criar duas propostas de solução, para o estudante ver a diferença na hora de resolver."* Com isso em mente, notamos a demanda por uma ferramenta que auxilie na criação de exemplos trabalhados, o que destaca a importância de recursos e suporte para os docentes na elaboração de materiais de ensino.

4.5 Considerações sobre o capítulo

Fornecer aos docentes uma forma de padronizar e estruturar *worked examples* pode ser essencial para uma melhora considerável nos resultados de aprendizado. Nesse sentido, apresentamos a proposta de um *template* para auxiliar os docentes no uso de WE no ensino de programação. A partir dos resultados do estudo exploratório, percebemos que os docentes tiveram uma receptividade positiva, destacando utilidade do *template* para melhorar o planejamento das aulas e a organização dos WE. Além disso, as discussões dos docentes revelaram *insights* sobre a aplicabilidade do *template* em diferentes contextos de ensino e as possibilidades de personalização e adaptação para atender às necessidades específicas de cada docente e turma. Entendemos que este é um estudo inicial e que ainda há espaço para aprimoramentos e investigações adicionais. Dado isso, os docentes também forneceram opiniões em relação ao *layout* e a forma como o *template* encontra-se estruturado e concordamos que há a necessidade de realização de melhorias, objetivando uma maior coerência e coesão do *template*.

O *template* elaborado, se torna parte fundamental na construção do CoderBot,

tendo em vista que o mesmo apresenta exemplos baseados em um estudo que passou por etapas de empirismo. Vale ressaltar que o *template* passou por adaptações para inclusão dos exemplos na ferramenta, e portanto, foram eliminadas etapas redundantes, e até mesmo informações que foram consideradas de mais valor ao docente, foram removidas na apresentação do exemplo ao estudante.

Foi mencionado também sobre a necessidade de uma solução computacional que simplifique e agilize o uso do *template* no contexto do ensino: aos docentes como forma de criação dos *worked examples* e aos estudantes como forma de apoio a aprendizagem.

5 PROPOSTA DO CODERBOT

Este capítulo apresenta o assistente virtual proposto que possui o intuito de apoiar a aprendizagem de programação em cursos de graduação.

5.1 Considerações Iniciais

Buscando compreender o cenário atual do uso de *chatbots* como ferramentas de apoio à aprendizagem de conteúdos de programação, foi conduzido MSL com o objetivo de identificar estudos que apresentassem evidências do uso desses assistentes virtuais. O MSL proporcionou *insights* sobre a utilização dos *chatbots* como ferramenta de apoio pedagógico, destacando que, embora a área esteja em constante exploração, ainda há amplo espaço para novas pesquisas.

Grande parte dos trabalhos encontrados concentra-se em aspectos relacionados ao suporte de disciplinas ou ao ensino de conteúdos específicos de programação. Contudo, observou-se uma carência de padrões estruturais nos exemplos apresentados aos estudantes. Diante dessa lacuna, e respaldados pela metodologia de aprendizagem baseada em exemplos, identificou-se a necessidade de elucidar os aspectos cruciais na construção de exemplos, resultando no desenvolvimento de um *template*, publicado no Simpósio Brasileiro de Informática na Educação (SBIE 2023) (MIRANDA et al., 2023), conforme detalhado no Capítulo 4. Para a definição da proposta deste mestrado, utilizaram-se as evidências obtidas nos estudos anteriores.

A proposta deste mestrado é o desenvolvimento de um assistente virtual, denominado CoderBot, com foco em auxiliar na aprendizagem de programação por meio da Abordagem Baseada em Exemplos. O CoderBot adota princípios fundamentados em estratégias de aprendizagem ativa, proporcionando exemplos assertivos (corretos) e errôneos. Essa abordagem tem como objetivo enriquecer as concepções mentais dos alunos, proporcionando uma experiência mais eficaz na assimilação de conhecimentos. Os atributos selecionados para a construção dos exemplos foram extraídos do *template* definido no estudo apresentado no Capítulo 4.

5.2 Proposta Inicial do CoderBot

O CoderBot é uma ferramenta educacional que se destaca pela inovação, especialmente no ensino de programação. Trata-se de um assistente virtual que oferece uma abordagem interativa e eficaz para que os alunos aprimorem suas habilidades de programação. A ferramenta fornece exemplos de códigos corretos, orientando os usuários passo a passo, além de apresentar códigos incorretos, estimulando a identificação de problemas e oferecendo *feedback* imediato. O assistente virtual CoderBot pode ser acessado em: <http://chatcoderbot.s3-website-sa-east-1.amazonaws.com/>.

O CoderBot tem os seguintes objetivos educacionais:

- Apoiar a aprendizagem de programação em cursos de graduação, por meio de uma abordagem baseada em exemplos, com foco na prática e no desenvolvimento da lógica de programação;
- Facilitar a compreensão da lógica de programação, utilizando exemplos corretos e incorretos como ferramentas de aprendizado;
- Estimular o desenvolvimento do pensamento computacional, incentivando os estudantes a analisar, interpretar e solucionar problemas de código;
- Oferecer *feedback* imediato e personalizado, ajudando os estudantes a identificar erros e a aprimorar suas habilidades de programação;
- Promover um ambiente de aprendizado interativo e engajador, que estimule a participação ativa dos estudantes.

Além disso, o CoderBot integra-se a um site intuitivo, tornando o aprendizado de programação mais acessível e engajadora. Essa integração promove uma experiência de aprendizagem dinâmica e eficiente. Com sua interface amigável e recursos perspicazes, o CoderBot se estabelece como um recurso valioso para estudantes e educadores, contribuindo para o avanço do ensino de programação.

A seguir, será apresentado um cenário de uso do CoderBot que foi gerado com apoio de Inteligência Artificial[1]. Esse cenário destaca a aplicabilidade prática do assistente virtual em um contexto educacional, proporcionando uma visão holística do processo de aprendizagem.

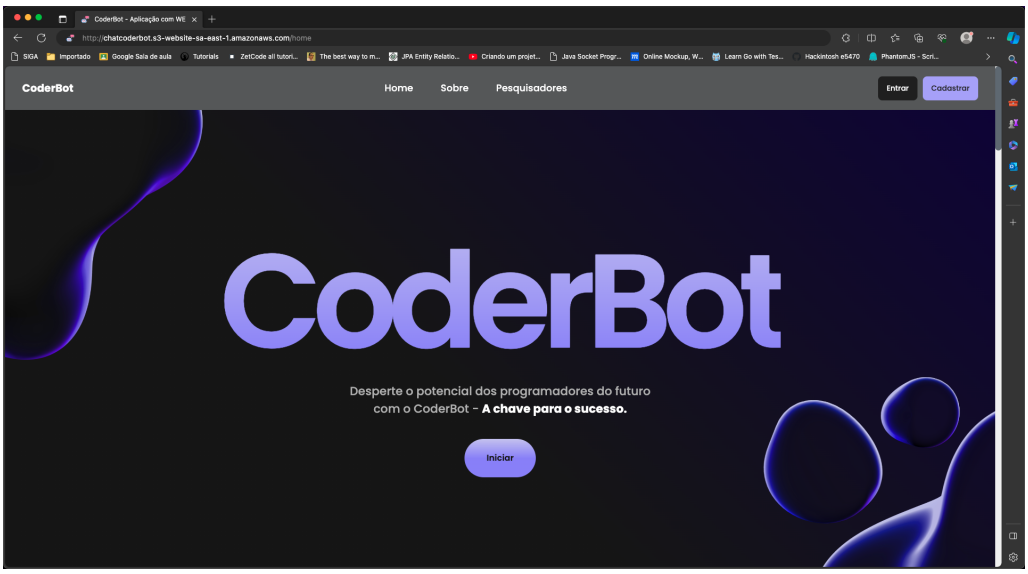
[1] Inteligência artificial desenvolvida pela OpenAI (<https://openai.com/chatgpt>).

Em sua jornada de aprendizado em programação, Ana Joaquina da Silva, uma estudante iniciante do curso de Engenharia de Software da Universidade Federal do CoderBot, recebeu dos professores da disciplina de Algoritmos e Programação uma lista de exercícios de sobre vetores para praticar após a aula. Para aprimorar seus conhecimentos, ela decidiu utilizar o assistente virtual CoderBot. Para isso, Ana acessa a plataforma CoderBot por meio do site. Ela decide criar uma conta para aproveitar todas as funcionalidades da ferramenta. Após realizar o login, Ana é recebida pela tela inicial do CoderBot. Na tela inicial, Ana explora os diversos tópicos disponíveis, procurando por algo que corresponda ao seu atual ponto de estudo. Optando por aprimorar seus conhecimentos em vetores, ela seleciona o conteúdo relevante. Dentro do tópico de vetores, Ana encontra diferentes *worked examples* disponíveis. Ela escolhe um exemplo que parece desafiador, mas interessante. Ao selecionar o exemplo, Ana é apresentada à descrição do problema, ao resultado esperado e às perguntas reflexivas. Ana decide começar com um exemplo correto. Ela segue os passos apresentados, analisando a explicação detalhada e o código passo a passo. Se Ana encontra alguma dificuldade, ela pode escolher explorar um exemplo incorreto para entender os possíveis erros comuns. Após cada exemplo, Ana recebe *feedback* interativo do CoderBot. Seja escolhendo um exemplo correto ou incorreto, a ferramenta fornece orientações precisas, destacando pontos-chave e oferecendo explicações claras. O *feedback* instantâneo ajuda Ana a consolidar seus conhecimentos. Ana decide praticar com diversos *worked examples* dentro do mesmo tópico. Ela escolhe exemplos com graus crescentes de dificuldade para desafiar suas habilidades. A cada interação, Ana ganha confiança em sua capacidade de compreender e resolver problemas relacionados a vetores. Após uma sessão produtiva de estudo, Ana decide finalizar sua interação com o CoderBot. Ela tem a opção de retornar à tela inicial para explorar outros tópicos, revisitar seus registros ou simplesmente encerrar a sessão. Como resultados do uso constante do CoderBot, Ana percebeu melhorias significativas em sua compreensão de vetores e em suas habilidades de programação. Isso ocorre devido a Ana usar dos exemplos que aprendeu na ferramenta para construir seus próprios mapas mentais. A abordagem interativa e personalizada do assistente virtual contribui para um ambiente de aprendizado eficaz e dinâmico.

A seguir, serão apresentados o CoderBot e sua forma de utilização, bem como sua estrutura de páginas, onde estudantes podem explorar tópicos e praticar programação.

A Figura 5 ilustra a tela inicial do site, destacando o menu com *links* para a tela principal (*Home*), informações sobre a ferramenta (*Sobre*), os pesquisadores envolvidos e botões para realizar login, acessar a aplicação e cadastrar novos usuários.

Figura 5 – Tela Inicial



Fonte: De autoria própria.

Ao acessar o *link Sobre* (Figura 6), é possível visualizar informações da ferramenta como, funcionamento, conteúdos disponíveis e suas principais vantagens.

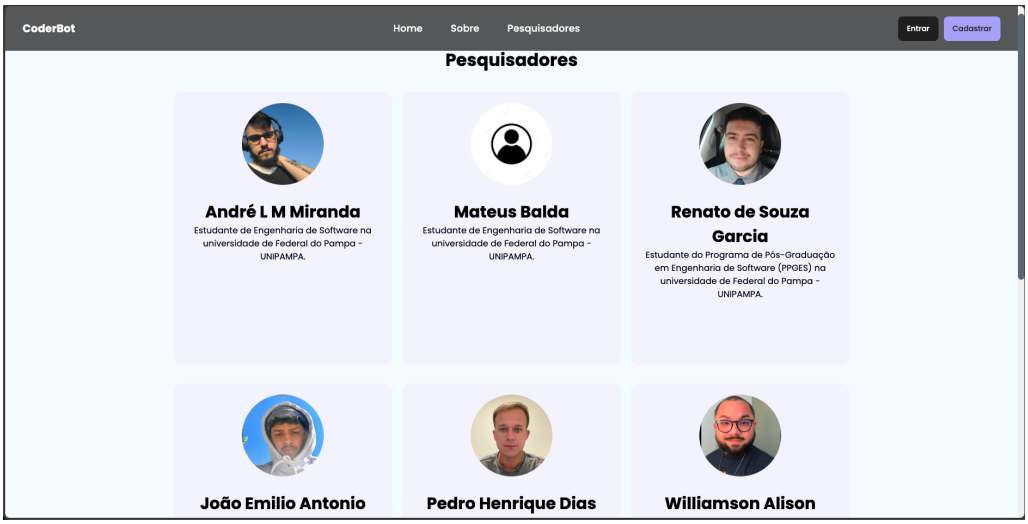
Figura 6 – Tela Sobre

Fonte: De autoria própria.

Na página *Pesquisadores* (Figura 7), os usuários podem consultar dados e informa-

ções sobre os membros diretamente envolvidos no desenvolvimento e no aprimoramento contínuo do CoderBot.

Figura 7 – Tela Pesquisadores

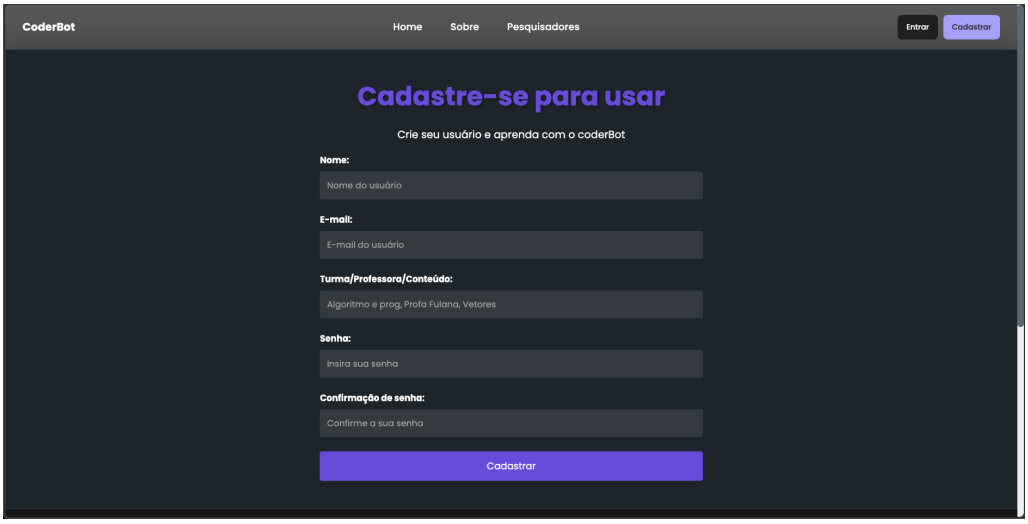


Fonte: De autoria própria.

Ao clicar no botão *Cadastrar*, o usuário será direcionado para a tela de registro, onde são requisitadas informações essenciais, como **nome, e-mail, turma/professor/conteúdo, senha e confirmação de senha**, conforme mostrado na Figura 8.

Ao concluir o processo de cadastro, o usuário é automaticamente autenticado no sistema, passando a ter acesso à interface do assistente virtual.

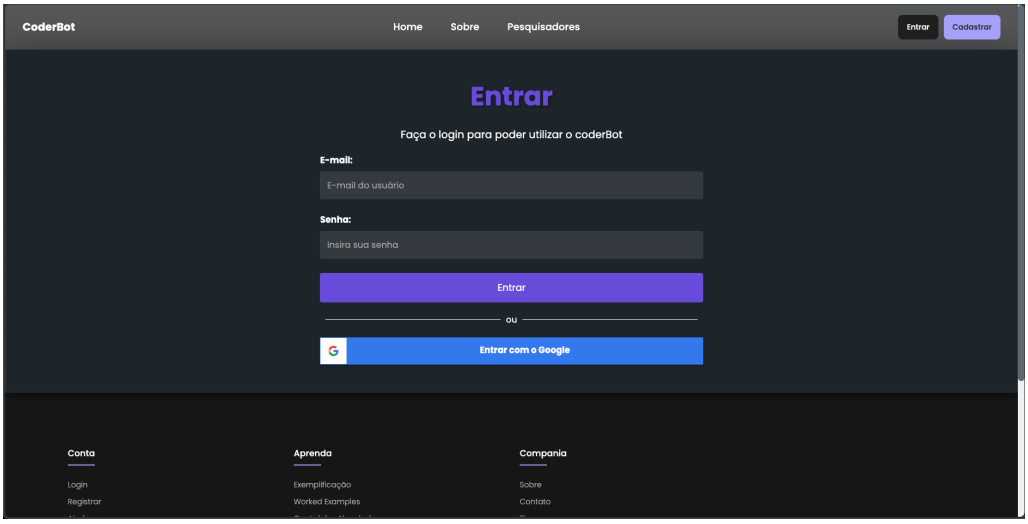
Figura 8 – Tela de Cadastro



Fonte: De autoria própria.

Se o usuário já tiver realizado o cadastro anteriormente, ele poderá acessar o sistema utilizando suas credenciais, conforme apresentado na Figura 9.

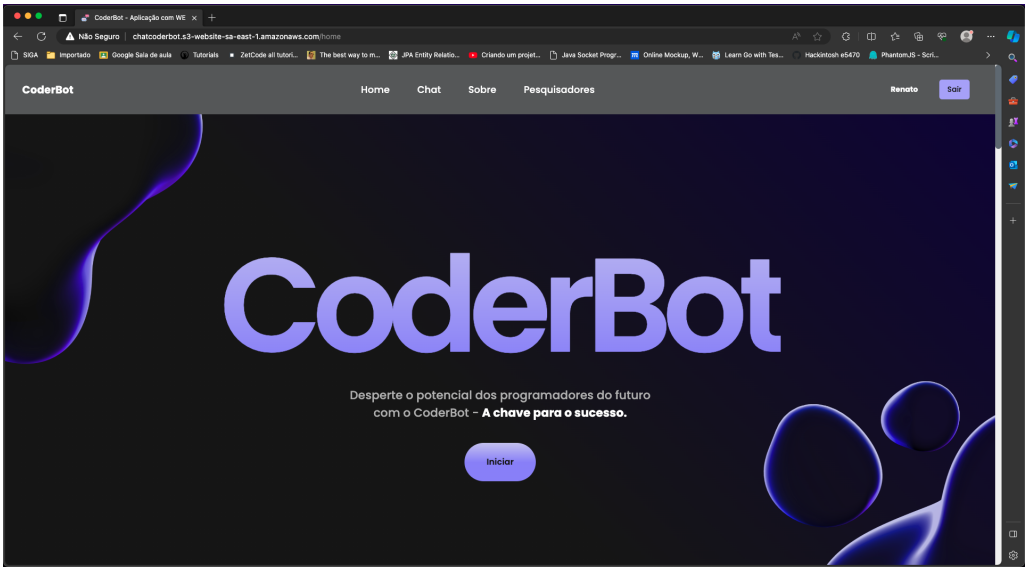
Figura 9 – Tela de Acesso (*Login*)



Fonte: De autoria própria.

Na tela inicial, após a autenticação do usuário “Renato”, é perceptível o surgimento do menu “Chat”, como ilustrado na Figura 10. Esse menu fornece acesso às funcionalidades de chat do CoderBot, proporcionando uma interface intuitiva para que o usuário interaja e usufrua dos recursos educacionais disponíveis na plataforma.

