PLATAFORMA EDUC-DEV DE APRENDIZAGEM EM PROGRAMAÇÃO COM GAMIFICAÇÃO USANDO INTELIGÊNCIA ARTIFICIAL PARA CORREÇÃO

EDUC-DEV PLATFORM FOR PROGRAMMING LEARNING WITH *GAMIFICATION* USING ARTIFICIAL INTELLIGENCE FOR CORRECTION

Gabriel Dantas Lopes*
Diego Rocha Lima**

RESUMO

O ensino de programação representa um desafio recorrente para instituições de ensino, especialmente diante das dificuldades enfrentadas por estudantes iniciantes em compreender lógica e estruturas de código. Neste contexto, foi desenvolvida uma plataforma de aprendizagem que integra práticas de codificação com elementos de gamificação e correção automática baseada em inteligência artificial (IA), visando tornar o processo mais envolvente e eficaz. A proposta considerou os requisitos de sistema e de usuário e resultou na implementação parcial do *frontend* e *backend* da plataforma. A aplicação conta com desafios práticos e *feedbacks* personalizados gerados por IA, permitindo que os alunos acompanhem seu progresso e aprimorem suas habilidades de forma autônoma. Além disso, a plataforma foi avaliada com alunos da disciplina de Introdução à Programação, demonstrando-se promissora no apoio ao processo de ensino-aprendizagem. A solução busca promover uma experiência mais interativa, reduzir as taxas de desistência e contribuir para o desenvolvimento de competências alinhadas às demandas do mercado.

Palavras-chave: Programação, Gamificação, Inteligência Artificial, Ensino, Correção Automática.

ABSTRACT

Teaching programming is a recurring challenge for educational institutions, especially due to the difficulties beginners face in understanding logic and code structures. In this context, a learning platform was developed that integrates coding practices with gamification elements and automatic feedback powered by artificial intelligence (AI), aiming to make the process more engaging and effective. The proposal was based on system and user requirements and resulted

* Graduando em Bacharelado em Ciência da Computação, Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE), Aracati, Ceará, Brasil. E-mail: gdantaslopes62@gmail.com
** Doutor em Engenharia Elétrica pela Universidade Federal do Rio Grande do Norte (UFRN), Docente do Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE), Aracati, Ceará, Brasil. Email: diego.rocha@ifce.edu.br

in the partial implementation of the platform's frontend and backend. The application includes practical challenges and personalized feedback generated by AI, enabling students to monitor their progress and improve their skills autonomously. Furthermore, the platform was evaluated with students from an Introduction to Programming course and proved to be a promising tool to support the teaching-learning process. The solution aims to provide a more interactive experience, reduce dropout rates, and foster the development of skills aligned with current market demands.

Keywords: Programming, Gamification, Artificial Intelligence, Education, Automated Correction.

1 INTRODUÇÃO

O avanço da tecnologia tem impulsionado o mercado de desenvolvimento de *software*, exigindo cada vez mais que programadores dominem algoritmos, estruturas de dados e boas práticas de desenvolvimento. No entanto, ensinar e garantir que os estudantes aprendam bem os fundamentos da programação continua sendo um dos principais desafios enfrentados pelas instituições de ensino superior. Por não conseguirem compreender de forma sólida um conteúdo, essas dificuldades causam, inevitavelmente, elevadas taxas de insucesso ou desistência dos estudantes de cursos de Computação (GOMES; MENDES; (2014), 2014).

Entre os anos de 2022 e 2023, houve um avanço significativo nas tecnologias de inteligência artificial, especialmente com o lançamento de modelos de linguagem natural como o *ChatGPT*, *Copilot* e *Gemini* (OPENAI, 2022; MICROSOFT, 2023; GOOGLE, 2023). Essas ferramentas demonstra ser bastante eficazes no mercado de trabalho, contribuindo para o aumento da produtividade em diversas áreas. No entanto, no contexto educacional, especialmente entre estudantes, o uso inadequado dessas tecnologias levantou preocupações. Muitos passaram a utilizá-las de forma passiva, como substituto do raciocínio próprio, o que pode comprometer o desenvolvimento do pensamento crítico e da autonomia no aprendizado.

Muitos estudantes iniciantes em cursos de Computação, especialmente nos primeiros períodos, enfrentam dificuldades em aplicar o que foi aprendido e em corrigir seus próprios erros. Uma das consequências desse cenário é a desmotivação com a área de programação, o que pode levar ao atraso ou até à evasão do curso (GOMES; MENDES; (2014), 2014). Além disso, a ausência de *feedback* prático e detalhado pode desmotivar e limitar o desenvolvimento das competências necessárias para atender às exigências do mercado. Os professores, por sua vez, enfrentam o desafio de ensinar em turmas grandes, que inviabiliza a realização de um acompanhamento individualizado, e heterogêneas, que apresentam disparidade de conhecimento e ritmo de aprendizagem (LAHTINEN et al., 2005).

Foi essa ausência de informações mais voltadas ao gerenciamento do processo de aprendizagem dos estudantes que motivou a procurar alternativas que pudessem potencializar a ação docente. Conforme Jr e Cortelazzo (2015), foram analisadas três metodologias: *blended learning*,

ambientes de aprendizagem ou *learning spaces* e sala de aula invertida ou *flipped classroom*, que, pela sua simples adoção, de forma individualizada, apresentam resultados positivos.

A proposta deste trabalho é contribuir para o aprendizado de programação por meio de uma plataforma interativa. A aplicação reúne práticas de codificação com elementos de gamificação, permitindo que o usuário desenvolva seu conhecimento de forma progressiva e motivadora. Essa abordagem não apenas facilita a identificação de pontos de melhoria, mas também incentiva o aprendizado contínuo de maneira interativa e prática. Ao preencher essa lacuna, espera-se formar profissionais mais capacitados e confiantes. Diante dessa situação, é necessário buscar maneiras de propor soluções que possam tratar essa condição de forma efetiva (BRITO; MADEIRA, 2015).

Para resolver o problema identificado, este trabalho propõe uma plataforma inovadora que combina aprendizado por desafios práticos com *feedbacks* automatizados. A aplicação conta com funcionalidades como níveis de habilidade e uma interface baseada em gamificação. Essa definição é amplamente adotada em pesquisas na área da educação, inclusive por autores como Brazil e Baruque (2011), que destacam o potencial da gamificação em promover o engajamento dos alunos e tornar o processo de ensino mais dinâmico e eficiente. Nesse contexto, o uso de gamificação tem se mostrado eficaz na obtenção de resultados superiores aos alcançados por métodos tradicionais de ensino, especialmente no que diz respeito à motivação, participação ativa dos estudantes e retenção do conhecimento.

Diante desses desafios, fica evidente a necessidade de métodos inovadores que não apenas facilitem a compreensão dos conceitos de programação, mas também promovam o engajamento e a motivação contínua dos estudantes. A utilização de plataformas de aprendizagem baseadas em gamificação surge como uma alternativa promissora para superar as limitações dos métodos tradicionais, proporcionando uma experiência de aprendizado mais interativa, personalizada e eficaz.

Assim, este trabalho tem como objetivo principal o desenvolvimento de uma plataforma de aprendizagem que combina práticas de programação com elementos de gamificação, visando melhorar o desempenho e a retenção de conhecimento dos estudantes de Computação. Acredita-se que a abordagem proposta possa colaborar na redução das taxas de evasão e insucesso acadêmico, além de preparar melhor os alunos para os desafios do mercado de trabalho.

A estrutura deste trabalho está organizada da seguinte forma: na Seção 2, é apresentada a fundamentação teórica, abordando os principais conceitos relacionados a plataformas de aprendizagem, gamificação e metodologias de ensino aplicadas à programação. Na Seção 3, detalha-se a proposta adotada para o desenvolvimento da plataforma. A Seção 4 descreve o processo de implementação e os resultados esperados. Por fim, na Seção 5, são discutidas as conclusões e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados os principais conceitos necessários para a compreensão da proposta do trabalho, como o conceitual de plataforma de aprendizagem, gamificação, conteúdos de introdução a programação, uso de inteligência artificial e juiz online, sendo assim, demonstrando uma abordagem completa juntamente com os objetivos do projeto.

2.1 Plataforma de aprendizagem

As plataformas de aprendizagem são geralmente compostas por recursos interativos, avaliações, simulações e exercícios práticos que proporcionam uma experiência de aprendizado mais envolvente. A utilização de estratégias como gamificação, personalização do ensino e inteligência artificial tem se mostrado eficaz para aumentar o engajamento e melhorar o desempenho dos estudantes (MEIRELES; FERNANDES, 2017). O uso de métricas e ferramentas de monitoramento também contribui para ajustes contínuos e para o acompanhamento do progresso dos usuários.

Além da experiência do usuário, o desenvolvimento de uma plataforma de aprendizagem exige a integração de múltiplos aspectos, incluindo programação, estratégias pedagógicas e metodologias de ensino eficazes. Elementos como gamificação, personalização do ensino e inteligência artificial podem potencializar o aprendizado, tornando o ambiente mais dinâmico e motivador. O uso de métricas e sistemas de acompanhamento de progresso também contribui para uma análise mais precisa do desempenho dos alunos, permitindo ajustes contínuos no processo educacional.

2.2 Mecanismo de Gamificação

A gamificação tem sido cada vez mais utilizada no contexto educacional como uma estratégia para aumentar o engajamento e a motivação dos estudantes. O conceito baseia-se na incorporação de elementos típicos de jogos, como pontos, *rankings*, medalhas e recompensas, em atividades que não são originalmente lúdicas (DETERDING et al., 2011). Essa abordagem busca transformar o processo educacional em uma experiência mais interativa e dinâmica, incentivando a progressão dos alunos por meio de desafios estruturados e *feedbacks* constantes.

A aplicação de gamificação no ensino de programação é particularmente promissora, pois essa disciplina exige prática contínua e resolução de problemas, características que se alinham bem à lógica dos jogos (SEABORN; FELS, 2015). Em um ambiente gamificado, os alunos podem visualizar seu progresso de maneira mais concreta, receber recompensas por conquistas e ter um *feedback* imediato, facilitando o aprendizado por tentativa e erro. Segundo Richards, Thompson e Graham (2014), um dos princípios fundamentais da gamificação é justamente fornecer retornos constantes ao usuário, permitindo que ele desenvolva um senso de progresso e domínio sobre o conteúdo.

Contudo, apesar dos benefícios potenciais, a efetividade da gamificação no ensino de programação ainda é um campo em desenvolvimento, e há necessidade de mais estudos que analisem seu impacto em longo prazo (DICHEVA et al., 2015). Além disso, a integração dessas mecânicas em plataformas de aprendizagem online ainda é pouco explorada, sendo um aspecto relevante para a evolução das metodologias educacionais (DICHEVA et al., 2015).

Diante desse cenário, desenvolvemos uma plataforma de ensino de programação que incorpora elementos de gamificação aliados à Inteligência Artificial (IA). A abordagem utilizada busca oferecer desafios progressivos, *feedbacks* personalizados e um sistema de evolução por habilidades, permitindo que o aluno visualize seu progresso e receba orientações detalhadas sobre seu desenvolvimento. Com isso, pretende-se não apenas tornar o aprendizado mais envolvente, mas também criar um ambiente que favoreça a autonomia do estudante e sua preparação para desafios do mundo real.

2.3 Ensino de Lógica de Progamação e Algoritmos

O ensino de Lógica de Programação e Algoritmos é a base para a formação de qualquer profissional da área de tecnologia da informação. Compreender esses conceitos é essencial não apenas para escrever códigos funcionais, mas também para desenvolver soluções eficientes, otimizadas e coerentes para problemas complexos.

2.3.1 Lógica de Progamação

A lógica de programação é um conceito fundamental para o desenvolvimento de *softwares*, pois permite estruturar e organizar o raciocínio necessário para a criação de soluções computacionais. Segundo Forbellone e Villar (2005), a lógica de programação envolve o uso de princípios racionais e técnicas que garantem a construção de programas coerentes e eficientes. Assim como a comunicação humana se baseia em idiomas com regras gramaticais e sintáticas específicas, a lógica de programação também segue padrões que permitem expressar instruções de maneira compreensível para os computadores.

Os algoritmos desempenham um papel essencial nesse contexto, pois funcionam como um meio de representar e formalizar um conjunto de instruções que resolvem um problema específico. De acordo com Forbellone e Villar (2005), um mesmo raciocínio lógico pode ser representado em diferentes linguagens de programação, mas sua essência permanece a mesma, independentemente da sintaxe utilizada. Os algoritmos, portanto, são fundamentais para garantir que o raciocínio lógico seja traduzido corretamente em código, evitando ambiguidades e garantindo a execução eficiente das tarefas planejadas.