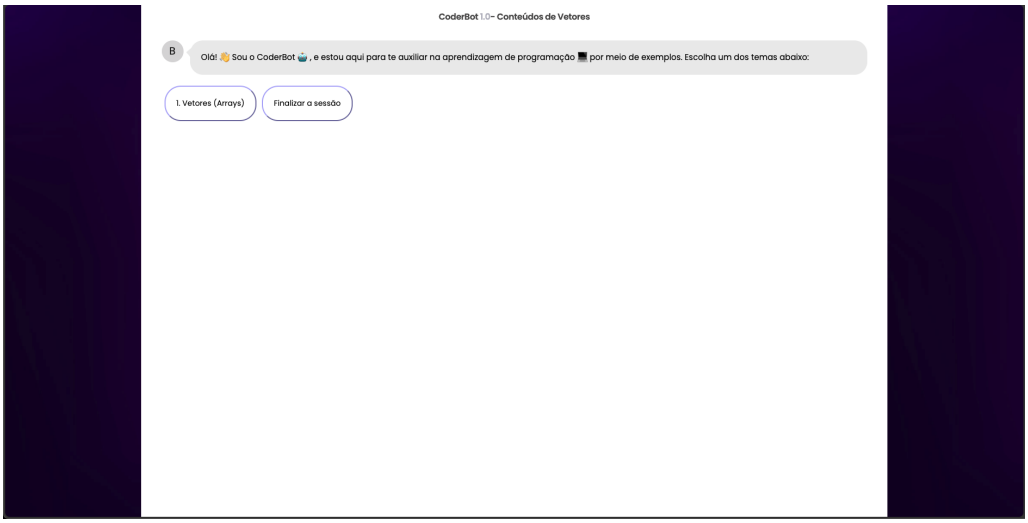
Figura 10 – Tela de Inicial Autenticada



Fonte: De autoria própria.

Ao acessar o menu “Chat” após a autenticação, o usuário é redirecionado para a tela representada na Figura 11, onde recebe uma mensagem de boas-vindas, seguida pelos conteúdos de programação disponíveis no assistente virtual.

Figura 11 – Tela de Apresentação do CoderBot

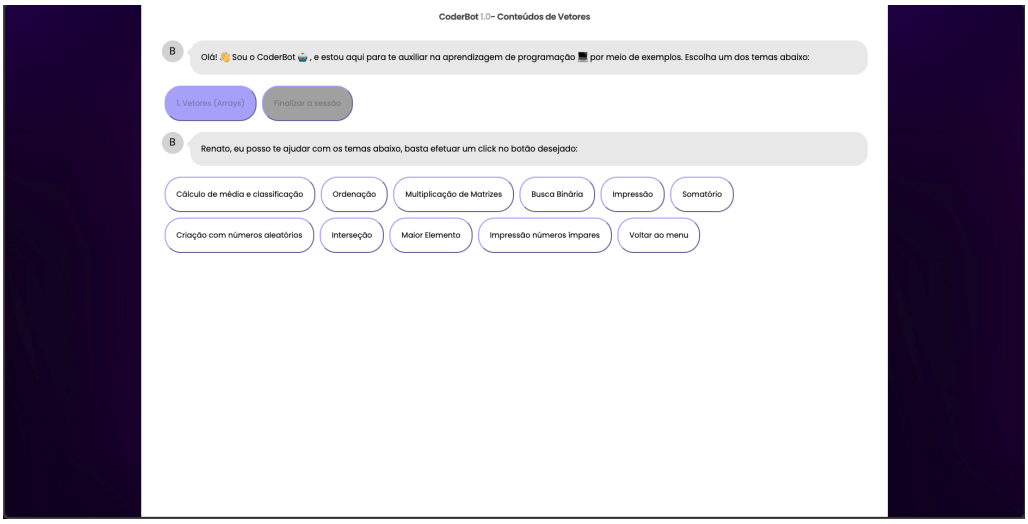


Fonte: De autoria própria.

A tela foi ajustada para focar no assistente virtual, com conteúdos exibidos como botões para interação intuitiva. Um botão permite encerrar a sessão e retornar à tela inicial (Figura 10).

Ao selecionar “1. Vetores (Arrays)” na Figura 11, o estudante acessa a tela da Figura 12, com os *worked examples* de Vetores.

Figura 12 – Tela de Apresentação do CoderBot - Conteúdo Vetores



Fonte: De autoria própria.

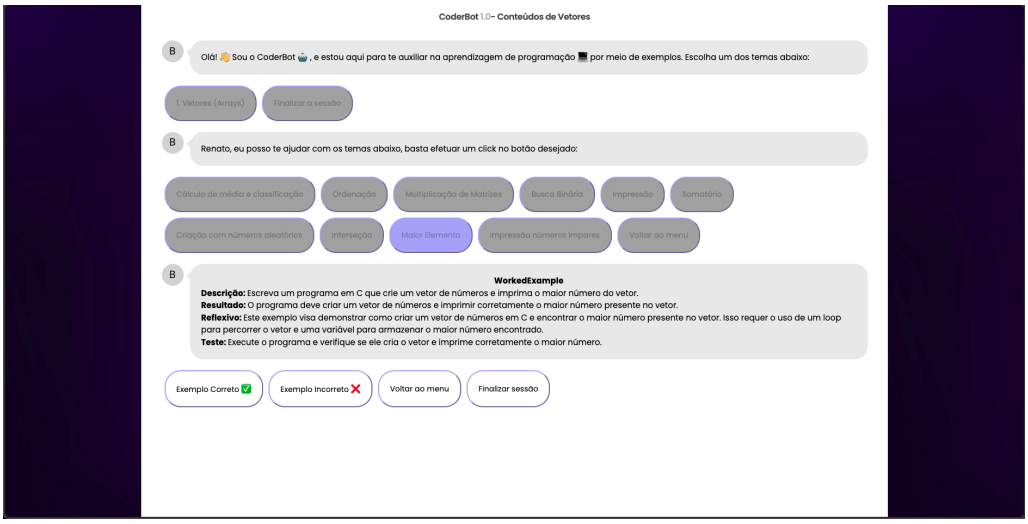
O botão “Voltar ao menu” retorna ao menu de conteúdos. Embora apenas o menu de Vetores tenha sido exibido como demonstração, é possível habilitar outros tópicos, como Listas e Funções, adaptando os conteúdos às necessidades do estudante.

Os exemplos são organizados pelos termos do título, e a seleção ocorre ao clicar no botão correspondente, redirecionando o estudante para a tela ilustrada na Figura 13.

Nessa etapa, são apresentados elementos essenciais do *template* descrito no Capítulo 4, como a **Descrição** do exemplo, o **Resultado** esperado, o aspecto **Reflexivo** e as instruções para os **Testes** da implementação.

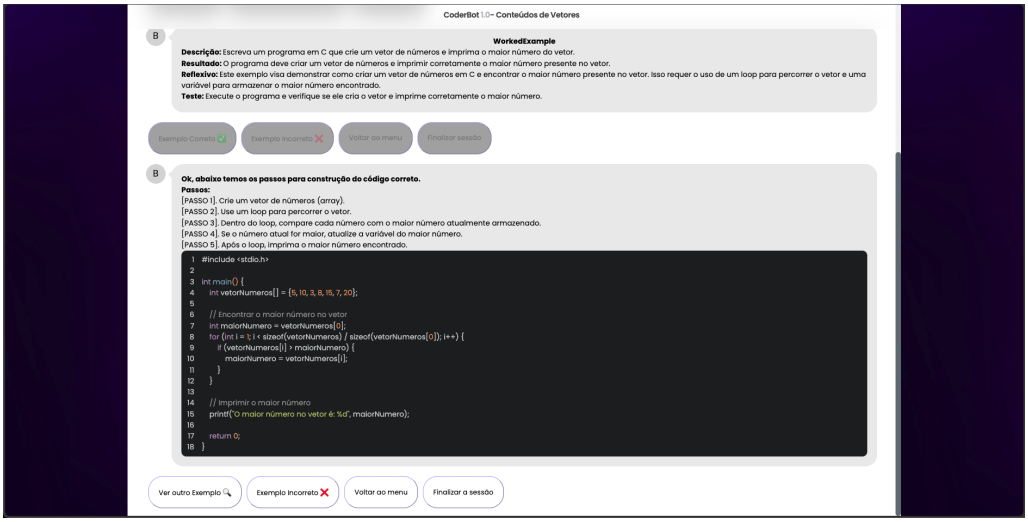
Em seguida, o estudante acessa um menu com as opções **Exemplo Correto**, **Exemplo Incorreto**, retorno ao menu inicial ou encerramento da sessão. Ao escolher **Exemplo Correto**, é direcionado à tela da Figura 14, que traz instruções detalhadas.

Figura 13 – Tela de Apresentação do CoderBot - Descrição do *Worked Example*



Fonte: De autoria própria.

Figura 14 – Tela de Apresentação do CoderBot - *Worked Example* - Correto



Fonte: De autoria própria.

Caso o estudante opte pela opção **Exemplo Incorreto**, o assistente apresentará uma solução com erros. Logo abaixo do código incorreto, é exibida a forma de **Teste**, seguida por uma pergunta ao estudante sobre sua capacidade de identificar onde ocorre o erro no exemplo.

A pergunta é composta por cinco opções de resposta: quatro alternativas para identificar o erro e uma para indicar que o estudante não conseguiu localizá-lo a partir do exemplo, conforme ilustrado na Figura 15.

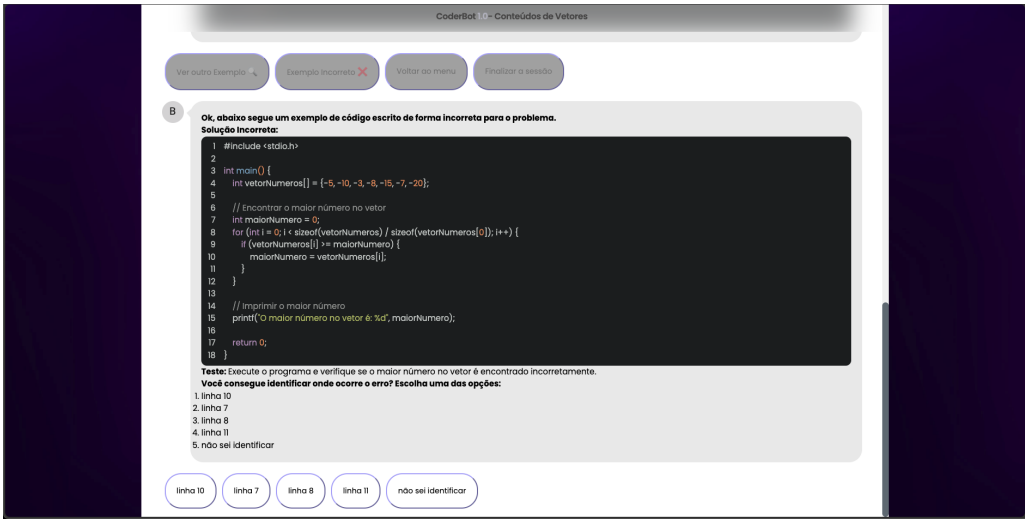
A escolha de uma opção no **Exemplo Incorreto** conduz a três possibilidades:

- O estudante escolhe a opção correta, é parabenizado e recebe um *feedback* explicando o motivo do erro no código. Em seguida, é apresentado o código correto (Figura 16).

- O estudante escolhe uma opção incorreta, recebe um *feedback* informando que a alternativa escolhida está errada e é informado sobre onde o erro ocorre. Posteriormente, o assistente apresenta os passos corretos para resolver o problema, seguidos pelo código correto (Figura 17).

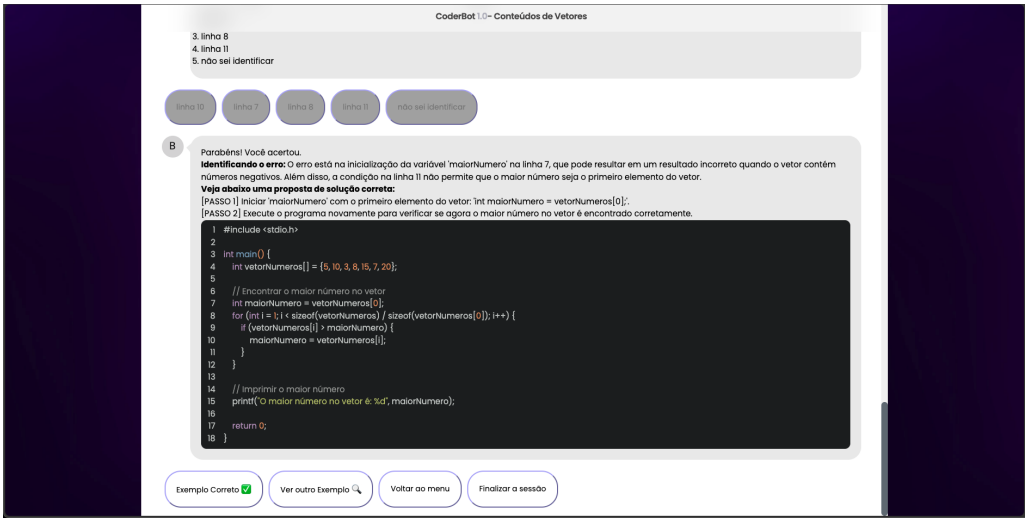
- O estudante escolhe a opção “não sei identificar”, recebe um *feedback* indicando onde o erro está e, em seguida, o assistente apresenta os passos para desenvolver uma solução correta, seguido pelo código correto (Figura 18).

Figura 15 – Tela de Apresentação do CoderBot - *Worked Example* - Incorreto



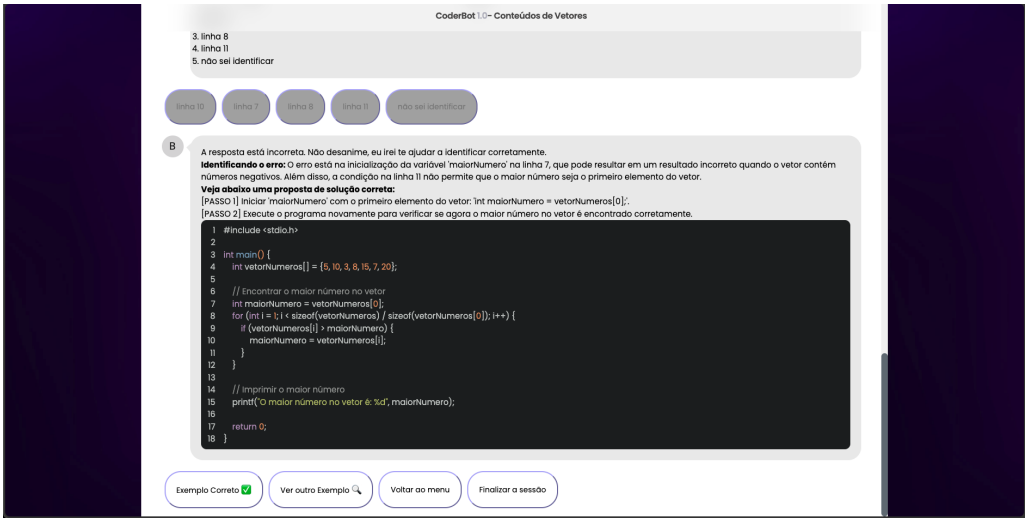
Fonte: De autoria própria.

Figura 16 – Tela de Apresentação do CoderBot - *Worked Example* - Alternativa Correta



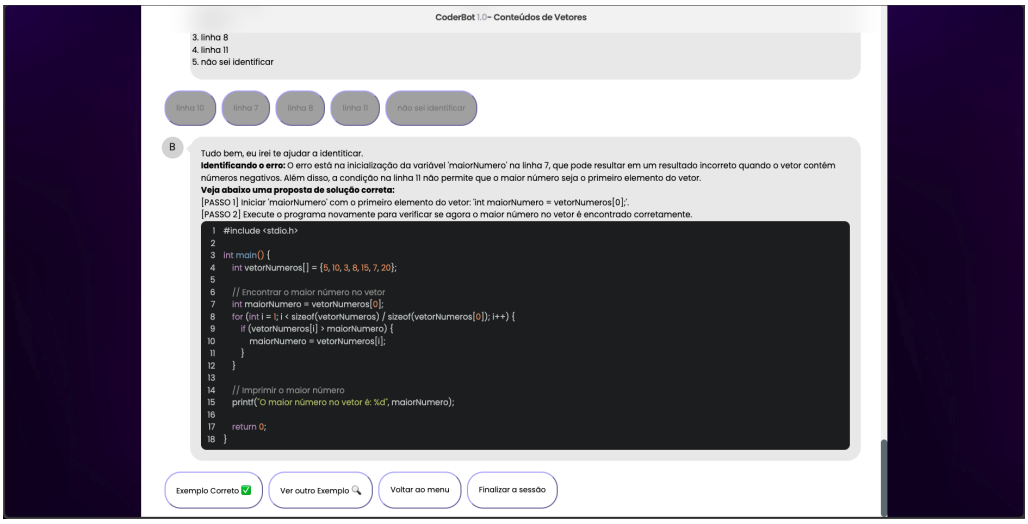
Fonte: De autoria própria.

Figura 17 – Tela de Apresentação do CoderBot - *Worked Example* - Alternativa Incorreta



Fonte: De autoria própria.

Figura 18 – Tela de Apresentação do CoderBot - *Worked Example* - Alternativa “não sei identificar”



Fonte: De autoria própria.

Vale ressaltar que, em ambas as telas de **Exemplo Correto** e **Exemplo Incorreto**, suas opções oferecem a possibilidade de alternar para o exemplo contrário ao qual o estudante se encontra, além de retornar ao menu inicial, que leva à tela apresentada na Figura 11, ou mesmo finalizar a sessão, o que direciona para a tela mostrada na Figura 13.

5.3 Considerações Finais

O assistente virtual CoderBot, como apresentado, incorpora uma abordagem denominada aprendizagem baseada em exemplos, implementando a técnica de *worked exam-*

ples, a qual foi examinada e investigada no contexto de ensino de programação. Além disso, utiliza atributos de um *template* resultante do estudo apresentado no Capítulo 4. As informações extraídas do mapeamento sistemático da literatura trouxeram aspectos relevantes para a construção do CoderBot, bem como para a forma de avaliação com a qual o mesmo foi validado. O CoderBot foi disponibilizado sob licença *Creative Commons*.

Como todo sistema, vale ressaltar que existem vantagens e desvantagens. Iniciando pelas limitações, é necessário entender que:

- O CoderBot já possui sua primeira versão finalizada, cujo qual passou por etapa de experimentação, onde foram apontadas lacunas que podem ser identificadas como possíveis novos requisitos em uma evolução do software;
- Há a necessidade de constantes avaliações para garantir que os exemplos e o próprio CoderBot continuem eficazes e aderentes ao seu objetivo;
- Existe também a necessidade do fator humano, tendo em vista que o CoderBot precisa ser alimentado por professores que desejam utilizá-lo. E, ainda sobre o fator humano, temos os exemplos produzidos por docentes e as próprias definições de erros para exemplos errôneos, que podem ser conflituosas em erros semânticos ou sintáticos;
- Atualmente, muitos dos *chatbots* ou assistentes virtuais são baseados em Inteligência Artificial (IA) generativa, capazes de construir soluções a partir de uma requisição detalhada. Entretanto, é necessário ter um certo grau de conhecimento prévio para formular uma requisição que traga exatamente o esperado pelo usuário;
- IAs generativas são classificadas de forma ampla, possuindo suas próprias formas de padronização, que são diferentes das propostas neste trabalho. Entretanto, poderia ser interpretado como uma limitação do CoderBot a ausência da geração de seus próprios exemplos, devido a não fazer uso de uma IA generativa.

Mesmo diante de algumas limitações deve ser de conhecimento que os pontos positivos do sistema são de grande importância, sendo eles:

- O CoderBot é uma ferramenta educacional inovadora que fornece a apresentação de exemplos de código corretos e incorretos, o que amplia consideravelmente o apoio à construção de mapas mentais para os discentes que o utilizam;

- Outro aspecto é que ele serve como um facilitador de aprendizagem por disponibilizar exemplos estruturados e personalizados;
- O CoderBot possibilita que os professores façam ajustes no processo de ensino;
- Os próprios exemplos se tornam um repositório dentro da própria ferramenta;
- O CoderBot estimula uma reflexão contínua para os estudantes, fazendo-os aprender a construção correta e identificar erros que possam conter nos códigos;
- Com o CoderBot, os professores podem disponibilizar materiais com base nas suas necessidades, e os estudantes podem consumir os materiais de acordo com seu ritmo, ditando uma forma personalizada de aprendizagem.