2.3.2 Algoritmos

Um algoritmo pode ser compreendido como um conjunto de instruções organizadas logicamente para alcançar um determinado objetivo. Segundo Forbellone e Villar (2005), um algoritmo é uma sequência finita de passos que levam à resolução de um problema de forma clara

e estruturada. No cotidiano, esse conceito pode ser exemplificado por processos como o preparo de uma receita de bolo, onde há uma sequência definida de etapas que devem ser seguidas para obter um resultado esperado.

A lógica na construção de algoritmos é essencial para garantir que as instruções sejam ordenadas de maneira coerente, permitindo que, ao serem executadas, produzam sempre o mesmo resultado sob as mesmas condições. Como destacado por Forbellone e Villar (2005), um algoritmo precisa especificar ações de forma precisa e objetiva, partindo de um estado inicial e chegando a um estado final bem definido após um tempo finito de execução. Dessa forma, ele estabelece um padrão de execução, garantindo previsibilidade e eficiência na solução de problemas.

2.4 Inteligência Artificial

Nesse ponto, iremos abordar conceitos que abordam a definição de Inteligência Artificial (IA) apresentando diversas referências.

2.5 Conceitos Fundamentais de Inteligência Artificial

A Inteligência Artificial (IA) pode ser compreendida a partir de quatro abordagens distintas, como apresentado por Russell e Norvig (2013) agir como humanos, pensar como humanos, pensar racionalmente e agir racionalmente. Essa classificação permite compreender diferentes perspectivas adotadas ao longo da história da IA, bem como os objetivos e métodos associados a cada uma.

A primeira abordagem, agir como humanos, está relacionada ao famoso Teste de Turing, proposto por Turing (1950), que avalia a inteligência de uma máquina com base em sua capacidade de interagir com seres humanos de forma indistinguível. Para alcançar esse nível, sistemas precisam incorporar áreas como processamento de linguagem natural, representação de conhecimento, raciocínio automatizado e aprendizado de máquina.

A segunda abordagem, pensar como humanos, envolve a modelagem cognitiva, em que se busca reproduzir os processos mentais humanos por meio de algoritmos computacionais. Essa linha de pesquisa tem interseções com a ciência cognitiva, utilizando métodos experimentais para validar se os modelos computacionais simulam de forma verossímil o comportamento humano (RUSSELL; NORVIG, 2013).

A terceira abordagem, pensar racionalmente, tem como base os princípios da lógica formal, conforme iniciados por Aristóteles. Nessa perspectiva, um sistema inteligente é aquele que realiza inferências corretas, deduzindo conclusões válidas a partir de premissas conhecidas. Embora poderosa, essa abordagem enfrenta limitações práticas, principalmente devido à complexidade computacional envolvida.

A quarta e mais abrangente abordagem, agir racionalmente, define inteligência como a capacidade de tomar decisões que maximizam o resultado esperado, mesmo em cenários de incerteza. Essa perspectiva, baseada no conceito de agente racional, tem ganhado destaque

por sua aplicabilidade em sistemas reais, permitindo o desenvolvimento de agentes autônomos capazes de perceber o ambiente, adaptar-se a mudanças e perseguir objetivos de forma eficiente.

Neste trabalho, a abordagem adotada está alinhada à concepção de agente racional, visto que a plataforma proposta utiliza IA para fornecer *feedbacks* personalizados aos usuários, com base em seus desempenhos, e adaptar os desafios apresentados conforme seu progresso. Essa estratégia visa promover um aprendizado contínuo, motivador e adaptativo, ampliando a eficácia do processo de ensino-aprendizagem.

Além das abordagens clássicas, destaca-se nos últimos anos a ascensão da Inteligência Artificial Generativa (IAG), um subcampo da IA que se concentra na criação autônoma de conteúdos diversos, como textos, imagens, códigos e sons. Essa tecnologia baseia-se principalmente em modelos de aprendizado profundo, como os transformadores, sendo os modelos de linguagem de grande escala (LLMs) os mais representativos, a exemplo do *ChatGPT*, *Gemini*, *Claude* e *LLaMA*. A IAG difere das demais abordagens por não apenas simular comportamentos inteligentes, mas por gerar novos artefatos com base em padrões aprendidos a partir de grandes volumes de dados. Essa capacidade tem implicações significativas na educação, especialmente no ensino de programação, uma vez que permite oferecer sugestões de código, explicações em linguagem natural e *feedbacks* adaptativos com alto grau de personalização. No entanto, também levanta desafios quanto ao uso ético e à dependência excessiva de respostas automatizadas. Segundo Bommasani et al. (2022), os modelos fundacionais impulsionaram essa nova era da IA, tornando-se plataformas versáteis para diversas tarefas cognitivas. Dessa forma, a Inteligência Artificial Generativa representa um avanço relevante na construção de ambientes de aprendizagem mais dinâmicos e eficazes.

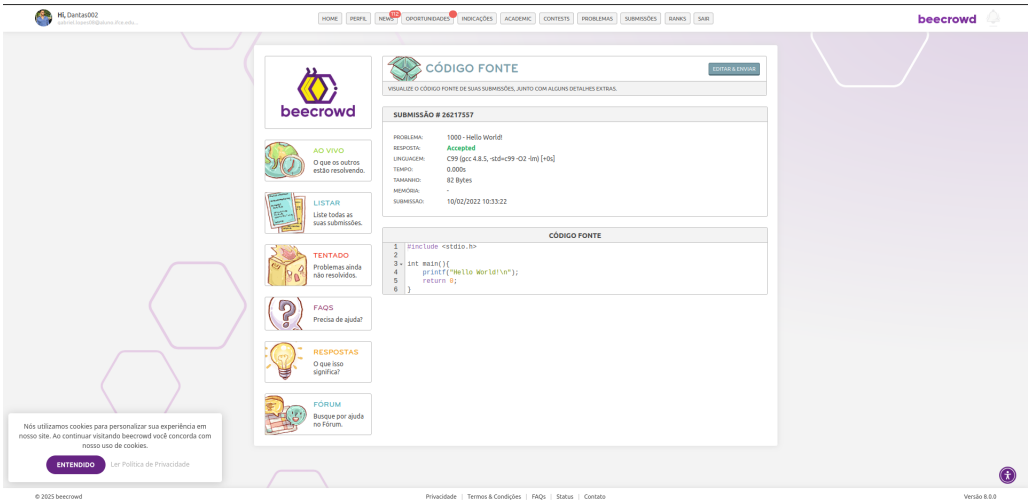
2.6 Juiz *Online* no Ensino de Programação

O ensino de programação exige que os alunos pratiquem a resolução de problemas por meio da implementação de códigos, mas a correção manual dessas atividades pode ser um desafio para professores, especialmente em turmas numerosas. Para lidar com essa questão, os sistemas de Juiz *Online* surgiram como uma alternativa eficiente para a avaliação automática de código-fonte, conforme mostra a Figura 1, onde mostra todas as especificações e condições de submissão, como o tempo de execução e se foi aceito ou não.