À frente de toda essa contextualização, e mesmo sem trabalhar diretamente com IA generativa, o CoderBot tem um apelo de eficácia no apoio à aprendizagem de programação. Tal aspecto pode ser observado em Lidén and Nilros (2020), que indica o interesse em começar com pouca funcionalidade e progredir, melhorando continuamente. Isso porque implementações mais básicas costumam ser mais fáceis de engajar estudantes, quando comparadas com aplicações complexas que utilizam IA generativa; entretanto, devem ser consideradas como aspectos a serem implementados futuramente.

6 AVALIAÇÃO DO CODERBOT

Este capítulo apresenta o processo de avaliação do assistente virtual proposto.

6.1 Considerações Iniciais sobre o Capítulo

Diante das evidências levantadas pelo MSL e do estudo exploratório sobre a construção do *template* de *worked example* para o ensino de programação, foram elaborados pela equipe os requisitos para desenvolvimento do CoderBot, um assistente virtual para apoiar a aprendizagem de programação. Na etapa de desenvolvimento do CoderBot, foram realizados testes manuais, concomitantemente, para verificação da conformidade da ferramenta em relação aos requisitos, bem como para a busca de eventuais problemas que poderiam ocorrer durante a sua execução.

Mesmo diante de um desenvolvimento cuidadoso, é necessário disponibilizar o software para avaliar a viabilidade deste, além de identificar aspectos vantajosos, novos requisitos ou mesmo potenciais melhorias para a ferramenta. Para tal, realizou-se um estudo experimental exploratório que será detalhado nas próximas seções.

6.2 Planejamento do Estudo

Nesta subseção será apresentado o planejamento do estudo experimental realizado para avaliação do assistente virtual CoderBot.

6.2.1 Contexto do Experimento

O estudo experimental teve como foco principal avaliar o potencial do CoderBot como ferramenta de apoio à aprendizagem de programação. A execução do experimento ocorreu em duas universidades e dois institutos, onde os participantes cursavam disciplinas iniciais de Computação em cursos de graduação. As instituições participantes foram: Universidade Federal do Pampa (UNIPAMPA), a Universidade Federal do Amazonas (UFAM), o Instituto Federal de Mato Grosso do Sul (IFMS) e o Instituto Federal do Pará (IFPA).

6.2.2 Planejamento do estudo

O planejamento do estudo teve início com a identificação dos artefatos necessários para a execução do experimento. Foram definidos formulários a serem preenchidos pelos participantes, utilizando a ferramenta *Google Workspace*[1], sendo eles:

- Termo de Consentimento Livre e Esclarecido (TCLE), adotado para garantir a confidencialidade dos dados fornecidos e o anonimato dos participantes;
- Formulário de caracterização de perfil, utilizado para coletar informações detalhadas sobre os conhecimentos e características dos participantes (professores e estudantes);
- Documento contendo o roteiro do estudo com o link do CoderBot, a lista de exercícios a ser realizada e as instruções necessárias para a realização do experimento;
- Ficha de avaliação da ferramenta (percepção de uso), composta por um questionário pós-uso com perguntas abertas sobre a percepção dos estudantes em relação ao CoderBot;
- Definição das métricas e questionários a serem coletadas pelo assistente virtual, incluindo informações sobre cliques do usuário, áreas de acesso, fluxos seguidos e tempos despendidos em diferentes momentos do software. Esses dados foram utilizados para compreender melhor o uso da ferramenta e permitir inferências sobre sua utilização;
- Exemplos selecionado a partir das informações coletadas junto aos docentes. Foi definida a carga de *worked examples* adaptados para a linguagem de programação estudada por cada turma. A produção dos exemplos iniciou-se com a equipe construindo exemplos didáticos baseados no *template*, seguida pela utilização do *ChatGPT*[2] para a geração de outros exemplos. Após a produção, os exemplos foram avaliados pelos pesquisadores e, posteriormente, integrados ao assistente virtual;
- Lista contendo cinco exercícios para aplicação aos estudantes, os quais deveriam ser resolvidos utilizando como apoio os exemplos disponíveis no assistente virtual.

[1] Google Workspace é uma suíte de produtividade baseada em nuvem desenvolvida pelo Google, que inclui ferramentas como Gmail, Docs, Drive, Formulários e outros.

[2] O ChatGPT é um modelo de linguagem baseado na arquitetura de Transformers usando LLM GPT (Generative Pre-trained Transformer), desenvolvido pela OpenAI.

Vale destacar que todo o planejamento foi realizado e validado por uma equipe composta por um aluno de mestrado, três estudantes de graduação e dois professores doutores que orientaram o desenvolvimento deste trabalho. Durante o processo, antes da execução do experimento, sempre que os pesquisadores identificavam algum problema, uma nova versão dos artefatos era elaborada, garantindo maior precisão e aderência aos objetivos do experimento.

6.2.3 Execução do experimento

Inicialmente, foi realizado um estudo piloto com dois estudantes de graduação da Unipampa para verificar a viabilidade e eficácia do experimento em alcançar seus objetivos. Os resultados foram satisfatórios, não sendo necessário realizar ajustes no roteiro do estudo.

O experimento foi conduzido em quatro instituições, envolvendo um total de 103 estudantes, distribuídos em cinco turmas. Cada turma participou do experimento em dias distintos, sempre contando com o suporte do docente responsável pela disciplina e da equipe de pesquisa envolvida neste trabalho.

Os docentes das Instituições de Ensino Superior (IES) foram convidados por e-mail, que apresentava o objetivo do estudo e fornecia orientações detalhadas. Após aceitarem participar, um dos pesquisadores entrou em contato com cada docente para combinar a data de condução do estudo e, posteriormente, enviou um e-mail com o roteiro de preparação. Os quatro docentes que aceitaram participar passaram por um contato inicial para que se compreendesse a disciplina que ministravam, o momento (conteúdo) da ementa em que se encontravam e os horários de aula, a fim de definir a data de aplicação.

Os docentes, aqui identificados como **D1**, **D2**, **D3**, **D4** e **D5**, são vinculados à rede pública federal de ensino no Brasil: **D1** e **D2**, à Universidade Federal do Pampa; **D3**, ao Instituto Federal de Mato Grosso do Sul; **D4**, ao Instituto Federal do Pará; e **D5**, à Universidade Federal do Amazonas. Ao todo, o experimento contou com a participação de 103 estudantes, sendo 19 da UNIPAMPA, 15 do IFMS, 18 do IFPA e 51 da UFAM.

A Tabela 10 apresenta um panorama relacionando os dados de contextualização do experimento.

Vale a pena destacar que foi observada uma diferença na execução do experimento em relação ao tempo, devido à logística e à distribuição dos horários das disciplinas. O tempo médio para a realização das atividades foi de 136 minutos, variando entre 40 mi-

Tabela 10 – Caracterização das Turmas.

Instituição	Docente	Curso	Disciplina	Semestre	Conteúdo	Estudantes
UNIPAMPA	D1	Ciência da Computação	Algoritmos e Programação para Computação	1	Funções	8
UNIPAMPA	D2	Engenharia de Software	Algoritmos e Programação	1	Arrays	11
IFMS	D3	Tecnologia em Análise e Desenvolvimento de Sistemas	Programação de Computadores	2	Listas	15
IFPA	D4	Tecnologia em Análise e Desenvolvimento de Sistemas	Linguagem de Programação 1	2	Arrays	18
UFAM	D5	Engenharia de Software	Algoritmos e Estruturas de Dados 1	2	Funções	51

Fonte: De autoria própria.

nutos (mínimo) e 150 minutos (máximo), o que indica a viabilidade do uso do CoderBot durante as aulas.

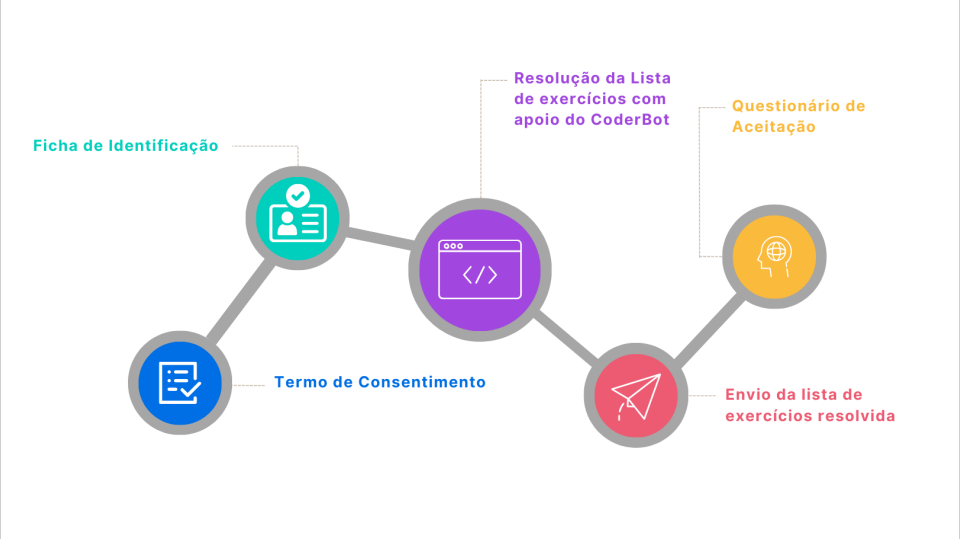
Na data agendada, o estudo foi realizado com os estudantes. A ordem de execução do experimento encontra-se detalhada na Figura 19. Durante o experimento, os docentes atuaram como moderadores, sendo responsáveis por transmitir as informações sobre as atividades aos participantes. Inicialmente, foi solicitado que os estudantes assinassem o TCLE, concordando em participar do estudo e autorizando a utilização dos dados de seus trabalhos para análise. Todos os participantes aceitaram e assinaram o termo. Além disso, foram informados, no momento da entrega do documento, que poderiam optar por não participar do experimento a qualquer momento, assegurando que a participação e o fornecimento dos dados ocorreram de forma voluntária.

Em seguida, os estudantes responderam a um formulário de caracterização de perfil, contendo perguntas relacionadas à sua experiência em programação. A maioria relatou não possuir experiência prévia, dado que se encontravam nos semestres iniciais do curso. Posteriormente, os moderadores conduziram um treinamento sobre o CoderBot, abordando suas funcionalidades e modo de uso, além de explicar como a ferramenta poderia ser utilizada para auxiliar nas atividades de programação.

Posteriormente, os moderadores distribuíram uma lista de exercícios contendo atividades de programação, as quais deveriam ser resolvidas pelos participantes utilizando o CoderBot como ferramenta educacional de suporte. Após a conclusão das atividades, os estudantes enviaram as listas de exercícios, com as resoluções anexadas, aos professores, que as repassaram à equipe de desenvolvimento do CoderBot. Em seguida, os participantes receberam questionários para avaliar sua percepção em relação a:

- Aceitação: O questionário de aceitação foi elaborado com base nos indicadores do Modelo de Aceitação de Tecnologia (TAM), propostos por Venkatesh e Davis

Figura 19 – Etapas de condução do experimento



Fonte: De autoria própria.

(2000) (ver Tabela 12). A teoria subjacente a este modelo busca compreender como os usuários aceitam e utilizam novas tecnologias. Dessa forma, é possível coletar índices relacionados à adoção de uma tecnologia.

- Usabilidade: O *System Usability Scale* (SUS) é um questionário consolidado (ver Tabela 13), desenvolvido por Brooke *et al.* (1996), utilizado para medir aspectos como capacidade de aprendizagem, eficiência, memorização, minimização de erros e satisfação de aplicações.

- Experiência: O *User Experience Questionnaire* (UEQ), segundo Nakamura *et al.* (2019), é um questionário que utiliza uma escala de sete pontos e uma técnica composta por 26 adjetivos, distribuídos em seis fatores: (i) atratividade, (ii) perspicuidade, (iii) eficiência, (iv) confiabilidade, (v) estimulação e (vi) novidade. Os fatores são servem para avaliar a experiência do usuário em relação ao uso da ferramenta.

- Autoeficácia (*self-efficacy*) percebida: O questionário de autoeficácia foi baseado nos indicadores propostos por Raiche *et al.* (2023) (ver Tabela 15) para medir a autoeficácia dos participantes, avaliando suas crenças sobre a capacidade de desenvolver as atividades com o uso da ferramenta.

- Motivação Intrínseca: O *Intrinsic Motivation Inventory* (IMI) destina-se a avaliar

a motivação intrínseca do participante. O IMI é constituído por 22 afirmativas, distribuídas em quatro fatores: (i) Interesse/Prazer (mede o interesse e o prazer ao realizar uma atividade específica), (ii) Competência Percebida (avalia como os indivíduos se envolvem em uma atividade e quão eficazes se sentem ao realizar as tarefas), (iii) Pressão/Tensão (mede a pressão/tensão para finalizar uma tarefa com sucesso) e (iv) Escolha Percebida (avalia a percepção de tomada de decisões dos estudantes).

- Experiência de Fluxo: O *Flow Experience Questionnaire* (FLQ) é um questionário utilizado para medir imersão e engajamento, baseado na teoria do *flow*, definida por Csikszentmihalyi e Csikszentmihalyi (1975), que se refere ao estado mental de um participante em relação à atividade desenvolvida, caracterizado por um alto nível de foco durante o envolvimento. Jackson e Eklund (2002) desenvolveram e validaram o mecanismo da *Flow State Scale* (FSS), que permite mensurar o estado de *flow* em diferentes contextos. A aplicação do Questionário de Experiência de Fluxo foi realizada para medir as sensações psicológicas dos participantes durante o uso da ferramenta. Os participantes foram questionados sobre as questões apresentadas na Tabela 14. Vale ressaltar que, nesta pesquisa, foi utilizada a versão *short* do FSS, que analisa nove dimensões essenciais do modelo original proposto por Csikszentmihalyi (1997).

- Atitudes dos Estudantes: O *Attitude Questionnaire* (ATQ) foi aplicado para avaliar os comportamentos, crenças e atitudes dos estudantes em relação à utilização do CoderBot no apoio à aprendizagem de programação, para tal, utilizamos um questionário baseado em Likert (1932).

- Percepção de Aprendizagem Percebida: O *Perceived Learning Assessment Questionnaire* (PLAQ) foi utilizado para coletar indicadores sobre a forma como os participantes percebiam seu próprio processo de aprendizagem ao utilizar a ferramenta.

Todos os questionários utilizados podem ser encontrados no Apêndice A desta dissertação de Mestrado.

6.2.4 Análise de dados

A análise dos dados foi realizada de forma quantitativa utilizando métodos estatísticos para a definição de valores máximos e mínimos, média e desvio padrão. Os dados coletados das respostas ao questionário foram tabulados e, em seguida, utilizando a ferramenta R Studio, foram gerados gráficos de barras empilhadas para facilitar a visualização e análise. A análise respeitou as características de cada questionário e suas particularidades, que são especificadas para a sua interpretação. Vale ressaltar que houve um tratamento dos dados coletados. Para isso, adicionou-se nos questionários questões de controle, que objetivavam verificar se o estudante estava realmente prestando atenção no questionário e em suas respostas. Após esse tratamento, o número de respostas consideradas válidas foi para 88.

As respostas foram fornecidas pelos estudantes nos questionários TAM, IMI, FLQ, ATQ, PLAQ em uma escala *Likert* de cinco pontos, variando nas opções "Discordo totalmente", "Discordo", "Neutro", "Concordo" e "Concordo totalmente". Já os dados de respostas sobre a autoeficácia foram fornecidos também em uma escala de cinco pontos, entretanto, as opções eram: "Muito Insatisfeito", "Insatisfeito", "Neutro", "Satisfeito" e "Muito Satisfeito".

No caso do questionário SUS, além da coleta dos dados e da classificação, há um cálculo a ser realizado para gerar um *score* final com o somatório dos valores obtidos. O formulário é composto de dez questões (ver Tabela 13), contendo opções de resposta em uma escala *Likert* variando de 1 (Discordo Fortemente) a 5 (Concordo Fortemente). O *score* do SUS é calculado para permitir uma avaliação sobre o grau de usabilidade do sistema. O cálculo do *score* leva em consideração todos os itens do questionário, onde as questões ímpares (1, 3, 5, 7, 9) recebem a nota atribuída pelo usuário na escala *Likert*, e essa nota é subtraída de 1 (nota do usuário - 1). Já para os itens pares (2, 4, 6, 8, 10), o cálculo é realizado subtraindo a nota recebida pelo usuário de 5 (5 - nota do usuário). A determinação do *score* final envolve o somatório dos valores das questões ímpares e pares, multiplicado por 2,5 (BROOKE et al., 1996). O valor obtido no questionário SUS varia de 0 a 100, onde pontuações abaixo de 60 são classificadas como ruins, entre 60 e 69 como fracas, entre 70 e 79 como médias, entre 80 e 89 como boas e 90 ou mais como excelentes.

O questionário UEQ, por sua vez, também utilizou a escala *Likert*, porém de sete pontos, aplicada ao contexto de cada questão, sendo determinados por 26 adjetivos especi-

ficados pelo questionário. O valor respondido indica a posição do participante em relação aos adjetivos apresentados. Por exemplo, quando questionados sobre o produto, a escala variava em sete pontos, onde 7 (sete) correspondia a “Agradável” e 1 (um) correspondia a “Desagradável”.

6.3 Resultados da Avaliação

6.3.1 Visão Geral dos Participantes

No início, foram identificados 115 estudantes de graduação que responderam ao questionário de caracterização. Entretanto, foram eliminados e-mails duplicados ou casos em que o aluno acessou primeiramente com um e-mail pessoal e depois modificou para o e-mail institucional. Com isso, o número de participantes foi reduzido para 103. A Tabela 11 apresenta uma visão geral dos participantes.

Tabela 11 – Caracterização dos participantes

Participantes	Possíveis Respostas	%
Gênero	Homem	73,04%
	Mulher	23,48%
	Não-binário	2,61%
	Não Informado	0,87%
Faixa Etária (em anos)	Menos de 21	58,26%
	Entre 21 e 30	38,26%
	Entre 31 e 40	2,61%
	Mais de 40	0,87%
Ano de Ingresso	Antes de 2015	0%
	2015 à 2018	1,74%
	2019 à 2022	20,68%
	Em 2023	77,39%
Experiência em Desenvolvimento	Nenhuma	22,61%
	Pouca	46,96%
	Média	25,22%
	Alta	5,22%
Experiência com Agentes Conversacionais	Nenhuma experiência	17,39%
	Baixa (Utilizou pelo menos uma vez)	12,71%
	Média (Utilizou de 1 a 4 vezes)	33,04%
	Alta (Utilizou mais de 4 vezes)	37,39%
Experiência em Aplicações Educacionais	Não conhece e não utilizou	5,22%
	Conhece, mas nunca utilizou	2,61%
	Utilizou com baixa frequência	42,61%
	Utiliza com frequência	49,57%

Fonte: De autoria própria.

No que tange ao aspecto de conhecimento, os docentes que ministravam as disciplinas em que o experimento foi realizado foram identificados como **D** e a numeração correspondente à ordem de execução do experimento, no caso, de 1 a 5. Deve-se observar que o **D2** compreende a execução por dois docentes que ministravam a matéria em conjunto.

A experiência dos docentes participantes foi relatada da seguinte forma: **D1** conta

com 1 ano de docência geral e em matérias de programação; **D2**, composta por dois docentes, sendo que um possui 27 anos de docência, atuando há pelo menos 10 anos com programação, e o segundo possui 4 anos de experiência, atuando tanto com docência quanto com programação. **D3** possui 10 anos e meio de experiência, atuando tanto com docência geral quanto com programação. **D4** atua como docente há 13 anos, sendo 8 anos com disciplinas de programação. Por fim, **D5** possui 9 anos de experiência em docência, dos quais 3 anos são em programação. Quase todos os docentes que participaram do experimento possuíam doutorado, exceto **D3**, que era especialista no momento do experimento.

Além disso, realizou-se uma análise específica sobre os comentários dos estudantes coletados a partir dos questionários. As respostas foram analisadas qualitativamente seguindo os procedimentos de codificação. O objetivo da análise qualitativa foi codificar, categorizar e sintetizar dados, a fim de identificar as dificuldades e os benefícios percebidos pelos estudantes após o uso do CoderBot.