De acordo com Francisco, Júnior e Ambrósio (2016), um Juiz Online é um sistema que disponibiliza problemas de programação para os alunos resolverem e submeterem suas soluções na forma de código. O sistema então executa testes automatizados, verificando se a solução submetida gera os resultados esperados para entradas predefinidas. Esses sistemas são amplamente utilizados em competições de programação, como a Maratona de Programação da Sociedade Brasileira de Computação (SBC), e vêm sendo gradualmente adaptados para o ensino de Programação Introdutória (CS1).

Os Juízes Online trazem benefícios significativos, como a agilidade na correção, a possibilidade de *feedback* imediato, e o incentivo à prática contínua por parte dos estudantes.

Figura 1 – Submissão de problema do Beecrowd



Fonte: (BEECROWD, 2025)

No entanto, também apresentam desafios, como a limitação dos *feedbacks* oferecidos, que normalmente apenas indicam se o código está certo ou errado, sem detalhar quais aspectos poderiam ser melhorados.

3 TRABALHOS RELACIONADOS

O trabalho de Marques (2023) desenvolveu uma ferramenta móvel chamada ALPROG, voltada ao ensino de lógica de programação. O objetivo principal do estudo foi criar uma solução acessível para estudantes de diferentes áreas acadêmicas, promovendo o ensino de forma mais interativa e dinâmica. Conforme mostra as Figuras 2, 3 e 4, a ferramenta conta com um aplicativo móvel para aprendizado dos alunos e uma plataforma *web* para que os docentes possam gerenciar conteúdos e acompanhar o progresso dos estudantes.

A pesquisa incluiu uma revisão sistemática da literatura para identificar ferramentas existentes no ensino de lógica de programação e realizou testes com 30 estudantes. A validação da ferramenta foi feita com base no Modelo de Aceitação de Tecnologia (TAM) e no Modelo QUIS, permitindo avaliar a percepção de utilidade e a aceitação dos usuários. Os resultados indicaram que a ferramenta foi bem recebida pelos alunos, melhorando a assimilação dos conteúdos e promovendo maior engajamento no aprendizado.

Enquanto Marques (2023) utiliza um aplicativo móvel para apoiar o ensino de lógica, foi desenvolvido uma plataforma de aprendizagem gamificada com correção automática por IA, permitindo que os alunos não apenas pratiquem lógica de programação, mas também recebam *feedbacks* personalizados sobre seus códigos. A integração da gamificação e da inteligência artificial complementa as abordagens discutidas no estudo de Marques, oferecendo uma solução inovadora para o ensino de programação.

O trabalho de Siqueira (2024) apresenta o desenvolvimento de uma ferramenta como mostra nas Figuras 5, 6 e 7 integrada ao árbitro virtual Dikastis, utilizada para corrigir e fornecer

Figura 2 – Tela de Login

Figura 3 – Categorias

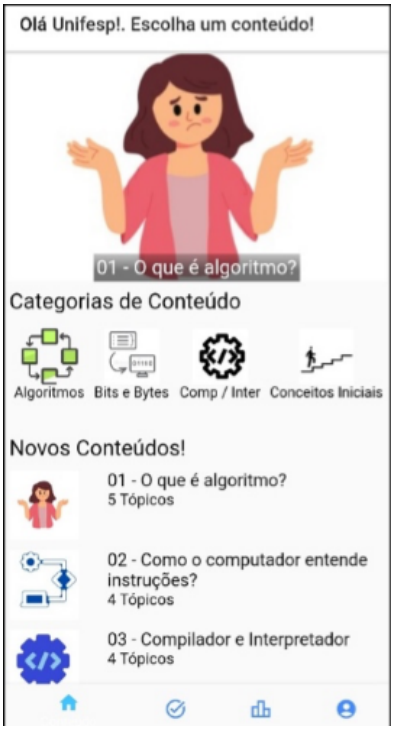


Figura 4 – Conteúdos

Fonte: (MARQUES, 2023)

feedbacks sobre atividades de programação. O principal objetivo do estudo foi aprimorar o processo de ensino-aprendizagem das disciplinas introdutórias de programação, garantindo *feedbacks* mais detalhados e personalizados. A pesquisa destaca a importância do suporte individualizado oferecido pelos monitores e o impacto da ferramenta no engajamento dos estudantes.

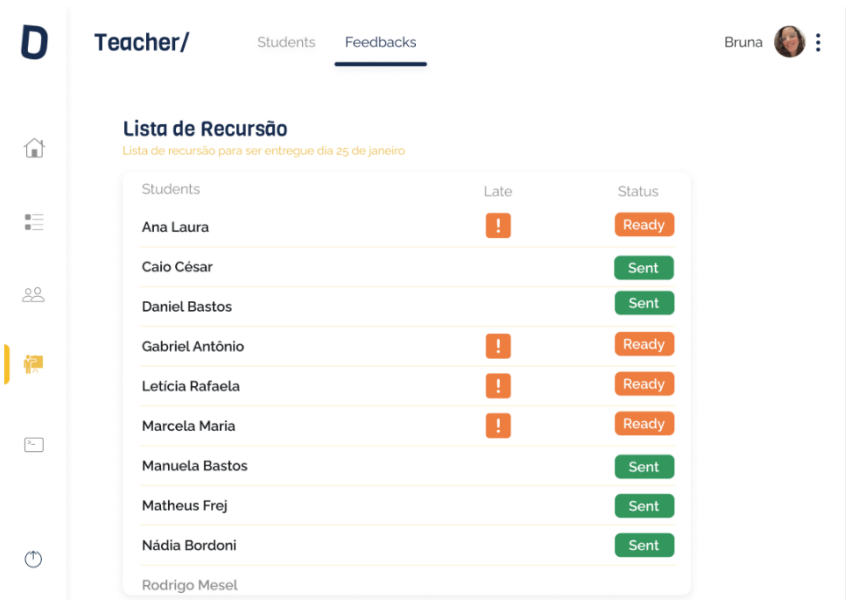
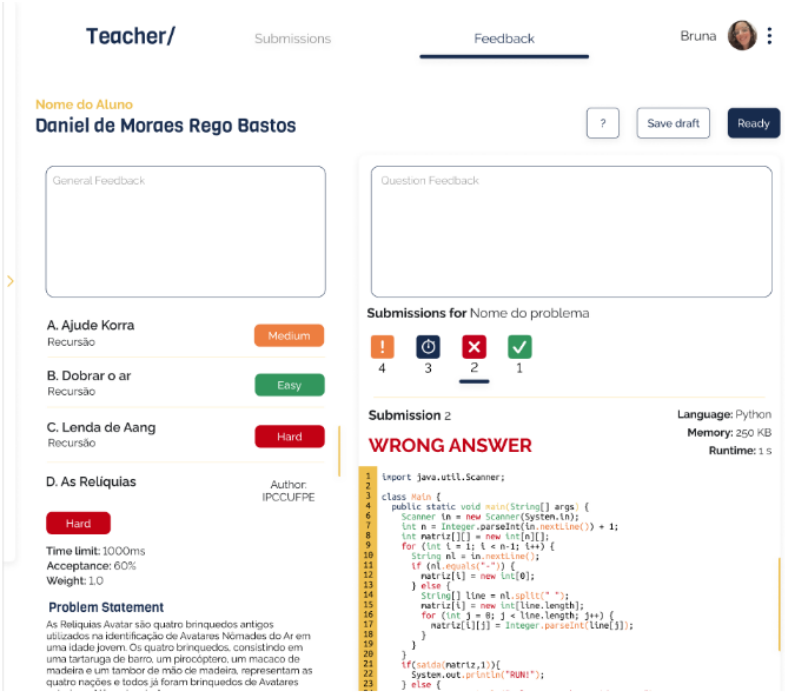