Esta análise foi inspirada em Silva *et al.* (2019), que estabelece um procedimento de análise qualitativa em quatro etapas.

Na **primeira etapa**, foram analisados todos os comentários dos estudantes, filtrando aqueles sem resposta ou fora de contexto. Na **segunda etapa**, realizou-se a codificação aberta, criando códigos a partir das respostas dos participantes. Na **terceira etapa**, foi realizada a codificação axial, agrupando os códigos em subcategorias e categorias. Por fim, na **quarta etapa**, avaliou-se a consistência dos resultados, com a análise conduzida por um pesquisador e discutida com outros especialistas para minimizar viéses. Os resultados e discussões forneceram *insights* para a melhoria do CoderBot.

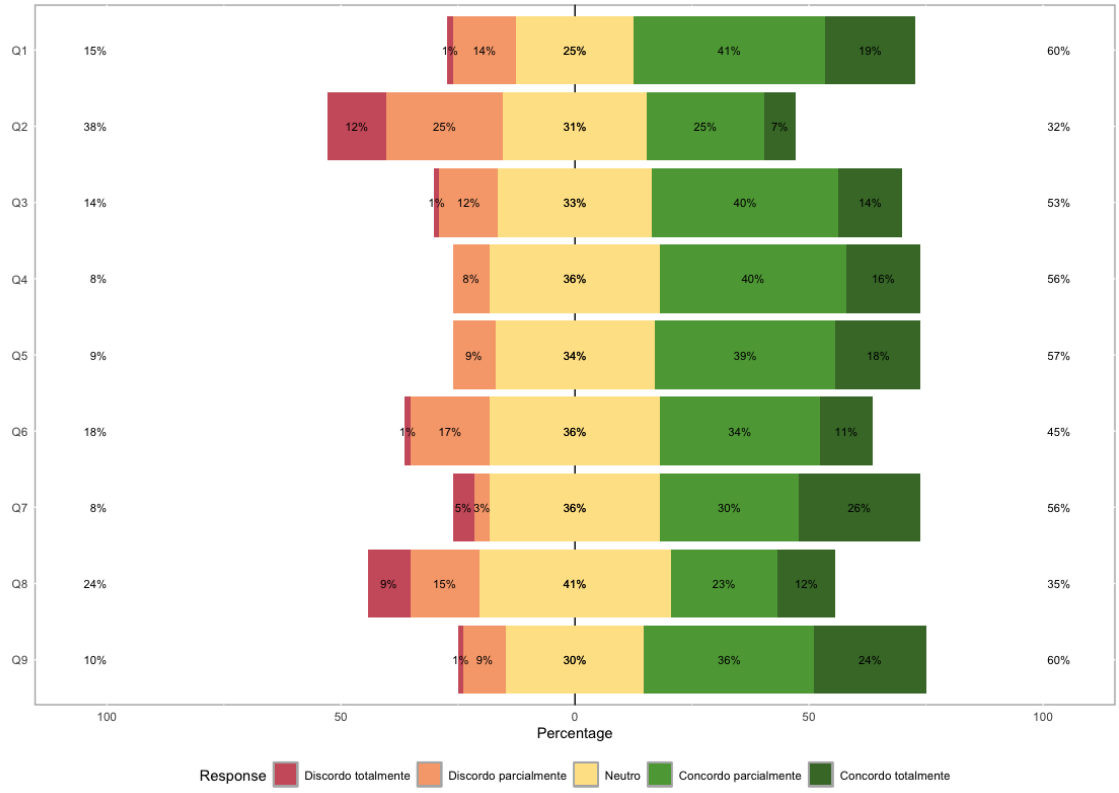
6.3.2 Resultados com relação à Percepção de Aprendizagem

6.3.2.1 Percepção dos Estudantes sobre Experiência de Fluxo

Com base na Figura 20, observa-se uma discordância de 37% em **Q2**, que aborda a realização das tarefas de maneira espontânea e automática. Esse resultado sugere que os participantes enfrentaram certa dificuldade em se familiarizar com a ferramenta.

Por outro lado, os resultados em **Q1** indicam que os participantes se sentiram competentes e confiantes para realizar as atividades propostas, com 60% de concordância. Além disso, em **Q9**, 60% dos participantes relataram ter tido uma experiência extrema-

Figura 20 – Percepção dos Estudantes sobre Experiência de Fluxo



Fonte: De autoria própria.

mente gratificante ao utilizar o CoderBot como apoio à aprendizagem.

Outro dado relevante é o de **Q5**, no qual 57% dos participantes afirmaram estar completamente focados durante o uso da ferramenta, o que reflete uma boa taxa de imersão. Esse aspecto é corroborado pelo comentário de P57: *"Com um conteúdo claro e direto facilitando a leitura. Os tópicos do passo a passo da execução do código me ajudaram a entender como funciona cada linha de comando."*

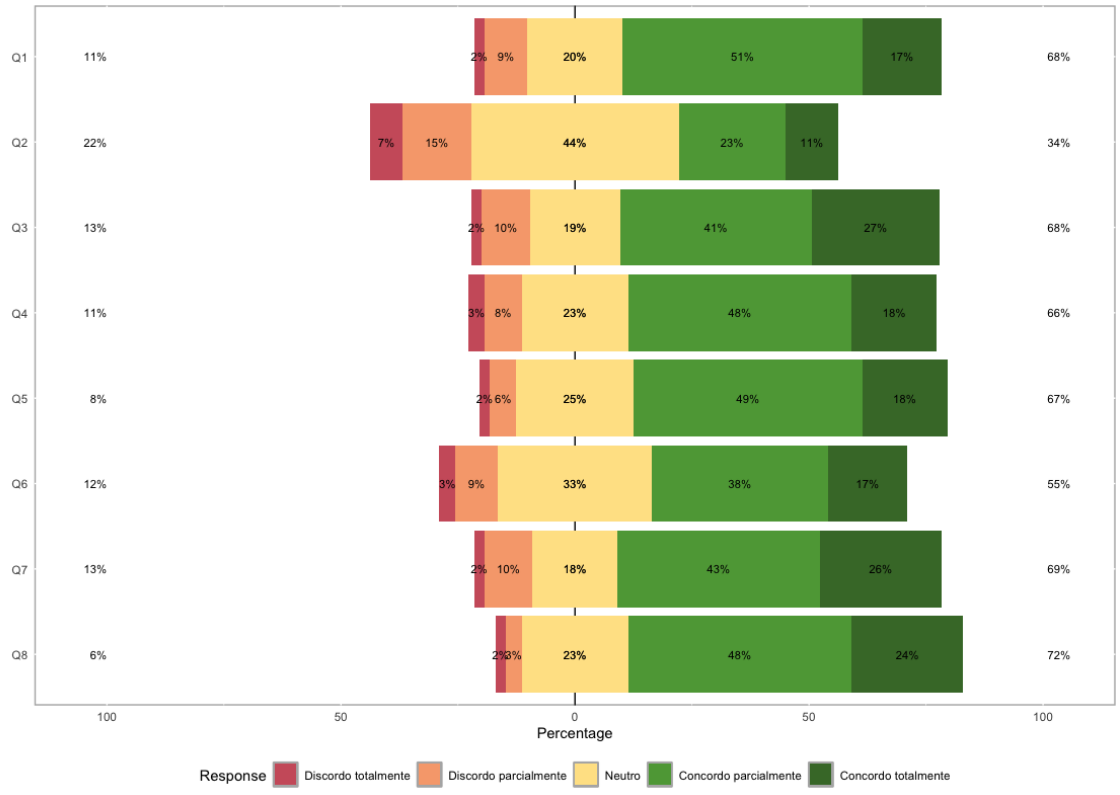
O maior ponto de neutralidade, com 41%, foi observado em **Q8**. Apesar da imersão relatada, os participantes não perceberam diferença significativa na passagem de tempo em comparação com a execução das tarefas sem o CoderBot. Esse resultado pode ser explicado pelo fato da ferramenta não se configurar como um jogo ou algo que gere entretenimento.

6.3.2.2 Percepção dos Estudantes sobre a Aprendizagem Percebida

Ao observar a Figura 21, é possível inferir uma alta taxa de concordância na maioria dos itens. Por exemplo, em **Q1**, 68% dos participantes responderam positivamente

sobre a contribuição da ferramenta para a aprendizagem do conteúdo ensinado na disciplina, como ilustrado no comentário do participante P48: *"Com o uso do CoderBot consegui entender trechos de código. Achei interessante o método de utilização dele."*. Em **Q4**, 66% dos participantes indicaram que o CoderBot os apoiou na interpretação dos conteúdos aprendidos na disciplina, o que é complementado por **Q5**, com 67% de concordância em relação à aplicação prática dos conteúdos. Os comentários dos participantes P58 e P64 corroboram essa análise. O participante P58 afirmou: *"Ajudou a melhorar, é bem simples, fácil de entender e manusear."*, enquanto P64 destacou: *"Melhorou, pois na maioria dos exercícios eu consegui resolver."*, evidenciando o auxílio na interpretação, entendimento e aplicação para a resolução de exercícios.

Figura 21 – Percepção dos Estudantes sobre Aprendizagem Percebida



Fonte: De autoria própria.

Apesar dos aspectos positivos destacados, o item **Q2** apresenta uma alta taxa de neutralidade (44%) e discordância (22%), superando em menos de 15% a taxa de concordância (34%). Isso sugere que a maioria dos alunos ficou neutra em relação à eficiência da ferramenta em comparação com outras atividades da disciplina. Por fim, o item **Q8** apresentou um ótimo resultado, indicando que a ferramenta colaborou significativamente

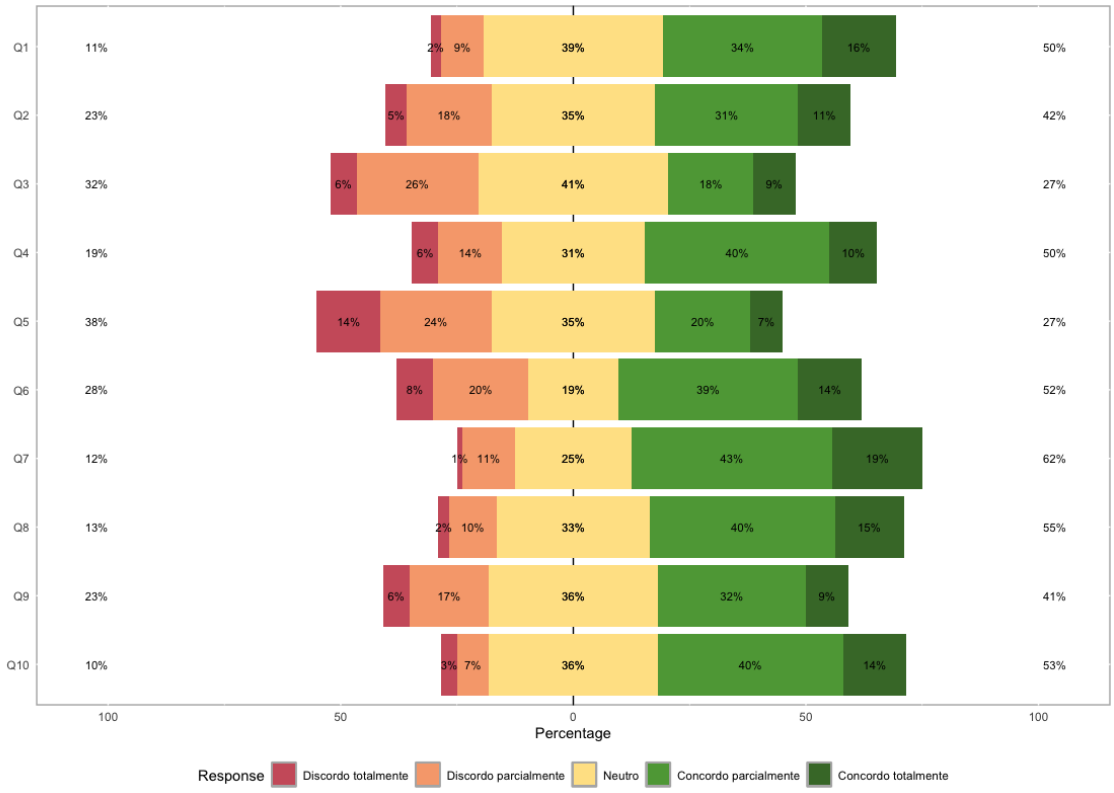
para que os participantes realizassem a elaboração correta de códigos relacionados ao conteúdo da disciplina, com 72% de concordância. Essa informação é corroborada pelo comentário do participante P84: *"O CoderBot melhorou a minha aprendizagem, visto que me ajudou rapidamente na resolução de problemas que eu tive dificuldade em resolver."*

6.3.3 Resultados com relação às Percepção Emocional

6.3.3.1 Percepção das Atitudes dos Estudantes

Ao analisar a Figura 22, que apresenta as atitudes dos estudantes em relação ao CoderBot, observa-se que metade dos participantes relataram sentir-se mais motivados a aprender com o apoio da ferramenta (**Q1**). Nesse sentido, o estudante P30 destacou: *"Ao apresentar exemplos, fica mais fácil a absorção do aprendizado, e, visualizando os exemplos, é mais fácil compreender o conteúdo"*. Além disso, 39% dos participantes mantiveram-se neutros em relação a essa afirmação.

Figura 22 – Percepção das Atitudes dos Estudantes



Fonte: De autoria própria.

Outro aspecto relevante é observado no item **Q7**, que avalia a utilidade do CoderBot durante a aprendizagem. Este item apresentou o maior índice de concordância (parcial e total) no gráfico, com 62% indicando uma percepção positiva quanto à eficiência da ferramenta. Neste sentido, o participante P56 afirmou: *"me ajudou a compreender os exercícios, porém de forma que ele não faça o código por mim, possibilitando-me pensar em como resolver de forma mais simples"*. Em contrapartida, 12% dos participantes discordaram da utilidade da ferramenta, enquanto 25% permaneceram neutros.

No item **Q10**, que avalia a experiência geral com o CoderBot, apenas 10% dos participantes discordaram da afirmação de que a ferramenta proporcionou uma boa experiência de uso. Por fim, o item **Q3**, que avaliou sobre a sensação de se sentir mais conectado com os outros ao adotar o Coderbot, apresentou a maior taxa de neutralidade. Este resultado reflete a ausência de funcionalidades colaborativas na versão atual da ferramenta, uma limitação que foi percebida pelos participantes, já que o CoderBot não oferece recursos para interação entre os estudantes.

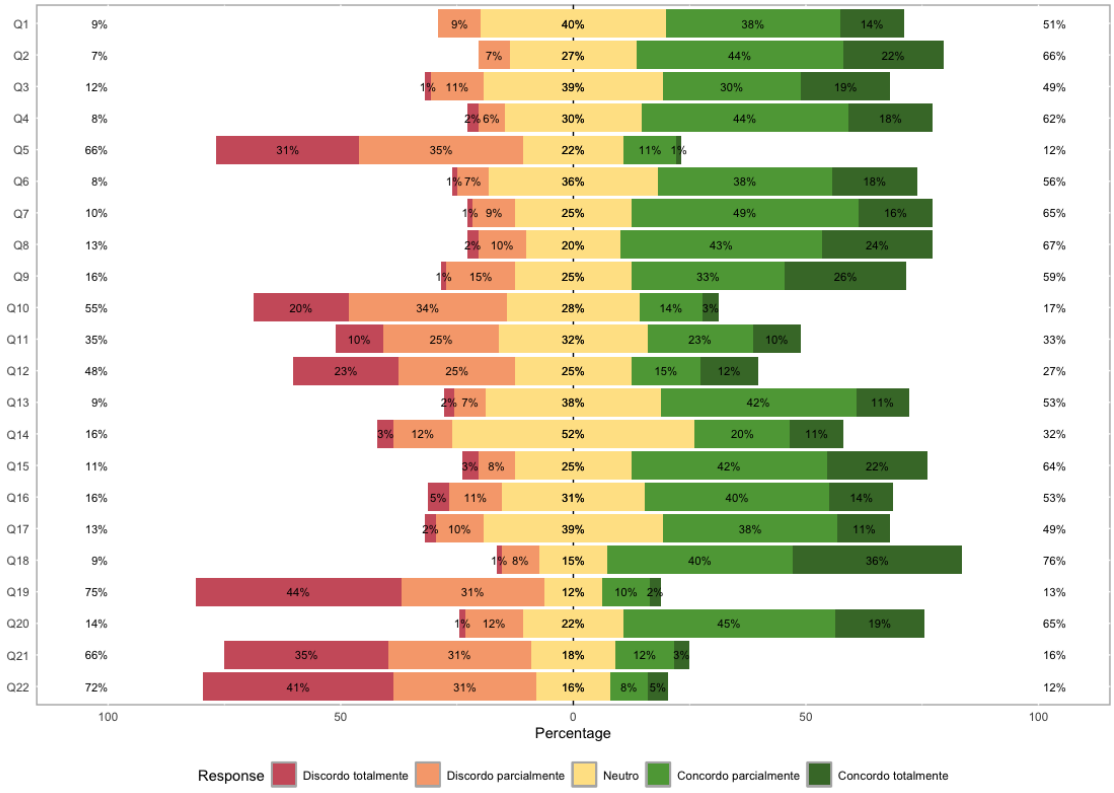
6.3.3.2 Percepção dos estudantes sobre Motivação Intrínseca

O instrumento de avaliação de *Motivação Intrínseca* (IMI) foi utilizado para mensurar a motivação intrínseca dos participantes durante o uso da ferramenta, considerando os contextos de aprendizagem e realização de tarefas. O IMI foi aplicado considerando os seguintes aspectos: **Interesse/Prazer**, **Escolha Percebida**, **Competência Percebida** e **Pressão/Tensão**, conforme pode ser observado na Figura 23.

Destacando o aspecto de **Interesse/Prazer**, o item **Q1** apresentou uma alta taxa de neutralidade (40%), mas com 51% de concordância, indicando que mais da metade dos alunos refletiram positivamente sobre o quanto gostaram de realizar as tarefas durante o uso do CoderBot. Já no item **Q2**, 66% dos participantes consideraram a realização da tarefa muito interessante, sendo este o item com menor índice de discordância e neutralidade.

Por outro lado, o item **Q5** apresentou 66% de discordância, dado que tratava de questionar se os participantes acharam as tarefas realizadas com o CoderBot muito chatas. Apenas 12% concordaram com a afirmação, enquanto 22% mantiveram-se neutros, indicando que a maioria dos alunos não considerou as tarefas monótonas. Esse aspecto de apoio à aprendizagem, reconhecendo o auxílio oferecido pelo CoderBot e a facilidade proporcionada, pode ser interpretado como uma vantagem no processo de desenvolvimento.

Figura 23 – Percepção dos Estudantes sobre Motivação Intrínseca



Fonte: De autoria própria.

No aspecto de **Escolha Percebida**, o item **Q8** revela que os participantes sentiram liberdade para realizar as tarefas usando o CoderBot, com uma taxa de concordância de 67%, enquanto 33% se mostraram neutros ou discordantes. Essa percepção é refletida na opinião do participante P25, que destacou: *"Além disso, ajuda a dar uma 'ideia' ou um 'norte' sobre o que fazer na questão. Dito isso, vejo apenas vantagens no uso do mesmo!"*, evidenciando que o CoderBot ofereceu liberdade e inspiração para a execução das tarefas. Já o item **Q11** apresentou uma alta divergência de opiniões: 35% dos participantes discordaram sobre a necessidade de realizar as tarefas utilizando o CoderBot, 33% concordaram, enquanto 25% se mantiveram neutros. Essa variação pode ser atribuída ao nível de experiência prévia de alguns estudantes. No item **Q12**, que aborda a falta de escolha para realizar as tarefas usando o CoderBot, 48% discordaram dessa afirmação, representando a maioria, enquanto 25% se mantiveram neutros e 27% concordaram. Esses resultados podem ser explicados pelo contexto do experimento, onde a participação era voluntária, permitindo que os estudantes optassem por explorar a ferramenta conforme sua perspectiva e necessidade.

No aspecto de **Competência Percebida**, os participantes demonstraram uma alta taxa de neutralidade (52%) ao avaliarem seu desempenho em relação aos outros ao realizar tarefas com o CoderBot (**Q14**). No entanto, a concordância (32%) superou a discordância (16%), indicando que, embora muitos tenham se mantido neutros, uma parcela significativa reconheceu sua própria competência. Essa neutralidade pode ser atribuída à ausência de um sistema de pontuação ou ranqueamento na ferramenta, o que dificulta a comparação de desempenho entre os participantes. No item **Q15**, observa-se um resultado positivo, com 64% dos participantes satisfeitos com seu desempenho nas tarefas realizadas com o CoderBot, enquanto 25% permaneceram neutros e apenas 11% discordaram. O impacto da ferramenta em melhorar o desempenho, especialmente ao relembrar conceitos previamente aprendidos. Em relação ao sentimento de competência após utilizar o CoderBot por um período para apoiar a realização das tarefas (**Q17**), 49% dos participantes concordaram que a ferramenta contribuiu para aumentar sua competência, enquanto 39% permaneceram neutros e apenas 13% discordaram. Essa percepção de melhora pode ser interpretada como o desenvolvimento de um *background* teórico, resultado da carga cognitiva proporcionada pela ferramenta.

Por fim, no aspecto de **Pressão/Tensão**, os resultados apontam para um ambiente de uso confortável proporcionado pelo CoderBot. No item **Q18**, 76% dos participantes afirmaram não ter sentido nervosismo ao realizar as tarefas com a ferramenta, enquanto 15% permaneceram neutros e apenas 9% indicaram nervosismo. Esse dado é reforçado pelo item **Q20**, no qual 65% dos participantes relataram sentir-se relaxados ao desenvolver as tarefas, com 22% neutros e apenas 13% discordando dessa percepção. Os itens **Q21** e **Q22** avaliaram sentimentos de ansiedade e pressão, respectivamente, ao realizar as tarefas com o apoio do CoderBot. A maioria dos participantes discordou da presença desses sentimentos, com 66% discordando de se sentirem ansiosos (**Q21**) e 72% discordando de se sentirem pressionados (**Q22**). Esses resultados indicam que o CoderBot proporcionou um ambiente tranquilo e de baixo estresse para os participantes.

Assim, é possível concluir que o uso do CoderBot contribuiu para minimizar sentimentos de ansiedade e pressão durante a realização das atividades, o que representa um aspecto positivo tanto para a aceitação quanto para a usabilidade da ferramenta.

6.3.4 Avaliação de Interface do CoderBot

6.3.4.1 Percepção dos Estudantes sobre a Escala de Usabilidade do Sistema

Conforme reportado anteriormente, a análise foi realizada considerando o questionário SUS, escolhido por ser uma metodologia rápida e eficaz para a avaliação de sistemas de diferentes tipos e complexidades. Segundo Sauro (2011), para que um sistema seja considerado de boa usabilidade, o *score* mínimo esperado é de 68 pontos. A aplicação da escala SUS ao sistema em questão resultou em um *score* de 79,2 pontos, classificando-o como de média usabilidade. Além disso, foi calculado o desvio padrão da amostra, que revelou uma variação de 11,7 pontos em relação à média, indicando certa consistência nas respostas dos participantes.

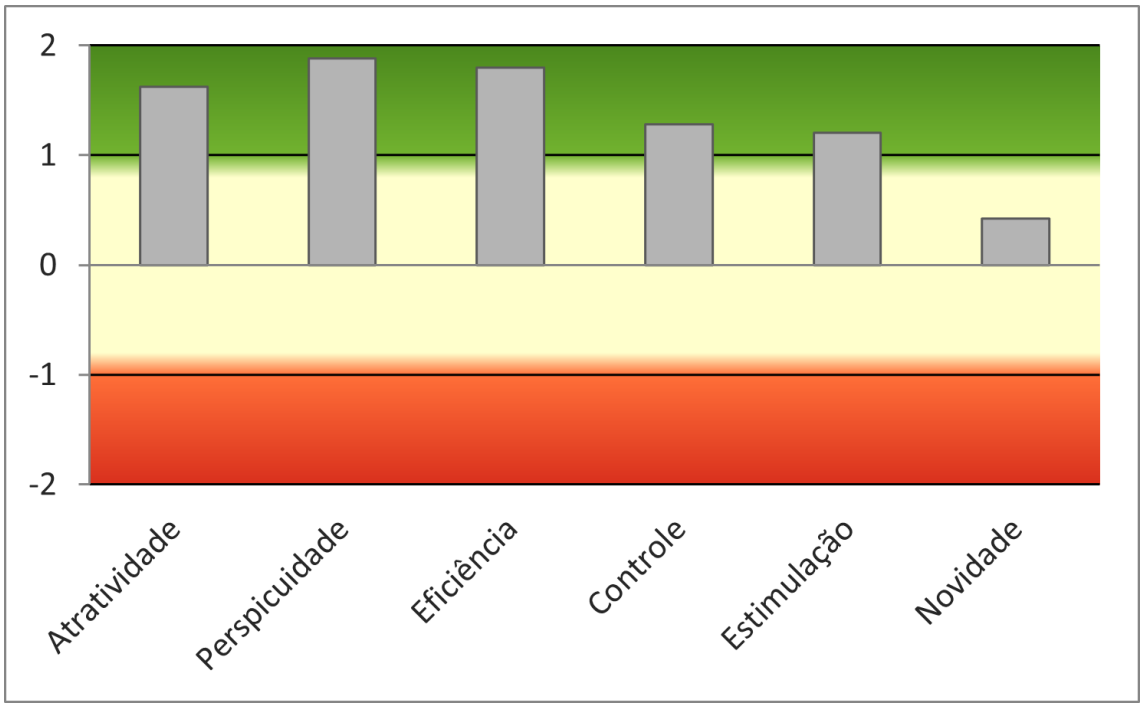
6.3.4.2 Percepção dos Estudantes sobre Experiência do Usuário

A avaliação das respostas coletadas no Questionário UEQ foi realizada com o objetivo de compreender aspectos da experiência do usuário durante a utilização do CoderBot. Os itens abordados pelo UEQ englobam aspectos de **Atratividade**, **Perspicuidade**, **Eficiência**, **Controle**, **Estimulação** e **Novidade**. Os dados foram tabulados e inseridos na ferramenta "UEQ Data Analysis Tool Version 12", para análise dos dados, os quais foram obtidos no site oficial do UEQ <https://www.ueq-online.org/>.

A partir do uso da ferramenta é possível gerar diversas informações, tendo em vista que a mesma já realiza a análise gerando o gráfico que pode ser observado na Figura 24.

Os dados indicados na Figura 24 indicam que o CoderBot possui um "bom" grau de **Atratividade**, ou seja, desperta ao usuário o desejo de utilização. Já em **Perspicuidade**, a classificação foi de "bom" a "excelente", significando a facilidade de aprendizagem e uso do produto. Em **Eficiência** obteve o grau de "bom" a "excelente", indicando que o mesmo é útil e cumpre seu objetivo oferecendo uma experiência de apoio à aprendizagem de programação. A categoria de **Controle** e **Previsibilidade** indicou uma classificação de "acima da média" indicando que o usuário se sentiu no controle da ferramenta na maioria dos usos. **Estimulação** também foi classificada como "acima da média" indicando que para a maioria dos usuários que o uso do CoderBot é estimulante e divertido. Finalizando a análise, tem-se o primeiro aspecto que obteve uma classificação "abaixo da média", no caso, o item de **Novidade**, onde apresenta o grau de inovação, tal índice

Figura 24 – Resultados da aplicação do Questionário da Percepção dos Estudantes sobre Experiência do Usuário



Fonte: De autoria própria.

alcançado pode ser explicado devido ao surgimento de *chatbots* baseados em IA generativas, onde o usuário obtém soluções a partir de uma requisição detalhada, o que diverge do foco do CoderBot quanto ferramenta de apoio à aprendizagem de programação, como foi previamente relatado na Subseção 5.3.

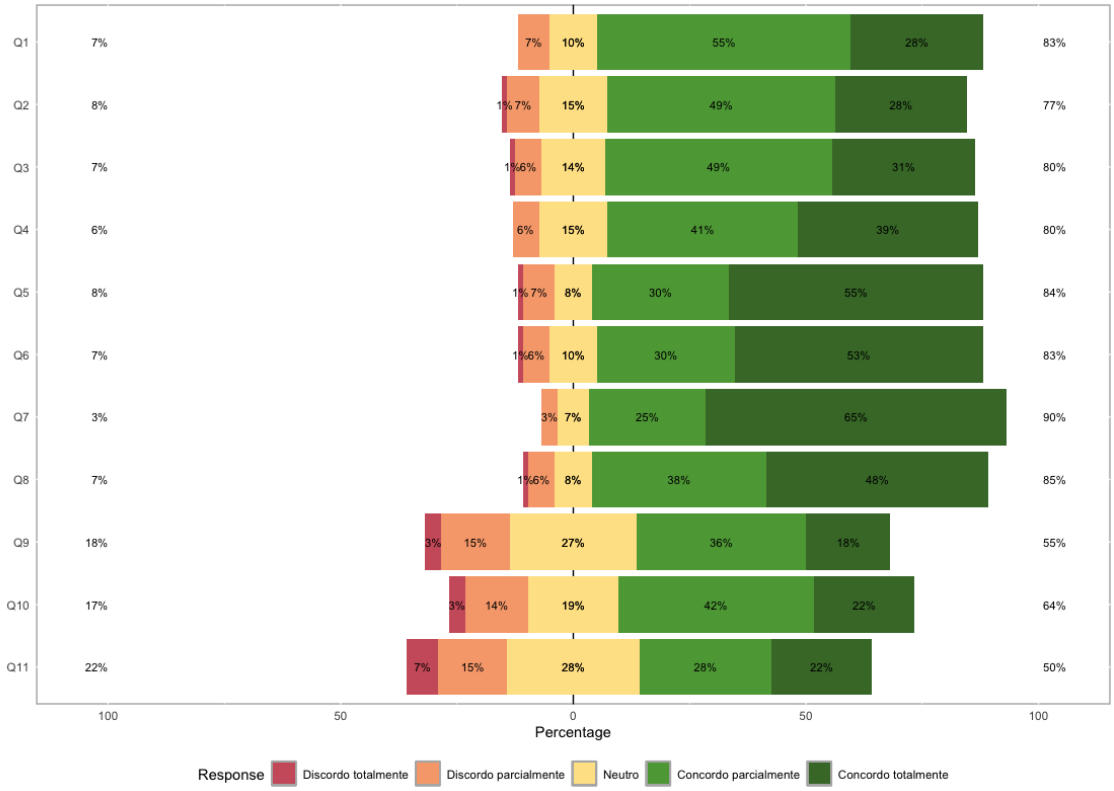
6.3.5 Aceitação da Tecnologia

6.3.5.1 Questionário TAM

A Figura 25 apresenta os resultados das percepções dos estudantes avaliando o CoderBot utilizando o TAM. Os resultados relativos à **Utilidade Percebida**, evidenciou-se que 83% dos estudantes concordaram que o CoderBot melhora o desempenho na resolução de problemas de programação (UP1). O estudante Além disso, 80% dos estudantes concordaram que o *CoderBot* pode ser útil no aprendizado de programação (UP4). Sobre isso, E83 comentou o seguinte: "*ajuda no aprendizado de programação, por trazer exemplos práticos que são vistos durante as aulas, fazendo com que o estudante entenda*

sobre a lógica de programação proposta pelos exercícios" – E83. Esses comentários indicam que o CoderBot foi bem recebido pelos estudantes como uma ferramenta útil para o ensino de programação, reforçando sua eficácia como suporte educacional.

Figura 25 – Aceitação dos estudantes.



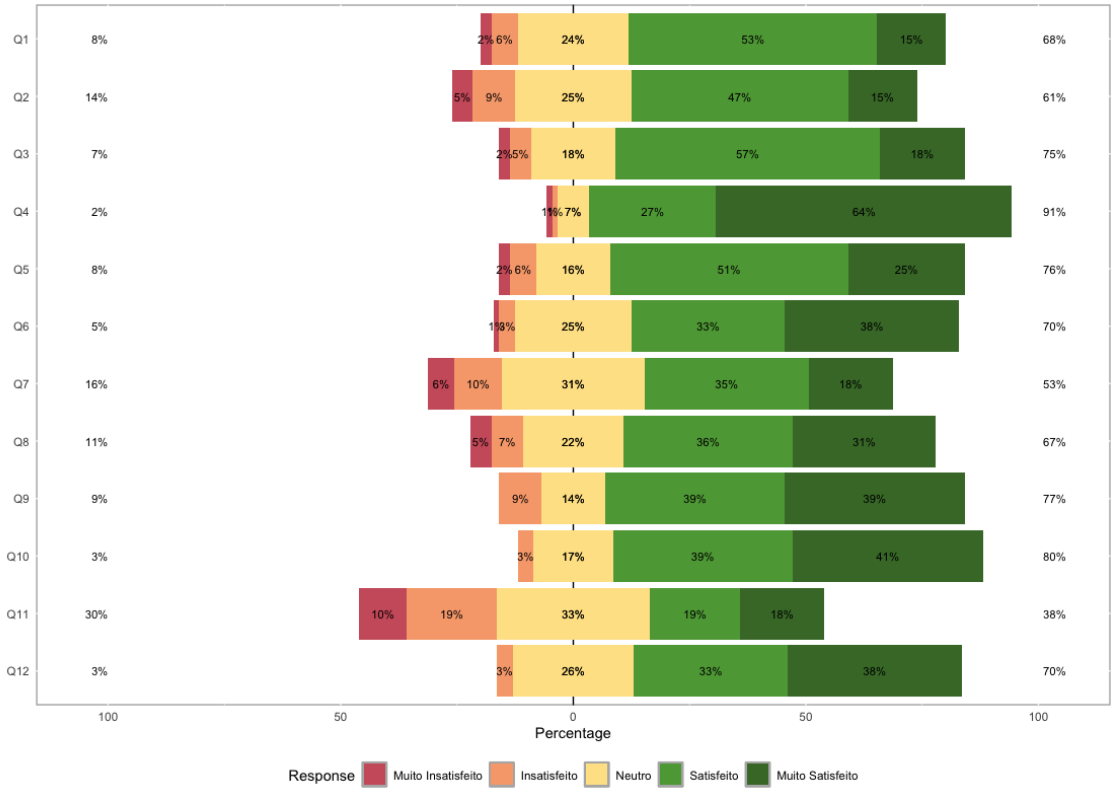
Fonte: De autoria própria.

A análise da **Facilidade de uso** evidenciou uma concordância acima de 85% neste indicador. No item FUP3, E17 afirmou: "*o uso do Coderbot foi fácil, ajudando a resolver exercícios simples e, em alguns momentos, exercícios de dificuldade média*". Portanto, a inclusão do CoderBot no ensino não apresenta maiores dificuldades para os estudante. Apesar de mais de 56% dos estudantes terem uma percepção positiva sobre a Intenção de Uso Percebida, uma quantidade razoável de estudantes permaneceu neutra (média de 24,7%) e discordando (média de 19%). E33 sugeriu: "*para o intuito de aprendizado seria importante adicionar comentários nos códigos para auxiliar no entendimento do mesmo.*" Esses dados indicam que, embora haja uma aceitação geral, há espaço para melhorias na funcionalidade e abrangência do CoderBot.

6.3.5.2 Questionário Autoeficácia

A Figura 26 apresenta os resultados da Autoeficácia percebidas pelos os estudantes. No indicador de **Satisfação**, observou-se que a maioria dos estudantes (média de 68% dos estudantes) ficou satisfeita com o CoderBot. Por exemplo, E81 reforçou que ficou satisfeito, pois o CoderBot "*ajudou em dar um norte para cada questão, o que economiza tempo, isso é ótimo*". Além disso, 68% dos estudantes concordaram que gostaram do CoderBot. O estudante E18 comentou "*mesmo que o exercício não peça exatamente a função que tem no CoderBot, é fácil de utilizar como base para adaptar para o problema real da questão.*" No que diz respeito à **Usabilidade**, este obteve uma média de concordância de 79%, a maior entre os indicadores de autoeficácia. O item U1 obteve concordância de 91% dos estudantes, demonstrando a facilidade de uso do CoderBot.

Figura 26 – Autoeficácia percebida pelos estudantes.



Fonte: De autoria própria.

No indicador de **Benevolência**, a concordância positiva média foi de 65%. O item B3, que avaliou se a interação com o CoderBot foi agradável, foi o que obteve os melhores resultados (77% de concordância). Por outro lado, o item B1, que avaliou se as dúvidas foram corretamente compreendidas, mostrou uma quantidade relevante de participantes

permanecendo neutros (31% dos respondentes).

Por fim, no indicador **Credibilidade**, este foi o que apresentou menor taxa de concordância (média de 62,5%). O item C1, que questionava se o CoderBot deveria ser integrado às práticas de ensino de programação, teve respostas majoritariamente positivas, com 80% de concordância. O estudante E18 ressaltou que o CoderBot "*colaborou na resolução de alguns exercícios pela simplicidade na explicação e na estrutura dos exemplos.*" No entanto, o item C2 mostrou uma neutralidade significativa (33%). O estudante E48 afirmou que o CoderBot é "*muito compreensível e prático, porém, não explica o que iniciantes precisam saber, por isso discordo em utilizar somente ele para os conteúdos dados em aulas.*" Portanto, melhorias na explicação da lógica dos códigos seriam úteis.

6.3.6 Resultados Qualitativos

Após uma análise qualitativa dos resultados, foi possível identificar quatro principais categorias que abordam variados aspectos das visões dos envolvidos.

A primeira categoria que emergiu foi: **CODERBOT AJUDOU A MELHORAR A APRENDIZAGEM**. Esta categoria trata da assistência proporcionada pelo CoderBot, abordando como ele auxiliou no aprendizado e em quais aspectos específicos ele foi útil.

A subcategoria CoderBot ajudou a compreender a linguagem ensinada abrange comentários que declararam como o CoderBot auxiliou na aprendizagem da linguagem de programação utilizada nos exercícios. Por exemplo, E01 mencionou: *"o CoderBot me ajudou a realizar os exercícios passados e me ajudou a compreender melhor a linguagem C"*. Outro estudante complementou: *"tenho familiaridade com outra linguagem de programação, que não é C, e o Coderbot me ajudou a ter um início rápido com a sintaxe e a estrutura de um programa em C"* – E31. Os estudantes ressaltaram ainda que o CoderBot facilitou no entendimento dos exercícios, com um deles observando: *"ele facilitou na resolução dos exercícios, pois, ele otimiza o pensamento, através de um código bem escrito, fica fácil de entender qual a lógica para aquela questão"* – E08. E83 destacou que o CoderBot auxiliou na compreensão da lógica de cada exercício: *"ele ajuda no aprendizado de programação, por trazer exemplos práticos que são vistos durante as aulas de Introdução à Programação e Estruturas de Dados, etc., fazendo com que o aluno entenda sobre a lógica de programação que cada um dos exercícios propõem."*

A subcategoria CoderBot ajudou na resolução de problemas inclui citações sobre como CoderBot auxiliou na resolução de problemas e no desenvolvimento dos exercícios.

E16 relatou: *"o uso do CoderBot dá algumas luzes para a resolução de problemas"*), enquanto E14 complementou: *"o CoderBot me auxiliou na resolução dos problemas, mesmo não sendo o problema exato que eu estava tendo, mas a ferramenta me mostrou passo-a-passo para resolver o problema maior"*. Além disso, foi percebido que o CoderBot ajudou os iniciantes em programação, com estudantes destacando o apoio fornecido para quem está começando a aprender programação. O estudante E67 mencionou: *"CoderBot é uma ferramenta útil para quem está iniciando a programação"*, e outro estudante adicionou: *"serviu para auxiliar em questões básicas e para quem está começando uma linguagem é bastante interessante"* – E60.