Figura 5 – Listagem de alunos

A autora realizou um estudo de caso com turmas das disciplinas Introdução à Progra-



h

Figura 6 – Escrita de *Feedbacks*

mação e Programação 1 da Universidade Federal de Pernambuco, analisando a eficiência da nova ferramenta em comparação com métodos tradicionais de feedback. Os resultados indicaram que a implementação da solução dentro do Dikastis permitiu maior interação entre alunos e monitores, facilitou o processo de feedback e melhorou a retenção do conhecimento.

Este trabalho se relaciona com o presente estudo ao evidenciar a necessidade de *feedbacks* personalizados no ensino de programação. Enquanto Siqueira (2024) foca na integração de um sistema de *feedbacks* ao árbitro virtual para suporte acadêmico, este projeto expande essa abordagem ao incorporar inteligência artificial para analisar e corrigir códigos automaticamente. Além disso, a proposta deste trabalho adiciona elementos de gamificação, proporcionando uma experiência de aprendizado mais envolvente e adaptativa para os estudantes.

O trabalho de Resende (2024) foca na análise da qualidade do código dentro de um ambiente acadêmico, este projeto propõe um sistema que, além de corrigir automaticamente as soluções dos alunos, utiliza inteligência artificial para fornecer *feedbacks* personalizados e gamificação para manter os alunos engajados. Ambas as abordagens compartilham o objetivo de melhorar a experiência de aprendizado, mas a proposta deste trabalho amplia a metodologia ao integrar mecanismos de motivação e evolução adaptativa para os estudantes.

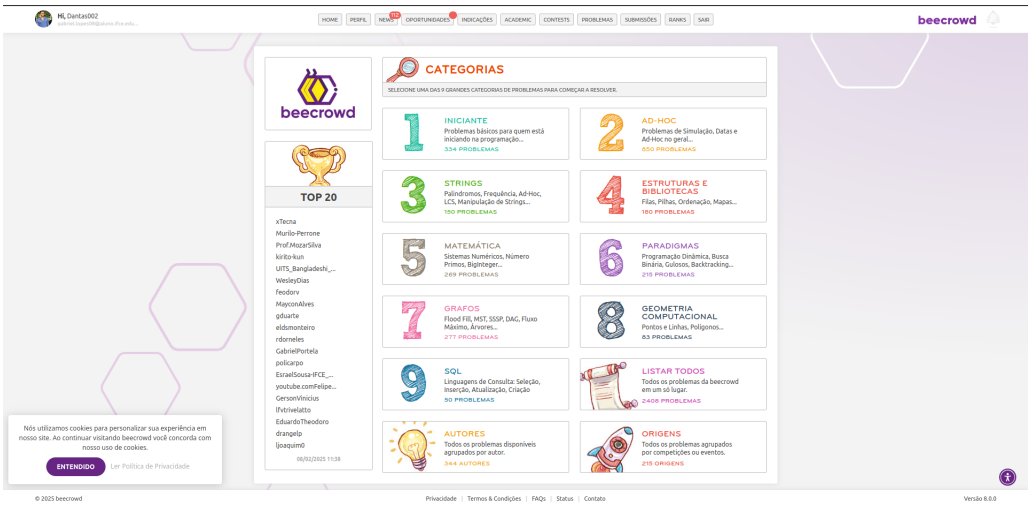
Um exemplo representativo de plataforma também voltada ao ensino de programação é o *Beecrowd*, ambiente online que disponibiliza exercícios e desafios de lógica e algoritmos para que estudantes e profissionais possam praticar suas habilidades em diversas linguagens de programação. A Figura 8 ilustra a interface de categorias de problemas disponibilizadas pela plataforma.

Na Figura 9, observa-se a tela exibida ao selecionar um problema. A interface apresenta

Figura 7 – *Feedback* concluído

Fonte: (SIQUEIRA, 2024)

Figura 8 – Tela de categoria de problemas do *Beecrowd*

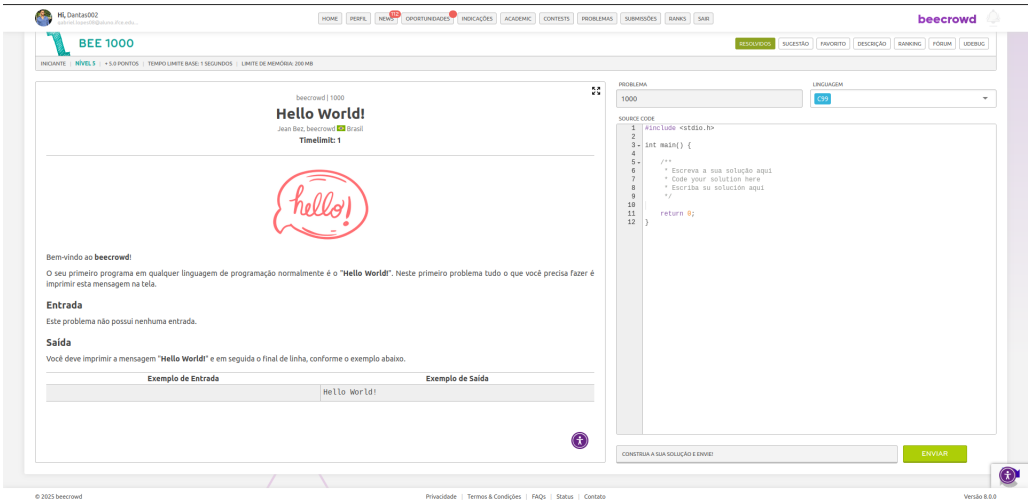


Fonte: (BEECROWD, 2025)

o enunciado do exercício, seguido das especificações de entrada e saída esperadas. Abaixo, há um editor de código que permite ao usuário implementar sua solução e submetê-la diretamente na plataforma para validação automática.