Outra subcategoria evidenciada foi que o CoderBot provê uma explicação didática e eficaz dos exercícios, com estudantes ressaltando a clareza e a eficácia das explicações fornecidas pelo CoderBot. E57 comentou que o CoderBot disponibiliza *"um conteúdo claro e direto facilitando a leitura. Os tópicos do passo a passo da execução do código me ajudaram a entender como funciona cada linha de comando"*. E73 ainda destacou: *"foi muito mais claro em visualizar o código e em questão de aprendizado me ajudou muito, com o passo a passo fica mais fácil de compreender cada linha"*. Isso evidencia que a explicação clara e direta, com instruções passo-a-passo, pode ter facilitado a aprendizagem e compreensão dos estudantes. Por fim, a subcategoria que aborda como o CoderBot ajudou a relembrar conteúdos previamente aprendidos também foi identificada. E82 mencionou o seguinte: *"o uso do CoderBot me ajudou a melhorar, consegui me lembrar de como se utiliza o Array (aprendi no semestre retrasado), confesso que me esclareceu muitas coisas"*. E15 complementou dizendo: *"ele foi bom para algumas questões, por exemplo, quando eu precisei lembrar de alguns algoritmos que eu já tinha feito na lista, como as do somatório"*.

A segunda categoria emergente foi: **EXEMPLOS PRESENTES NO CODERBOT AJUDARAM NO ENTENDIMENTO**. Esta categoria abrange comentários dos estudantes que relataram como os exemplos fornecidos pelo CoderBot facilitaram a compreensão dos exercícios e de maneira isso ocorreu.

Uma subcategoria importante que se destacou foi CoderBot apresenta exemplos corretos e incorretos. Com base nos resultados, percebeu-se que essa abordagem melhora a compreensão dos estudantes sobre a resolução de problemas. Por exemplo, o estudante E49 afirmou que o CoderBot apresenta *"uma maneira correta e errada de fazer o exercício, assim, o aluno também aprende qual caminho não trilhar no desenvolvimento do código"*. Esta perspectiva enfatiza a importância de mostrar ambos os

exemplos, corretos e incorretos, durante a explicação de um exercício, ajudando os alunos a entenderem claramente as duas formas de abordagem, a correta e a incorreta. Outra subcategoria identificada foi CoderBot apresenta a resolução do exercício detalhadamente. Os estudantes destacaram que a resolução detalhada dos exercícios pelo CoderBot os ajudou a entender melhor como resolver as questões. O estudante E57 mencionou que: *"os tópicos do passo a passo da execução do código me ajudaram a entender como funciona cada linha de comando."* Similarmente, E73 observou: *"com o passo a passo fica mais fácil de compreender cada linha, foi muito mais claro em visualizar o código e em questão de aprendizado me ajudou muito."*

Além disso, foi notado que CoderBot apresenta uma simplicidade na explicação dos exemplos de códigos, especificamente na forma como as resoluções dos exercícios foram exibidas. Isso é exemplificado pelo comentário de E18, que disse ter conseguido resolver os códigos dos exercícios devido à explicação clara e simplificada fornecida pelo CoderBot: *"acredito que o CoderBot colaborou na resolução de alguns exercícios pela simplicidade na explicação e na estrutura dos exemplos."* Finalmente, houve relatos de que os exemplos auxiliaram a iniciar o código, indicando que os exemplos ajudaram os estudantes a darem o primeiro passo na resolução dos exercícios. Por exemplo, E76 mencionou que *"o CoderBot foi útil em auxiliar o início do código dando uma breve ideia de como o código poderia ser."* O estudante E66 disse que *"a gama de dados para exemplos em demonstração é pequena, porém mostrou o caminho inicial para resolução das questões."*

Os códigos relacionados a esta categoria indicam que, embora muitos estudantes tenham tido uma boa percepção do CoderBot, alguns encontraram dificuldades significativas em sua utilização. A categoria emergente foi: **CODERBOT NÃO AJUDOU A MELHORAR A APRENDIZAGEM**, abordando os motivos pelos quais a ferramenta não foi útil para determinados estudantes.

Uma das dificuldades percebidas foi que é necessário realizar muita leitura a partir do CoderBot. O estudante E02 expressou que *"na minha impressão inicial, a ferramenta tem muita leitura e pouca digitação"* e também destacou que poderia utilizar outras ferramentes disponíveis para realizar a mesma função, considerando o CoderBot limitado. Esta percepção sugere que o excesso de leitura pode ser um obstáculo para estudantes que preferem uma abordagem mais prática. Outra dificuldade percebida foi que o CoderBot requer conhecimento prévio para ser efetivamente utilizado. Nesse sentido, o estudante E37 comentou: *"se quem o utilizar não tiver o mínimo de conhecimento em*

programar não ajudaria muito na resolução do problema.“ E52 complementou dizendo: *“ele é bem intuitivo e claro para quem já conhece um pouco sobre programação, como ainda estou no início, tive dificuldade em entender a interseção, lista, ponteiros e algumas características mais específicas.”* Esses relatos evidenciam que estudantes em estágio inicial de aprendizagem podem enfrentar dificuldades ao desenvolver os exercícios propostos com o CoderBot.

Os estudantes também relataram que o CoderBot complicou a resolução da atividade em vez de ajudar, deixando-os com mais dúvidas. E05 declarou: *“em outras questões do exercício, ele parcialmente acabou complicando o processo desenvolvimento da questão.”* P75 também corroborou, dizendo: *“às vezes, ele deixou ainda mais confuso as lógicas mais simples.”*. Esses comentários indicam que, para alguns, o CoderBot introduziu mais complexidade do que clareza. Uma quarta dificuldade mencionada foi a falta de clareza nos exemplos do CoderBot. Este problema foi destacado por comentários que mencionam a falta de clareza nos exemplos exibidos, dificultando o desenvolvimento dos exercícios. Nesse quesito, E38 destacou: *“não ajudou muito porque quando eu não entendia a lógica eu precisava de uma explicação”*, sugerindo que apenas o conteúdo apresentado pelo CoderBot não era suficiente para resolver as questões. Somando a esse cenário, E88 comentou: *“o Coderbot apresenta códigos prontos com poucos comentários, deveria apresentar de maneira mais clara a lógica por trás das questões, já que a lógica de programação é mais importante que a codificação em si.”* Esses relatos indicam a necessidade de exemplos mais detalhados e explicativos.

A última categoria emergente, **SUGESTÕES DE MELHORIAS PARA O CODERBOT**, explora os comentários dos estudantes que propuseram aprimoramentos e ajustes para tornar o CoderBot ainda mais eficaz.

Um dos principais pontos ressaltados foi a falta de mais exemplos práticos para auxiliar no aprendizado e no desenvolvimento dos exercícios. O estudante E29 destacou que o CoderBot deveria exibir mais exemplos de programação com possíveis soluções: *“acho que deveria ter mais exemplos, mostrar mais formas de como se pode fazer uma mesma questão, não me recordo qual, mas alguma das questões ele usou boolean sendo que existe outras maneiras que não precisa usar, então acho que seria cabível ter pelo menos uns 3 exemplos de cada coisa (caso seja um pouco mais complexo).”* E77 complementou, dizendo: *“a disponibilidade de exemplos oferecida é baixa em problemas mais diversos; o Coderbot sairá para trás em relação a outros chatbots.”* Esses comentários sugerem que uma variedade maior de exemplos poderia

melhorar a compreensão e a versatilidade da ferramenta. Outro ponto mencionado foi a limitação das funcionalidades limitadas do CoderBot. E01 disse: *"por mais que ele tenha sido de muita ajuda, o seu estado atual é consideravelmente limitado, principalmente quanto a dúvidas mais complexas."* E41 complementou dizendo que a ferramenta foi útil, porém ainda é limitada: *"serviu para ajudar em questões básicas, mas ainda é um sistema limitado."* E46 sugeriu o adicionar um botão de copiar e colar o código de exemplo exibido no CoderBot, para poupar tempo dos estudantes: *"inserir um botão de copiar o código seria ótimo, para poupar tempo e apenas editar o que precisa."* Esses feedbacks indicam a necessidade de funcionalidades adicionais e maior flexibilidade no uso do CoderBot.

Os estudantes comentaram sobre a falta de comentários nos códigos. E88 observou que *"o Coderbot apresenta códigos prontos com poucos comentários, deveria apresentar de maneira mais clara a lógica por trás das questões, já que a lógica de programação é mais importante que a codificação em si."* Comentários detalhados poderiam fornecer um melhor entendimento dos exemplos de códigos apresentados, beneficiando especialmente os iniciantes. Outro aspecto discutido foi a adição de novas funcionalidades. P85 sugeriu melhorias na interface do chat e a opção de alterar a linguagem usada nas respostas: *"acredito que a interface do chat poderia ser um pouco melhor e uma opção para alterar a linguagem usada na resposta também seria legal."* E72 destacou a necessidade de descrições dos comandos utilizados nos exemplos: *"alguns comandos não são explicados para que servem... seria legal se tivesse a descrição dos comandos... se é para iniciantes, então eles deveriam pelo menos saber a descrição de cada comando."* Essas sugestões indicam que melhorias na interface e nas funcionalidades do CoderBot poderiam torná-lo mais acessível e útil para todos os níveis de usuários.

6.4 Discussão dos Resultados

Os resultados obtidos no experimento suscitam discussões relevantes sobre o processo de ensino-aprendizagem de programação. Entre elas, destaca-se, talvez, a mais importante: a relevância do uso de metodologias ativas combinadas com tecnologias emergentes para promover uma aprendizagem mais instigante e engajada.

Os resultados do Questionário aplicado para avaliar a Percepção das Atitudes dos estudantes evidenciou que mais da metade dos alunos se sentiram mais motivados e engajados à aprender programação utilizando o CoderBot. Comentários como o do par-

ticipante P30, que afirma: “Ajudou, ao apresentar exemplos fica mais fácil a absorção do aprendizado e visualizando os exemplos é de mais fácil atendimento.” indicaram que o CoderBot facilitou a absorção de aprendizado por meio dos exemplos, o que corrobora com a ideia de engajamento. É possível perceber o estímulo ao pensamento crítico, com comentário de P8 que afirma a facilidade de entender após analisar um código bem escrito: “Sinceramente, ele facilitou a resolução dos exercícios, pois, ele otimiza o pensamento, através de um código bem escrito, fica fácil de entender qual a lógica para aquela questão...” Outro ponto indicando a participação ativa na aprendizagem é definida pelo comentário de P56 que indica: “me ajudou a compreender os exercícios, porém de forma que ele não faça o código por mim, possibilitando-me pensar em como resolver de forma mais simples” sobre compreender os exercícios sem que o CoderBot faça o código para eles, ou seja, apresentando exemplos que possuem códigos com lógica semelhante. Mesmo com todos esses aspectos positivos, é possível perceber, pela neutralidade na item 3 (“Eu me senti mais conectado com os outros ao adotar o Coderbot”), que os participantes sentiram falta de mais interatividade com a ferramenta e apontaram a necessidade de melhorias na interface para oferecer esse suporte, como é o caso de uma área de escrita e teste de código na própria ferramenta, como foi relatado.

Os dados e opiniões coletados pelo Questionário para avaliar a Percepção dos Estudantes sobre a Experiência de Fluxo abordaram as sensações psicológicas dos participantes no uso da ferramenta. Inicialmente, os participantes indicaram a necessidade de funções adicionais na ferramenta, como a inclusão de um botão para copiar códigos, o que sugere que a experiência seria mais espontânea e automática com a implementação de novos recursos.

Entretanto, as respostas à questão 2 (“Com o CoderBot, fiz as coisas de forma espontânea e automática, sem ter que pensar”) evidenciam uma sensação de competência e suficiência na realização das atividades com o CoderBot. Isso aponta que, mesmo diante da demanda por melhorias, a ferramenta demonstrou eficiência no seu contexto, proporcionando uma interface simples e intuitiva.

Em relação à imersão, ao foco e à percepção do tempo, os resultados são divergentes, indicando a ausência de um consenso. Embora 50% dos participantes tenham concordado positivamente, outros 30% se mantiveram neutros, e 10% discordaram. Essa dispersão pode ser justificada pela ausência de elementos de gamificação, que poderiam aumentar o engajamento do usuário com a ferramenta, mantendo-o mais focado na sua utilização do que na atividade em si.

Ressalta-se que o foco principal do CoderBot é apoiar o estudante na aprendizagem de programação, auxiliando na compreensão e execução de atividades em cursos introdutórios de programação.

O Questionário de Avaliação sobre a Aprendizagem Percebida trouxe indicadores positivos em termos de facilitação da compreensão (68% em Q1), interpretação (66% em Q4) e aplicação dos conteúdos aprendidos (67% em Q5), além de auxiliar na elaboração correta de código (72% em Q8). Comentários como o de P48, que afirma: "Com o uso do CoderBot consegui entender trechos de código. Achei interessante o método de utilização dele", sobre a compreensão de trechos de código, e o de P84, que declara: "O CoderBot melhorou a minha aprendizagem, visto que me ajudou rapidamente na resolução de problemas que eu tive dificuldade em resolver", sobre a ajuda na resolução de problemas, corroboram esses dados.

Os resultados destacam a eficácia do CoderBot como uma ferramenta educacional que facilita a aprendizagem de exemplos (códigos) de programação de forma clara e intuitiva.

Entretanto, mesmo com esses resultados relevantes, ainda é possível identificar pontos de melhoria, como em Q2, que indicou níveis significativos de neutralidade (44%) e discordância (22%). Esses dados sugerem que não há uma percepção generalizada de que o CoderBot seja mais eficiente em relação a outras metodologias de ensino. Essa percepção pode oferecer *insights* valiosos para a avaliação de outras abordagens metodológicas, com o objetivo de aprimorar a ferramenta e consolidá-la ainda mais no contexto da educação em computação, melhorando a experiência de aprendizagem e a eficácia pedagógica.

A aplicação do Questionário sobre a Perceção do Estudante sobre Motivação Intrínseca (IMI) sobre o uso do CoderBot revelou dados importantes acerca da motivação dos participantes no contexto da aprendizagem e realização de tarefas. O aspecto de Interesse/Prazer obteve bons resultados, com 66% dos participantes considerando interessante a realização das tarefas com o CoderBot, enquanto apenas 12% relataram que as tarefas eram chatas.

No quesito Escolha Percebida, a discordância quanto à necessidade do CoderBot para a realização das tarefas pode ser justificada pela diferença nos níveis de conhecimento dos estudantes, mesmo em disciplinas introdutórias de programação. Um exemplo disso é o relato de P10: "Resolvi as 3 primeiras questões com certa facilidade. Na 4ª, tive dificuldade de proceder por não ter aprendido a utilizar a função 'sizeof' da maneira

que foi empregada. No entanto, a lógica de comparar os valores dos índices um a um utilizando dois loops já era algo que eu havia pensado. Assim, o CoderBot acabou me passando uma sensação de redundância, e o conhecimento que me foi transmitido se limitou à sintaxe, algo que é facilmente encontrado na internet. Isso faz com que o CoderBot seja inferior a outras IAs disponíveis na web." Esse depoimento destaca que a limitação ao fornecimento de sintaxes, facilmente acessíveis online, pode reduzir os benefícios da ferramenta em relação a outras disponíveis na internet.

Quando os participantes foram questionados sobre a avaliação de seu próprio desempenho, no aspecto Competência Percebida, os dados indicaram alta neutralidade. Isso pode ser explicado pela ausência de uma funcionalidade que correlacione diretamente as tarefas realizadas com uma pontuação ou avaliação integrada na ferramenta, o que permitiria aos usuários medir seus acertos. Esse ponto revela um requisito importante para a evolução da ferramenta, pois orientaria o usuário a monitorar e refletir sobre seu próprio desempenho.

Por fim, no quesito Pressão e Tensão, os resultados foram positivos, com mais de 76% dos participantes indicando que não se sentiram nervosos ao realizar as tarefas com o CoderBot. Em itens relacionados a sentimentos de ansiedade e pressão, os índices também foram favoráveis, demonstrando que a maioria dos participantes não experimentou essas sensações negativas.

Essas informações refletem bons resultados e sugerem que o CoderBot tem o potencial de impactar positivamente a motivação dos estudantes, contribuindo para o apoio à aprendizagem de forma eficiente e acolhedora.

Os resultados trazidos da coleta de dados dos questionários que avaliaram as percepções dos estudantes em relação à interface do CoderBot trouxeram resultados positivos na maior parte dos aspectos, tendo como um ponto abaixo da média apenas em quesito de inovação, os participantes compararam a tecnologia às IAs generativas que estão com alta popularidade, vide ChatGPT, Copilot ou Gemini. Entretanto quando o assunto se foca em abordagem metodológica é perceptível a diferença e objetivo de cada tecnologia.

O CoderBot ainda possui espaço para expansão e inclusão de aspectos inovadores e até mesmo a possibilidade de inclusão de IAs generativas para geração de elementos, como por exemplo, os exemplos de programação para consumo da ferramenta.

Com base nos pontos abordados na discussão, pode-se concluir que o CoderBot é uma ferramenta com grande potencial para apoiar o aprendizado de programação. Considerando que muitos estudantes iniciantes enfrentam dificuldades que os levam a desistir

ou reprovar, o CoderBot pode oferecer suporte acessível e de alta disponibilidade, ajudando a reduzir essas barreiras ao aprendizado.

Vale ressaltar que o CoderBot, encontra-se com sua primeira versão finalizada, cujo qual passou por experimentação e por avaliação. Conforme a avaliação, é possível identificar novos requisitos, por meio dos comentários analisados, entretanto também é observável que o CoderBot proporcionou aos participantes boas sensações e confiabilidade durante sua utilização no apoio à aprendizagem.

No geral, percebeu-se por meio dos resultados que é justificado o uso de um assistente virtual com abordagem baseada em exemplos para apoiar estudantes de graduação iniciantes no aprendizado de programação.

Se a dificuldade em aprender programação é um dos motivos que levam os novatos a abandonarem cursos introdutórios, o apoio extra no início do processo, como o fornecido pelo CoderBot, pode ajudar a reduzir muitas das frustrações enfrentadas pelos estudantes.

6.5 Considerações sobre o Capítulo

Este capítulo teve como objetivo apresentar o planejamento da avaliação, coleta e resultado da análise de dados realizados na experimentação executada utilizando o CoderBot. Ele foi desenvolvido para apoiar os estudantes do ensino superior durante a aprendizagem de programação em etapas iniciais, aplicando a metodologia de *worked examples* com as vertentes de exemplos corretos e incorretos. A aplicação do experimento foi utilizada para verificar o comportamento dos estudantes durante o uso da ferramenta na resolução de atividades. Todos os questionários foram aplicados visando identificar os fatores na utilização da ferramenta, as possíveis melhorias, os sentimentos expressos pelos participantes e, por fim, aspectos de percepção de aprendizagem.

Diante da análise e interpretação dos dados é possível identificar que a ferramenta foi bem aceita e trouxe benefícios durante sua utilização, apoiando o estudante na compreensão de exemplos corretos e incorretos de código. Também vale ressaltar que a aplicação do experimento e questionários permitiu identificar aspectos a serem melhorados, como aumentar a quantidade de exemplos e/ou formas de resolução do mesmo exemplo, além de indicar novos requisitos, como teste e execução dentro do próprio CoderBot.

7 CONSIDERAÇÕES FINAIS

7.1 Conclusão

Este trabalho de mestrado apresenta uma pesquisa sobre o uso de um assistente virtual no apoio à aprendizagem de programação por meio da aprendizagem baseada em exemplos. Com o objetivo de alcançar essa proposta, foi construído e avaliado um assistente virtual que, a partir de interações via cliques em botões, fornece ao aluno exemplos baseados em *templates* previamente definidos. Esses exemplos têm a finalidade de demonstrar tanto casos corretos quanto incorretos, com o intuito de apoiar o aprendizado de programação.

A ferramenta é capaz de indicar os passos necessários para a construção de uma solução correta, ou de fornecer códigos incorretos, estimulando o aluno a analisar e identificar os problemas existentes no código. Além disso, a ferramenta questiona o aluno sobre o local do erro no código e fornece um *feedback* para que ele compreenda e aprenda com o erro.

Foi realizado um Mapeamento Sistemático da Literatura, com base nas diretrizes propostas por Petersen, Vakkalanka and Kuzniarz (2015), para identificar estudos que reportam o uso de *chatbots* no apoio ao ensino de programação. Nesse MSL, observou-se que poucos estudos identificados apresentavam um direcionamento semelhante ao proposto neste trabalho. No entanto, os estudos encontrados utilizavam *chatbots* como referência para conceitos, abordagens em formato de perguntas e respostas, suporte emocional durante a aprendizagem e até mesmo a introdução de conceitos mais avançados.

De acordo com o resultado do Mapeamento Sistemático da Literatura, foi possível constatar a ausência de padrões para a construção dos exemplos a serem apresentados aos estudantes. Diante disso, realizou-se um estudo exploratório que, com o apoio de docentes experientes na área de programação, resultou na definição e avaliação de um *template* para a construção dos *worked examples* a serem utilizados pela ferramenta.

Nesse sentido, o **CoderBot** foi desenvolvido e disponibilizado por meio de um sistema *Web* capaz de fornecer exemplos de programação baseados no *template* de *worked example* proposto nesse trabalho, os quais foram abordados na forma correta e incorreta. A ferramenta passou por testes pelos desenvolvedores, e teve a realização de experimento com estudantes de programação em etapas iniciais de cursos de graduação. O resultado dos dados coletados no experimento, indicaram o grande potencial deste tipo de ferra-

menta no apoio à aprendizagem de programação, mesmo diante do surgimento de IAs generativas, e as observações que indicam a possibilidade de inclusão de novos requisitos, a forma de abordagem do CoderBot o consolida como uma ferramenta educacional de apoio.

Nesse contexto, o CoderBot foi desenvolvido e disponibilizado por meio de um sistema web capaz de fornecer exemplos de programação baseados no *template* de *worked example* proposto neste trabalho. Esses exemplos foram apresentados nas formas correta e incorreta. A ferramenta foi submetida a testes realizados pelos desenvolvedores e também a um experimento com estudantes de programação nas etapas iniciais de cursos de graduação.

Os resultados dos dados coletados no experimento indicaram o grande potencial dessa ferramenta no apoio à aprendizagem de programação. Embora tenham sido observadas possibilidades de inclusão de novos requisitos, a abordagem proposta pelo CoderBot o consolida como uma ferramenta educacional de apoio relevante.

7.2 Implicações

Utilizar uma nova ferramenta sempre cria situações de impacto, e com o CoderBot não é diferente, portanto nessa seção são elencadas as principais implicações ao utilizar essa ferramenta.

7.2.1 Implicações para os discentes

Aos discentes temos a implicação de uma ferramenta que serve de apoio em seu aprendizado, entretanto, como ressaltado, ela necessita de ser usado em conjunto com outro método de aprendizagem, ou seja, a carga cognitiva relacionada ao aspecto teórico de elementos de programação não são explicados na ferramenta, e sim como escrever códigos para resolver problemas e como resolver problemas de código, por meio da metodologia ativa de Aprendizagem Baseada em Exemplos combinando as formas correta e errôneas de apresentação dos códigos. Cabe ao estudante em momento de aprendizagem, usar a ferramenta após contato inicial com o conteúdo da disciplina, para aprender estudando os elementos do *template* e códigos, ou mesmo, consolidar seus modelos mentais absorvendo a carga cognitiva dos exemplos corretos e incorretos.

7.2.2 Implicações para os docentes

No que diz respeito aos docentes, a utilização da ferramenta em sua versão atual demandará tempo para a produção dos exemplos que ela exibirá, em consonância com o conteúdo a ser ministrado. Isso pode impactar no aumento do tempo de planejamento da aula. Entretanto, com a configuração e disposição dos exemplos, a ferramenta funciona como um apoio disponível 24 horas por dia, permitindo que o estudante busque respostas ou esclareça dúvidas de forma autônoma, com *feedback* instantâneo, sem a necessidade de intervenção do docente. Dessa forma, a busca pelo docente ocorre apenas em situações específicas em que o CoderBot não conseguiu sanar a dúvida, reduzindo, assim, a demanda de esclarecimento de dúvidas rotineiras.

É importante ressaltar que está previsto como trabalho futuro a proposta de uma ferramenta de apoio a criação dos exemplos para o docente além de contemplar de geração de exemplos por meio de IAs generativas, o que resolveria a necessidade de envio de carga de exemplos por parte do docente, cabendo apenas o planejamento e a verificação da coerência com a disciplina em execução.

7.2.3 Implicações para os pesquisadores de Educação em Programação

As implicações do CoderBot para os pesquisadores de Educação em Programação envolvem aspectos relacionados à consolidação da ferramenta e à implementação de novos recursos, conforme indicado nos resultados do experimento. Além disso, a ferramenta abre uma gama de propostas que podem ser investigadas em novas pesquisas, como, por exemplo, a inclusão de elementos de gamificação e recursos de ambiente de desenvolvimento integrado (IDE), permitindo que o estudante codifique e receba *feedback* sobre seus códigos. Outro impacto relatado que também necessita de investigação é se o CoderBot, o assistente virtual para apoiar à aprendizagem de programação pode contribuir para a redução dos altos índices de reprovações e desistências em disciplinas de programação.

Espera-se que novos trabalhos e perspectivas dos educadores em programação sejam propostas de metodologias inovadoras a serem adotadas na ferramenta, visando propósito de evolução da tecnologia.

7.3 Contribuições do Trabalho

As contribuições desta dissertação de mestrado são:

- Desenvolvimento e avaliação de uma ferramenta para apoiar a aprendizagem de conteúdos de programação:
  - Um assistente virtual com interação baseada em cliques de botão para apresentação de exemplos trabalhados, tanto corretos como incorretos, fornecendo *feedback*;
- Definição e avaliação de um *template* para construção de *worked examples* que tenham os atributos relevantes para o aluno durante a aprendizagem de programação:
  - Definição de atributos importantes para construção do *template*.
- Disseminação dos resultados obtidos nesta pesquisa. Foram publicados trabalhos que apoiaram na construção desta proposta de mestrado, conforme listado a seguir:
  - **GARCIA, R. S.**; VALLE, P. H. D.; BASSO, F.; SILVA, W.. Em direção ao desenvolvimento de um chatbot baseado em exemplos para prática de conteúdos de programação. In: III Simpósio Brasileiro de Educação em Computação, 2023, Virtual. Anais Estendidos do III Simpósio Brasileiro de Educação em Computação (EDUCOMP Estendido), 2023.
  - MIRANDA, A.; **GARCIA, R.**; LUNARDI, G. M.; VILELA, R. F.; VALLE, P. H. D.; SILVA, W. Projeto e Avaliação de um Template de Worked Exemplos para o Ensino de Programação. In: XXXIV Simpósio Brasileiro de Informática na Educação (SBIE), 2023, Passo Fundo. XII Congresso Brasileiro de Informática na Educação, 2023.
  - MENDES, A.; **GARCIA, R.**; VILLA, J.; ORAN, A.; SANTANA, B. S.; GUEDES, G. T. A.; SILVA, D. G.; VALLE, P.; SILVA, W. Avaliando a Autoeficácia e a Aceitação do CoderBot em Cursos Introdutórios de Programação: um estudo exploratório. In: XXXV Simpósio Brasileiro de Informática na Educação (SBIE), 2024, Rio de Janeiro. Sociedade Brasileira de Computação, 2024. pp. 3264-3273.
  - VILLA, J.; **GARCIA, R.**; MIRANDA, A. L. M.; ORAN, A.; GUEDES, G. T. A.; SANTANA, B. S.; SILVA, D. G.; VALLE, P.; SILVA, W. Perspectiva

dos Estudantes sobre um Agente Pedagógico Baseado em Exemplos para a Aprendizagem de Programação: uma análise qualitativa. In: XXXV Simpósio Brasileiro de Informática na Educação (SBIE), 2024, Rio de Janeiro. Sociedade Brasileira de Computação, 2024. pp. 459-473.

– MIRANDA, A. L. M.; **GARCIA, R.**; ORAN, A. C.; GUEDES, G. T. A.; SANTANA, B. S.; SILVA, D. G.; VALLE, P. H. D.; SILVA, W. Avaliação de Usabilidade do CoderBot como Recurso Pedagógico no Ensino de Programação. In: WORKSHOP SOBRE BOTS NA ENGENHARIA DE SOFTWARE, 1., 2024, Curitiba. Sociedade Brasileira de Computação, 2024. p. 11-20.

- Foram publicados também trabalhos que apoiaram indiretamente ao longo desta proposta de mestrado, conforme listado a seguir:

  – SILVA, W. MIRANDA, A.; **GARCIA, R.**; RIBEIRO, M.O., VILELA, R. F.; NAKAMURA, W; LUNARDI, G. M.; VALLE, P. H. D.; Aprendendo com os Erros dos Outros: Um relato sobre a adoção de Exemplos Errôneos como ferramenta de Aprendizagem de Casos de Uso. In: XXXIV Simpósio Brasileiro de Informática na Educação (SBIE), 2023, Passo Fundo. XII Congresso Brasileiro de Informática na Educação, 2023.

  – CARGNELUTTI, R; BERNARDINO, M.; **GARCIA, R.**; SILVA, W.; Um Estudo Exploratório sobre o uso do ChatGPT na Melhoria e Revisão da Escrita de Artigos Científicos. In: XXXIV Simpósio Brasileiro de Informática na Educação (SBIE), 2023, Passo Fundo. XII Congresso Brasileiro de Informática na Educação, 2023.

  – CARVALHO, L. V.; VALLE, P. H. D.; LEIFHEIT, B. R.; CABREJOS, L. E. R.; NAKAMURA, W.; GUERINO, G. C.; **GARCIA, R. S.**; SILVA, W. What Do We Know About Usability Evaluation for Chatbots?: A Systematic Mapping Study. In: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAÇÃO (SBSI), 20., 2024, Juiz de Fora. Sociedade Brasileira de Computação, 2024.

7.4 Trabalhos Futuros

Como trabalhos futuros, destaca-se a proposta de desenvolvimento de uma funcionalidade que permita aos docentes cadastrar turmas e inserir exemplos de forma prática, promovendo maior flexibilidade na configuração do conteúdo.

A possibilidade de integrar Modelos de Linguagem de Grande Escala (LLMs) para melhorar a geração e personalização de exemplos também é uma prioridade, proporcionando um nível de customização da ferramenta para atender às necessidades individuais dos estudantes.

Outro ponto de grande relevância seria a comparação do **CoderBot** com outros *chatbots* educacionais, a fim de avaliar sua eficácia relativa no apoio à aprendizagem de programação.

Além disso, é importante investigar o andamento dos estudantes durante o uso da ferramenta, analisando seu progresso e o impacto nos resultados de aprendizado.

A avaliação da qualidade e da correção dos exemplos, juntamente com a análise dos *feedbacks* fornecidos, é fundamental para garantir a eficácia do aprendizado.

Outros elementos que podem impactar o engajamento dos alunos incluem a implementação de um sistema de ranking de exemplos e métricas detalhadas de uso. Esses recursos podem oferecer *insights* valiosos sobre o desempenho dos estudantes e a popularidade dos exemplos apresentados, contribuindo para uma análise mais abrangente do uso da ferramenta e para o aprimoramento de sua interface e funcionalidade.

REFERÊNCIAS

ABDUL-RAHMAN, S.-S.; du Boulay, B. Learning programming via worked-examples: Relation of learning styles to cognitive load. **Computers in Human Behavior**, v. 30, p. 286–298, 2014. ISSN 0747-5632. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S074756321300335X>.

ALMEIDA, A. V. de; ARAÚJO, F. P. O. Annebot: Um chatbot para auxiliar no processo de ensino e aprendizagem do pensamento computacional. In: SBC. **Anais Estendidos do I Simpósio Brasileiro de Educação em Computação**. [S.l.], 2021. p. 12–13.

ALVES, G.; REBOUÇAS, A.; SCAICO, P. Coding dojo como prática de aprendizagem colaborativa para apoiar o ensino introdutório de programação: Um estudo de caso. In: SBC. **Anais do XXVII Workshop sobre Educação em Computação**. [S.l.], 2019. p. 276–290.

ALVES, R. de S.; NASCIMENTO, G. M. do; SOUSA, R. R. de. Elementos do emprego de chatbots para auxílio no ensino de programação: Uma revisão sistemática da literatura. **Brazilian Journal of Development**, v. 7, n. 5, p. 43908–43927, 2021.

ARDIMANSYAH, M.; WIDIANTO, M. Development of online learning media based on telegram chatbot (case studies: Programming courses). In: IOP PUBLISHING. **Journal of Physics: Conference Series**. [S.l.], 2021. v. 1987, n. 1, p. 012006.

ATKINSON, R. K. et al. Learning from examples: Instructional principles from the worked examples research. **Review of Educational Research**, SAGE Publications Sage CA: Los Angeles, CA, v. 70, n. 2, p. 181–214, 2000.

BII, P. Chatbot technology: A possible means of unlocking student potential to learn how to learn. **Educational Research**, v. 4, n. 2, p. 218–221, 2013.

BILGIN, T. T.; YAVUZ, E. Integrating a dialogue tree based turkish chatbot into an open source python coding editor. In: IEEE. **2022 3rd International Informatics and Software Engineering Conference (IISEC)**. [S.l.], 2022. p. 1–5.

BRAN, A. M.; SCHWALLER, P. Transformers and large language models for chemistry and drug discovery. In: **Drug Development Supported by Informatics**. [S.l.]: Springer, 2024. p. 143–163.

BROOKE, J. et al. Sus-a quick and dirty usability scale. **Usability evaluation in industry**, London–, v. 189, n. 194, p. 4–7, 1996.

CALDERON, I.; SILVA, W.; FEITOSA, E. Um mapeamento sistemático da literatura sobre o uso de metodologias ativas durante o ensino de programação no brasil. In: SBC. **Anais do XXXII Simpósio Brasileiro de Informática na Educação**. [S.l.], 2021. p. 1152–1161.

CARREIRA, G. et al. Pyo, a chatbot assistant for introductory programming students. In: IEEE. **2022 International Symposium on Computers in Education (SIIE)**. [S.l.], 2022. p. 1–6.

CARTWRIGHT, N.; MCMULLIN, E. **How the laws of physics lie**. [S.l.]: American Association of Physics Teachers, 1984.

CHAO, P.-Y. Exploring students' computational practice, design and performance of problem-solving through a visual programming environment. **Computers & Education**, Elsevier, v. 95, p. 202–215, 2016.

CHAVES, A. P. Desenho de linguagem de chatbots: influência da variação da linguagem na experiência do usuário com chatbot assistente de turismo. In: SBC. **Anais Estendidos do XVIII Simpósio Brasileiro de Sistemas Colaborativos**. [S.l.], 2023. p. 42–47.

CHEN, T.; XU, L.; ZHU, K. Fritzbot: A data-driven conversational agent for physical-computing system design. **International Journal of Human-Computer Studies**, Elsevier, v. 155, p. 102699, 2021.

CHEN, X.; MITROVIC, A.; MATHEWS, M. Do erroneous examples improve learning in addition to problem solving and worked examples? In: SPRINGER. **Intelligent Tutoring Systems: 13th International Conference, ITS 2016, Zagreb, Croatia, June 7-10, 2016. Proceedings 13**. [S.l.], 2016. p. 13–22.

CLARIZIA, F. et al. Chatbot: An education support system for student. In: SPRINGER. **International Symposium on Cyberspace Safety and Security**. [S.l.], 2018. p. 291–302.

CORONADO, M. et al. A cognitive assistant for learning java featuring social dialogue. **International Journal of Human-Computer Studies**, Elsevier, v. 117, p. 55–67, 2018.

CORREIA, J. et al. Unveiling the potential of a conversational agent in developer support: Insights from mozilla's pdf. js project. 2024.

CSIKSZENTMIHALYI, M. Finding flow: The psychology of engagement with everyday life. **Basic Book**, 1997.

CSIKZENTMIHALYI, M. **Beyond boredom and anxiety**. [S.l.]: San Francisco: Jossey-Bass, 1975.

DAHIYA, M. A tool of conversation: Chatbot. **International Journal of Computer Sciences and Engineering**, v. 5, n. 5, p. 158–161, 2017.

DOHMKE, T.; IANSITI, M.; RICHARDS, G. Sea change in software development: Economic and productivity analysis of the ai-powered developer lifecycle. **arXiv preprint arXiv:2306.15033**, 2023.

EDWARDS, J. et al. Syntax exercises in cs1. In: **Proceedings of the 2020 ACM Conference on International Computing Education Research**. [S.l.: s.n.], 2020. p. 216–226.

FADHIL, A.; VILLAFIORITA, A. An adaptive learning with gamification conversational uis: The rise of cibopolibot. In: **Adjunct Publication of the 25th Conference on user modeling, adaptation and personalization**. [S.l.]: ACM, 2017. p. 408–412. ISBN 1450350674.

GARCES, S. et al. Engaging students in active exploration of programming worked examples. **Education and Information Technologies**, Springer, v. 28, n. 3, p. 2869–2886, 2023.

GARCIA, R. Dataset referente a RSL sobre chatbots e ensino de programaçao. 12 2024. Available from Internet: <https://figshare.com/articles/dataset/Pacote_de_Replica_o/27964311>.

GONDA, D. E.; CHU, B. Chatbot as a learning resource? creating conversational bots as a supplement for teaching assistant training course. In: IEEE. **2019 IEEE International Conference on Engineering, Technology and Education (TALE)**. [S.l.], 2019. p. 1–5.

GROSSE, C. S.; RENKL, A. Finding and fixing errors in worked examples: Can this foster learning outcomes? **Learning and instruction**, Elsevier, v. 17, n. 6, p. 612–634, 2007.

GUERINO, G. et al. Lições aprendidas sobre ensino remoto emergencial sob a perspectiva de docentes dos cursos de computação. In: **Educação e ensino em diferentes espaços e áreas do conhecimento: desafios, contextos e diálogos [recurso eletrônico]**. uritiba, PR, BR: Editora Bagai, 2021. p. 60–73. ISBN 978-65-81368-92-0. Available from Internet: <http://portal.acm.org/citation.cfm?id=887006.887010>.

GUO, P. J. Non-native english speakers learning computer programming: Barriers, desires, and design opportunities. In: **Proceedings of the 2018 CHI conference on human factors in computing systems**. [S.l.: s.n.], 2018. p. 1–14.

HAMZAH, W. M. A. F. W. et al. Using learning analytics to explore responses from student conversations with chatbot for education. **International Journal of Engineering Pedagogy**, v. 11, n. 6, 2021.

HEVNER, A. R. A three cycle view of design science research. **Scandinavian journal of information systems**, v. 19, n. 2, p. 4, 2007.

HIEN, H. T. et al. Intelligent assistants in higher-education environments: the fit-ebot, a chatbot for administrative and learning support. In: **Proceedings of the 9th International Symposium on Information and Communication Technology**. [S.l.: s.n.], 2018. p. 69–76.

HIREMATH, G. et al. Chatbot for education system. **International Journal of Advance Research, Ideas and Innovations in Technology**, v. 4, n. 3, p. 37–43, 2018.

HOBERT, S. Say hello to 'coding tutor'! design and evaluation of a chatbot-based learning system supporting students to learn to program. 2019.

HOSSEINI, R. et al. Improving engagement in program construction examples for learning python programming. **International Journal of Artificial Intelligence in Education**, Springer, v. 30, n. 2, p. 299–336, 2020.

HUANG, Y.-H. et al. Comparison of different approaches on example-based learning for novice and proficient learners. **Human-centric Computing and Information Sciences**, v. 5, 12 2015.

ISMAIL, M.; ADE-IBIJOLA, A. Lecturer's apprentice: A chatbot for assisting novice programmers. In: IEEE. **2019 international multidisciplinary information technology and engineering conference (IMITEC)**. [S.l.], 2019. p. 1–8.