Outro exemplo relevante é a plataforma *Duolingo*, como mostra a 10 amplamente reconhecida por seu uso eficaz de gamificação no processo de ensino-aprendizagem de idiomas. A plataforma adota elementos como recompensas, níveis, desafios diários e metas de progresso, proporcionando uma experiência altamente interativa e motivadora. Seu sucesso serve como inspiração direta para o presente trabalho, especialmente no que se refere ao uso de estratégias motivacionais para engajar estudantes no desenvolvimento de habilidades cognitivas. A incorpo-

Figura 9 – Tela do exercício do *Beecrowd*



Fonte: (BEECROWD, 2025)

ração desses elementos na plataforma proposta visa estimular a continuidade do aprendizado e promover maior retenção dos conteúdos abordados (DUOLINGO, 2011).

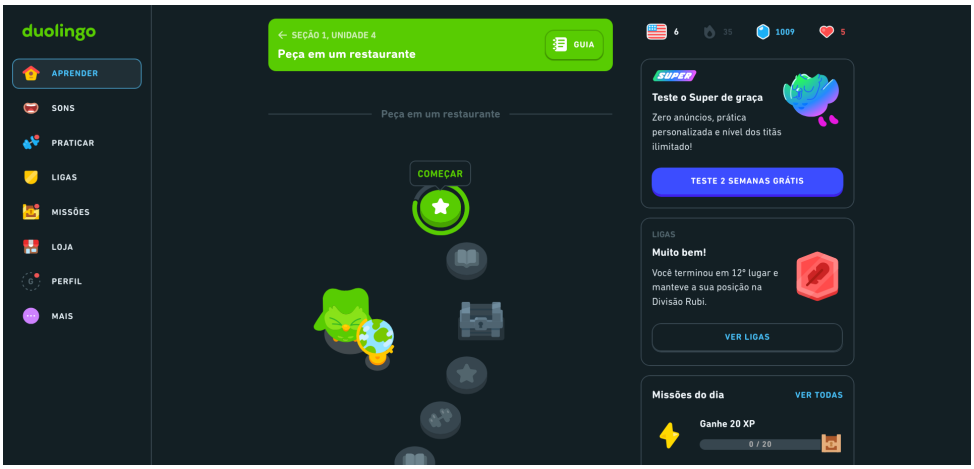


Figura 10 – Interface da plataforma Duolingo, destacando elementos de gamificação.

Fonte: (DUOLINGO, 2011)

Outra fonte de inspiração significativa é a plataforma *LeetCode*, como mostra na 11 amplamente utilizada por desenvolvedores para a prática de algoritmos e preparação para entrevistas técnicas. A estrutura de exercícios categorizados por nível de dificuldade, o suporte a múltiplas linguagens de programação, bem como o sistema de submissão com correção automática e retorno imediato de *feedbacks*, foram características que influenciaram diretamente o desenvolvimento da plataforma descrita neste trabalho. Tais elementos permitem uma aprendizagem prática e orientada por desempenho, aspectos também incorporados neste projeto (LEETCODE, 2015).

4 METODOLOGIA

Nesta seção, são apresentadas as três etapas do desenvolvimento da plataforma de aprendizagem gamificada com correção automatizada por inteligência artificial. As Figuras 12

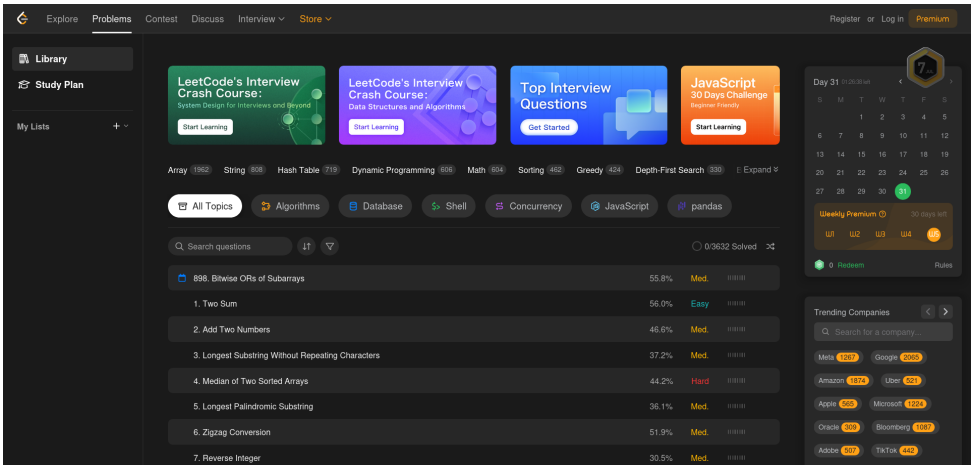


Figura 11 – Interface da plataforma *Leetcode*.

Fonte: (LEETCODE, 2015)

e 13 apresentam a página inicial da plataforma *Educ-dev*. Nelas, são destacados os principais diferenciais da aplicação, como o assistente com IA personalizada, o sistema de correção inteligente, dicas contextuais e um mentor virtual disponível 24/7. A segunda seção da tela reforça os pilares da plataforma: personalização, prática adaptada e comunidade inteligente, evidenciando o compromisso com uma experiência de aprendizado moderna e eficaz.

Figura 12 – Tela inicial da plataforma Educ-dev com destaques das funcionalidades principais

A primeira etapa aborda a definição dos requisitos e modelagem do sistema. A segunda etapa detalha o desenvolvimento e a arquitetura da plataforma. Por fim, a terceira etapa descreve a avaliação da plataforma e as métricas de uso.