IUNG, A. et al. Systematic mapping study on domain-specific language development tools. **Empirical Software Engineering**, Springer, v. 25, p. 4205–4249, 2020.

JACKSON, S. A.; EKLUND, R. C. Assessing flow in physical activity: The flow state scale–2 and dispositional flow scale–2. **Journal of sport and exercise psychology**, Human Kinetics, Inc., v. 24, n. 2, p. 133–150, 2002.

KASINATHAN, V. et al. Tictad: A chatterbot for learning visual c# programming based on expert system,". **Indonesian Journal of Electrical Engineering and Computer Science**, v. 11, n. 2, p. 740–746, 2018.

KEELE, S. et al. **Guidelines for performing systematic literature reviews in software engineering**. [S.l.]: Technical report, ver. 2.3 ebse technical report. ebse, 2007.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Citeseer, 2007.

KOPP, V.; STARK, R.; FISCHER, M. R. Fostering diagnostic knowledge through computer-supported, case-based worked examples: effects of erroneous examples and feedback. **Medical education**, Wiley Online Library, v. 42, n. 8, p. 823–829, 2008.

KUHRMANN, M.; FERNÁNDEZ, D. M.; DANEVA, M. On the pragmatic design of literature studies in software engineering: an experience-based guideline. **Empirical software engineering**, Springer, v. 22, p. 2852–2891, 2017.

KUO, Y.-C.; CHEN, Y.-A. The impact of chatbots using concept maps on correction outcomes–a case study of programming courses. **Education and Information Technologies**, Springer, v. 28, n. 7, p. 7899–7925, 2023.

LAI, X.; WONG, G. K.-w. Collaborative versus individual problem solving in computational thinking through programming: A meta-analysis. **British Journal of Educational Technology**, Wiley Online Library, v. 53, n. 1, p. 150–170, 2022.

LANE, H. C.; VANLEHN, K. Coached program planning: Dialogue-based support for novice program design. In: **Proceedings of the 34th SIGCSE technical symposium on Computer science education**. [S.l.: s.n.], 2003. p. 148–152.

LIDÉN, A.; NILROS, K. **Percieved benefits and limitations of chatbots in higher education**. 2020.

LIKERT, R. A technique for the measurement of attitudes. **Archives of Psychology**, 1932.

LÓPEZ-PERNAS, S. et al. Examining the use of an educational escape room for teaching programming in a higher education setting. **IEEE Access**, IEEE, v. 7, p. 31723–31737, 2019.

LUXTON-REILLY, A. Learning to program is easy. In: **Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education**. [S.l.: s.n.], 2016. p. 284–289.

LUXTON-REILLY, A. et al. Introductory programming: a systematic literature review. In: **Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education**. [S.l.: s.n.], 2018. p. 55–106.

MAGEIRA, K. et al. Educational ai chatbots for content and language integrated learning. **Applied Sciences**, MDPI, v. 12, n. 7, p. 3239, 2022.

MALIK, S. I. et al. Fostering the learning process in a programming course with a chatbot. **International Journal of Online Pedagogy and Course Design (IJOPCD)**, IGI Global, v. 12, n. 1, p. 1–17, 2022.

MCGINN, K. M.; LANGE, K. E.; BOOTH, J. L. A worked example for creating worked examples. **Mathematics Teaching in the Middle School**, National Council of Teachers of Mathematics, v. 21, n. 1, p. 26–33, 2015.

MCLAREN, B. M. et al. The efficiency of worked examples compared to erroneous examples, tutored problem solving, and problem solving in computer-based learning environments. **Computers in Human Behavior**, Elsevier, v. 55, p. 87–99, 2016.

MERRIENBOER, J. J. van. Instructional strategies for teaching computer programming: Interactions with the cognitive style reflection-impulsivity. **Journal of research on computing in education**, Taylor & Francis, v. 23, n. 1, p. 45–53, 1990.

MIRANDA, A. et al. Projeto e avaliação de um template de worked examples para o ensino de programação. In: **Anais do XXXIV Simpósio Brasileiro de Informática na Educação**. Porto Alegre, RS, Brasil: SBC, 2023. p. 1673–1684. ISSN 0000-0000. Available from Internet: <https://sol.sbc.org.br/index.php/sbie/article/view/26788>.

MITCHELL, C. M.; BOYER, K. E.; LESTER, J. C. When to intervene: Toward a markov decision process dialogue policy for computer science tutoring. In: CITESEER. **The First Workshop on AI-supported Education for Computer Science (AIEDCS 2013)**. [S.l.], 2013. p. 40.

MONDAL, A. et al. Chatbot: An automated conversation system for the educational domain. In: IEEE. **2018 International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)**. [S.l.], 2018. p. 1–5.

MONK, P. M. **Fundamentals of electroanalytical chemistry**. [S.l.]: John Wiley & Sons, 2008.

MORAIS, C. G. B. Ensino e aprendizagem de programação: estudo de caso no ensino superior. 2022.

NAKAMURA, W. T. et al. Are scale-based techniques enough for learners to convey their ux when using a learning management system? **Revista Brasileira de Informática na Educação**, v. 27, n. 1, p. 104–131, 2019.

NAKAMURA, W. T. et al. What factors affect the ux in mobile apps? a systematic mapping study on the analysis of app store reviews. **Journal of Systems and Software**, Elsevier, v. 193, p. 111462, 2022.

NGUYEN, H. D. et al. Design intelligent educational chatbot for information retrieval based on integrated knowledge bases. **IAENG International Journal of Computer Science**, v. 49, n. 2, p. 531–541, 2022.

OKONKWO, C. W.; ADE-IBIJOLA, A. Python-bot: A chatbot for teaching python programming. **Engineering Letters**, v. 29, n. 1, 2020.

OKONKWO, C. W.; ADE-IBIJOLA, A. Chatbots applications in education: A systematic review. **Computers and Education: Artificial Intelligence**, Elsevier, v. 2, p. 100033, 2021.

OKONKWO, C. W.; ADE-IBIJOLA, A. Revision-bot: A chatbot for studying past questions in introductory programming. **IAENG International Journal of Computer Science**, v. 49, n. 3, 2022.

ONDÁŠ, S.; PLEVA, M.; HLÁDEK, D. How chatbots can be involved in the education process. In: IEEE. **2019 17th International Conference on Emerging eLearning Technologies and Applications (ICETA)**. [S.l.], 2019. p. 575–580.

OROMA, J.; WANGA, H.; NGUMBUKE, F. Challenges of teaching and learning computer programming in a developing country: lessons from tanzania. In: IATED. **INTED2012 Proceedings**. [S.l.], 2012. p. 3820–3826.

PASCHOAL, L. N. et al. An experimental study on a conversational agent in software testing lessons. **Informatics in Education**, Vilnius University Institute of Data Science and Digital Technologies, 2022.

PENNEY, J. et al. Anticipating user needs: Insights from design fiction on conversational agents for computational thinking. In: SPRINGER. **International Workshop on Chatbot Research and Design**. [S.l.], 2023. p. 204–219.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, Elsevier, v. 64, p. 1–18, 2015.

RAICHE, A.-P. et al. Factors influencing acceptance and trust of chatbots in juvenile offenders' risk assessment training. **Frontiers in Psychology**, Frontiers Media SA, v. 14, p. 1184016, 2023.

RAMALINGAM, V.; LABELLE, D.; WIEDENBECK, S. Self-efficacy and mental models in learning to program. In: **Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education**. [S.l.: s.n.], 2004. p. 171–175.

RETNOWATI, E.; MARISSA. Designing worked examples for learning tangent lines to circles. **Journal of Physics: Conference Series**, v. 983, p. 012124, 03 2018.

REZENDE, F.; BARROS, S. de S. Students' navigation patterns in the interaction with a mechanics hypermedia program. **Computers & Education**, Elsevier, v. 50, n. 4, p. 1370–1382, 2008.

ROBINS, A.; ROUNTREE, J.; ROUNTREE, N. Learning and teaching programming: A review and discussion. **Computer science education**, Citeseer, v. 13, n. 2, p. 137–172, 2003.

ROBINS, A. V. 12 novice programmers and introductory programming. **The Cambridge handbook of computing education research**, Cambridge University Press, p. 327, 2019.

ROCA, M. D. L. et al. The impact of a chatbot working as an assistant in a course for supporting student learning and engagement. **Computer Applications in Engineering Education**, Wiley Online Library, p. e22750, 2024.

RODIAWATI, A.; RETNOWATI, E. How to design worked examples for learning patterns in mathematics. **Journal of Physics: Conference Series**, v. 1320, p. 012045, 10 2019.

RUAN, S. et al. Quizbot: A dialogue-based adaptive learning system for factual knowledge. In: **Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems**. [S.l.: s.n.], 2019. p. 1–13.

SAURO, J.; LEWIS, J. R. When designing usability questionnaires, does it hurt to be positive? In: **Proceedings of the SIGCHI conference on human factors in computing systems**. [S.l.: s.n.], 2011. p. 2215–2224.

SMUTNY, P.; SCHREIBEROVA, P. Chatbots for learning: A review of educational chatbots for the facebook messenger. **Computers Education**, v. 151, p. 103862, 2020. ISSN 0360-1315. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0360131520300622>.

STEINMACHER, I. et al. A systematic literature review on the barriers faced by newcomers to open source software projects. **Information and Software Technology**, Elsevier, v. 59, p. 67–85, 2015.

SWELLER, J.; MERRIENBOER, J. J. V.; PAAS, F. G. Cognitive architecture and instructional design. **Educational psychology review**, JSTOR, p. 251–296, 1998.

TONHãO, S.; COLANZI, T.; STEINMACHER, I. Using real worked examples to aid software engineering teaching. In: **Proceedings of the XXXV Brazilian Symposium on Software Engineering**. New York, NY, USA: Association for Computing Machinery, 2021. (SBES '21), p. 133–142. ISBN 9781450390613. Available from Internet: <https://doi.org/10.1145/3474624.3476970>.

TURPEN, C.; DANCY, M.; HENDERSON, C. Perceived affordances and constraints regarding instructors' use of peer instruction: Implications for promoting instructional change. **Physical Review Physics Education Research**, APS, v. 12, n. 1, p. 010116, 2016.

VENKATESH, V.; DAVIS, F. D. A theoretical extension of the technology acceptance model: Four longitudinal field studies. **Management science**, INFORMS, v. 46, n. 2, p. 186–204, 2000.

VERLEGER, M.; PEMBRIDGE, J. A pilot study integrating an ai-driven chatbot in an introductory programming course. In: IEEE. **2018 IEEE frontiers in education conference (FIE)**. [S.l.], 2018. p. 1–4.

WANG, X.-H. et al. Exploration and practice of blended teaching model based flipped classroom and spoc in higher university. **Journal of Education and Practice**, ERIC, v. 7, n. 10, p. 99–104, 2016.

WINKLER, R. et al. Sara, the lecturer: Improving learning in online education with a scaffolding-based conversational agent. In: **Proceedings of the 2020 CHI conference on human factors in computing systems**. [S.l.: s.n.], 2020. p. 1–14.

WOLLNY, S. et al. Are we there yet?-a systematic literature review on chatbots in education. **Frontiers in artificial intelligence**, Frontiers Media SA, v. 4, p. 654924, 2021.

YANG, T.-C.; LIN, Z.-S. Enhancing elementary school students' computational thinking and programming learning with graphic organizers. **Computers Education**, v. 209, p. 104962, 2024. ISSN 0360-1315. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0360131523002397>.

ZHANG, H.; KITCHENHAM, B.; PFAHL, D. Software process simulation modeling: an extended systematic review. In: SPRINGER. **International Conference on Software Process**. [S.l.], 2010. p. 309–320.

ZHAO, D. et al. Game-based learning: enhancing student experience, knowledge gain, and usability in higher education programming courses. **IEEE Transactions on Education**, IEEE, v. 65, n. 4, p. 502–513, 2022.

ANEXO A — QUESTIONÁRIOS

Tabela 12 – Afirmativas do Questionário TAM

Aspecto	**Q#**	**Questão**
Utilidade Percebida	Q1	Usar o CoderBot melhora o meu desempenho ao resolver problemas de programação.
	Q2	Usar o CoderBot melhora a minha produtividade na resolução de exercícios.
	Q3	O CoderBot facilita o meu trabalho durante o aprendizado de programação.
	Q4	Eu acho o CoderBot útil para me auxiliar no aprendizado de programação.
Facilidade de Uso Percebida	Q5	O CoderBot foi claro e fácil de entender para mim.
	Q6	Usar o CoderBot não demandou muito esforço mental.
	Q7	Eu acho que o CoderBot é um chatbot fácil de usar.
	Q8	Eu acho fácil lembrar como executar as tarefas usando o CoderBot.
Intenção de Uso Percebida	Q9	Assumindo que eu tenha acesso ao CoderBot, eu pretendo usá-lo futuramente para me apoiar no aprendizado de programação.
	Q10	Dado que eu tenha acesso ao CoderBot, eu prevejo que eu o usaria futuramente no aprendizado de programação.
	Q11	Eu pretendo usar o CoderBot para me auxiliar na resolução de exercícios no próximo mês.

Fonte: De autoria própria.

Tabela 13 – Questionário da Escala de Usabilidade de Sistemas (SUS)

Q#	**Questão**
Q1	Eu acho que eu gostaria de utilizar este sistema frequentemente.
Q2	Eu achei o sistema desnecessariamente complexo.
Q3	Eu achei o sistema fácil de usar.
Q4	Eu acho que eu precisaria da ajuda de uma pessoa com conhecimentos técnicos para eu conseguir utilizar o sistema.
Q5	Eu acho que as várias funcionalidades do sistema estão bem integradas.
Q6	Eu achei que tinha muita inconsistência no sistema.
Q7	Eu acho que a maioria das pessoas iriam aprender facilmente a utilizar esse sistema.
Q8	Eu achei o sistema muito incômodo de usar.
Q9	Eu me senti muito confiante em utilizar o sistema.
Q10	Eu precisei aprender muito antes de utilizar esse sistema.

Fonte: De autoria própria.

Tabela 14 – Questionário de Experiência de Fluxo

Q#	**Questão**
Q1	Com o CoderBot, eu senti que era competente o suficiente para atender às demandas da situação
Q2	Com o CoderBot fiz as coisas de forma espontânea e automática, sem ter que pensar
Q3	Com o CoderBot, eu tinha uma forte noção do que queria fazer
Q4	Com o CoderBot, eu tinha uma boa ideia sobre o meu desempenho enquanto estava envolvido na tarefa/atividade
Q5	Com o CoderBot, eu estava completamente focado na tarefa em questão
Q6	Com o CoderBot, tive uma sensação de total controle sobre o que estava fazendo
Q7	Com o CoderBot, eu não estava preocupado com o que os outros poderiam estar pensando de mim
Q8	Com o CoderBot, a maneira como o tempo passou parecia diferente do normal
Q9	Com o CoderBot, achei a experiência extremamente gratificante

Fonte: De autoria própria.

Tabela 15 – Questionário de Autoeficácia

Aspecto	**Q#**	**Questão**
Satisfação	Q1	O quanto você gostou de usar este chatbot?
	Q2	Quão útil foi o CoderBot para você no apoio à resolução dos exercícios?
	Q3	Como você avaliaria sua satisfação geral com o CoderBot?
Usabilidade	Q4	O Quão fácil foi usar o CoderBot?
	Q5	Quão compreensíveis foram as respostas fornecidas pelo CoderBot?
	Q6	Quão aceitável é o tempo gasto interagindo com o CoderBot?
Benevolência	Q7	Você sentiu que suas dúvidas foram corretamente compreendidas pelo CoderBot?
	Q8	Você sentiu que as respostas fornecidas pelo CoderBot foram claras?
	Q9	Você achou a interação com o CoderBot agradável?
Credibilidade	Q10	O CoderBot deve ser integrado nas práticas de ensino de programação?
	Q11	O CoderBot deve ser utilizado obrigatoriamente em aulas de programação?
	Q12	Você sentiu que o CoderBot era confiável?

Fonte: De autoria própria.

Tabela 16 – Questionário de Motivação Intrínseca

Aspecto	**Q#**	**Questão**
Interesse/Prazer	Q1	Enquanto realizava as tarefas usando o CoderBot, pensava no quanto gostei de realizá-las.
	Q2	Realizar as tarefas usando o CoderBot foi muito interessante.
	Q3	Realizar as tarefas usando o CoderBot foi divertido.
	Q4	Eu gostei muito de fazer as tarefas usando o CoderBot.
	Q5	As tarefas usando o CoderBot eram muito chatas.
	Q6	As tarefas usando o CoderBot eram muito interessantes.
	Q7	Eu descreveria as tarefas usando o CoderBot como muito agradáveis.
Escolha percebida	Q8	Senti que estava conseguindo fazer o que queria enquanto estava realizando as tarefas usando o CoderBot.
	Q9	Senti que era minha escolha fazer as tarefas usando o CoderBot.
	Q10	Eu realmente não tive escolha em fazer as tarefas usando o CoderBot.
	Q11	Eu senti que tinha de fazer as tarefas usando CoderBot.
	Q12	Eu fiz as tarefas da disciplina no CoderBot porque não tinha escolha.
Competência percebida	Q13	Eu acho que sou muito bom em realizar as tarefas usando o CoderBot.
	Q14	Acho que me saí muito bem ao realizar as tarefas usando o CoderBot, em comparação com outros alunos.
	Q15	Estou satisfeito com o meu desempenho nas tarefas usando o CoderBot.
	Q16	Eu me senti bastante competente ao realizar as tarefas usando o CoderBot.
	Q17	Depois de ter trabalhado nas tarefas da disciplina no CoderBot durante algum tempo, eu me senti bastante competente.
Pressão/Tensão	Q18	Não me senti nada nervoso ao fazer as tarefas usando o CoderBot.
	Q19	Eu me senti tenso enquanto fazia a tarefas usando o CoderBot.
	Q20	Eu me senti relaxado enquanto fazia as tarefas usando o CoderBot.
	Q21	Eu estava ansioso enquanto fazia as tarefas usando o CoderBot.
	Q22	Eu me senti pressionado enquanto fazia as tarefas usando o CoderBot.

Fonte: De autoria própria.

Tabela 17 – Questionário de Atitudes dos Estudantes sobre as Estratégias

Q#	**Questão**
Q1	Com o uso do CoderBot, me senti mais motivado para aprender que habitualmente
Q2	O CoderBot me permitiu melhorar minhas opiniões críticas
Q3	Eu me senti mais conectado com os outros ao adotar o CoderBot
Q4	O CoderBot me fez sentir parte da aula por entender melhor os conteúdos
Q5	Com o CoderBot, pude entender como expressar minhas opiniões livremente
Q6	Com o CoderBot, descobri as falhas no que anteriormente acreditei estar certo
Q7	Achei o CoderBot muito útil para a minha aprendizagem
Q8	CoderBot facilitou que eu participasse mais ativamente da minha aprendizagem
Q9	O CoderBot ajudou a aumentar minhas habilidades de pensamento crítico
Q10	O CoderBot ofereceu uma boa experiência para aprender as atividades referentes ao Marco 1

Fonte: De autoria própria.

Tabela 18 – Avaliação sobre a Aprendizagem Percebida

Q#	**Questão**
Q1	O CoderBot contribuiu para a minha aprendizagem no conteúdo ensinado na disciplina.
Q2	O CoderBot foi mais eficiente para minha aprendizagem, em comparação com outras atividades da disciplina.
Q3	O CoderBot auxiliou a relembrar os conceitos aprendidos sobre o tema.
Q4	O CoderBot contribuiu para interpretar os conceitos aprendidos na disciplina.
Q5	O CoderBot contribuiu para aplicar dos conceitos aprendidos durante a resolução dos problemas.
Q6	O CoderBot contribuiu para organizar as minhas atividades relacionadas ao conteúdo.
Q7	O CoderBot contribuiu para verificar se o código foi construído corretamente.
Q8	O CoderBot contribuiu para elaborar o código referente ao conteúdo da disciplina.

Fonte: De autoria própria.