4.1 Etapa 1 – Definição de Requisitos e Modelagem do Sistema

A primeira etapa consistiu na identificação e levantamento dos requisitos essenciais para o público-alvo, composto por estudantes e docentes de programação. Foram definidos requisitos como execução de códigos em múltiplas linguagens, gestão de conteúdo educacional, acompanhamento do progresso do usuário e *feedback* automatizado por inteligência artificial. A modelagem do sistema foi realizada utilizando princípios de *Domain-Driven Design (DDD)*,

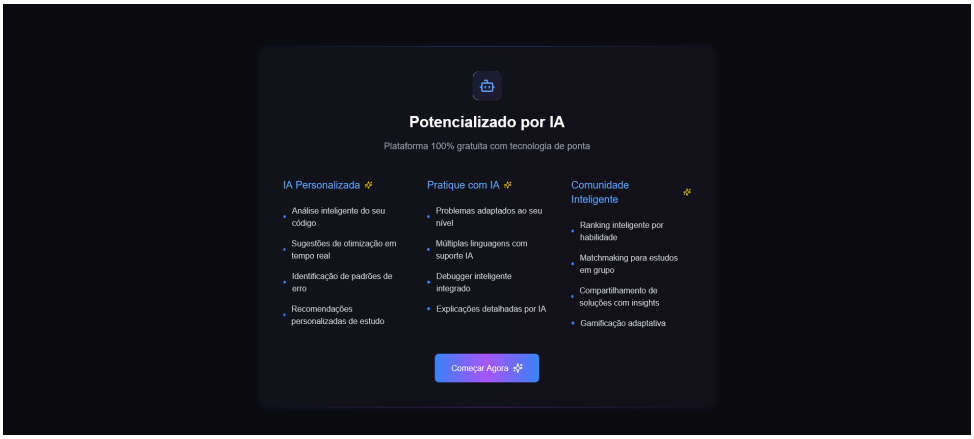


Figura 13 – Seção de apresentação dos recursos avançados com suporte de IA e gamificação

promovendo uma separação clara entre os domínios de negócio, aplicação e infraestrutura (EVANS, 2003). As interfaces foram planejadas e desenvolvidas diretamente no código, seguindo princípios de design responsivo e experiência do usuário.

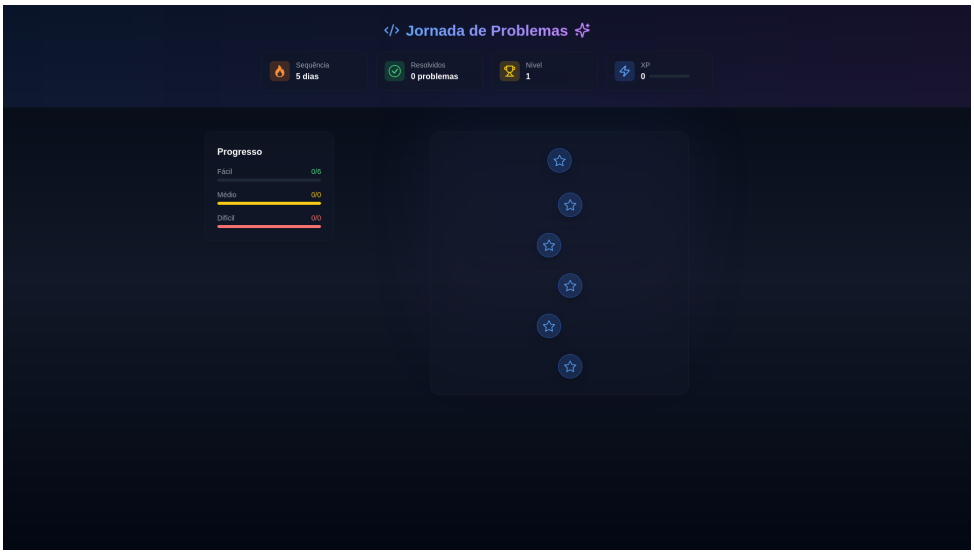


Figura 14 – Tela de seleção de problemas com estatísticas do usuário

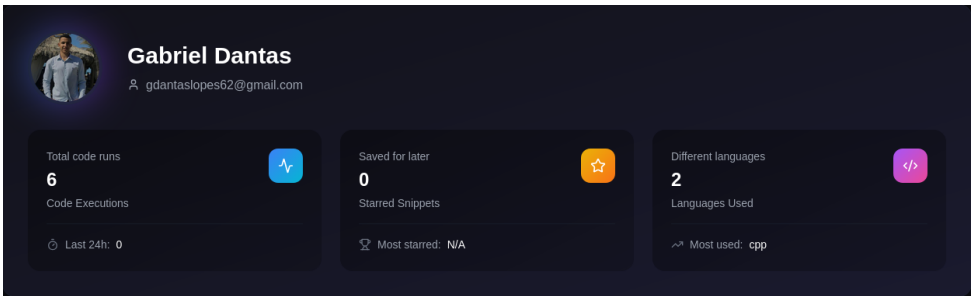


Figura 15 – Tela de perfil do usuário com estatísticas e conquistas

A Figura 14 apresenta a tela de seleção de problemas. Nela, o usuário pode visualizar todos os desafios disponíveis, organizados por nível de dificuldade, com inspiração da plataforma **??**). No topo, são exibidas estatísticas como sequência de resolução, total de problemas

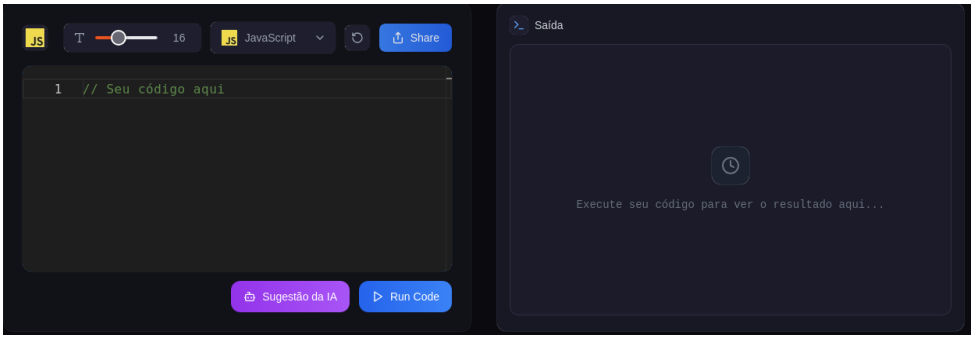


Figura 16 – Editor de código integrado à plataforma

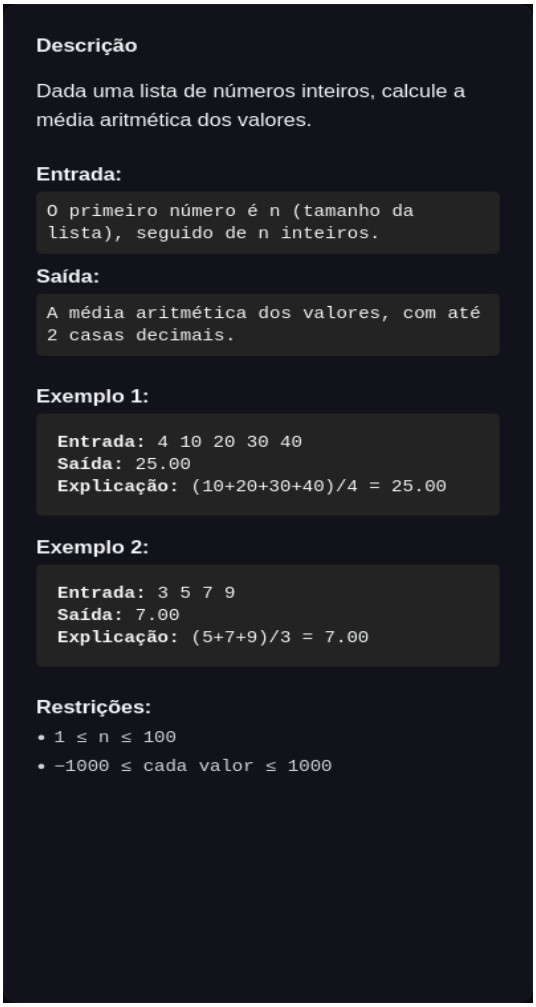


Figura 17 – Tela de problema específico com enunciado, exemplos e restrições

Figura 18 – Modal do assistente de IA com *feedback* detalhado

Fonte: Elaborado pelo autor.

resolvidos, nível atual e experiência (XP) acumulada. Na lateral, há um painel de progresso que mostra o desempenho do usuário em cada categoria de dificuldade, incentivando o avanço contínuo na plataforma.

A Figura 15 mostra a tela de perfil do usuário. Nessa seção, o estudante pode acompanhar seu histórico de resoluções, conquistas desbloqueadas e demais estatísticas de desempenho, promovendo o engajamento e a motivação para o aprendizado contínuo.

A Figura 16 apresenta o editor de código da plataforma. O editor permite ao usuário escrever, executar e testar suas soluções diretamente no navegador, com suporte as linguagens de programação como *C/C++*, *Python*, *Java* e *JavaScript*. Além disso, há opções para selecionar a linguagem, ajustar o tamanho da fonte, compartilhar o código e receber sugestões automáticas de melhoria.

A Figura 17 mostra a tela de resolução de um problema específico. O usuário tem acesso ao enunciado do desafio, exemplos de entrada e saída, explicações detalhadas e as restrições do problema. Essas informações auxiliam o estudante a compreender completamente o que é solicitado antes de iniciar a implementação da solução como é feito na plataforma **??**) onde foi uma outra inspiração na construção dessa tela.

A Figura 18 exibe o modal do assistente de IA. Por meio desse recurso, o usuário pode interagir com um assistente inteligente que fornece *feedback* detalhado sobre o código submetido, sugerindo melhorias, explicando conceitos importantes e fazendo perguntas para estimular o raciocínio do estudante, sem entregar a solução pronta e oferecendo sugestões explicando cada ponto de melhoria ou ideias que ajudam o aluno no autoquestionamento para aprender e resolver o problema.

As Figuras 19 a 22 detalham os diferentes tipos de *feedback* fornecidos pelo assistente de IA. A Figura 19 apresenta o menu de opções disponíveis para o usuário. A Figura 20 ilustra a resposta da IA quando solicitada a solução direta, reforçando a abordagem focada no aprendizado. A Figura 21 demonstra a explicação de conceitos importantes e a formulação de perguntas para estimular a reflexão do estudante. Por fim, a Figura 22 exibe dicas de estudo e alertas sobre pontos de atenção no código submetido.

Figura 19 – Blocos de *feedback* com cores diferentes do assistente de IA.

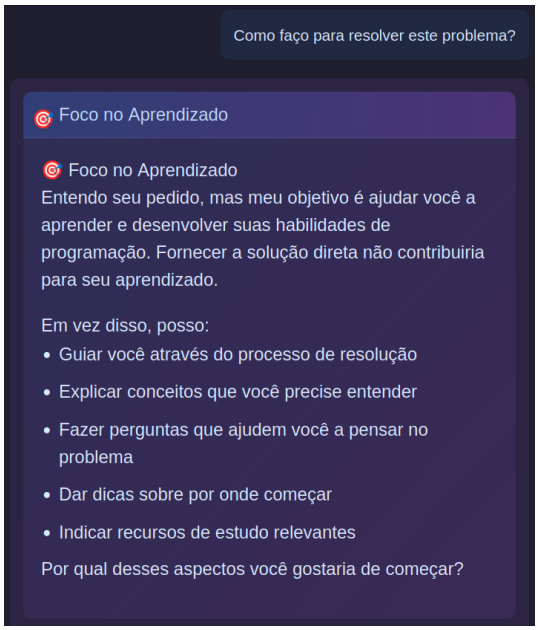


Figura 20 – *Feedback* com foco no aprendizado, evitando soluções diretas.

Fonte: Elaborado pelo autor.

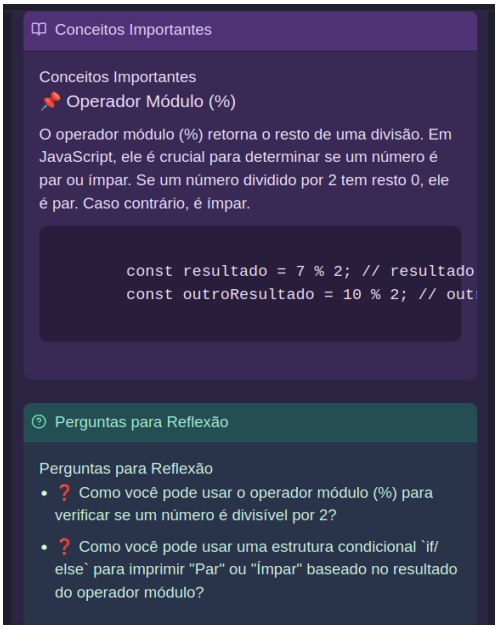


Figura 21 – *Feedback* com conceitos importantes e perguntas para reflexão.

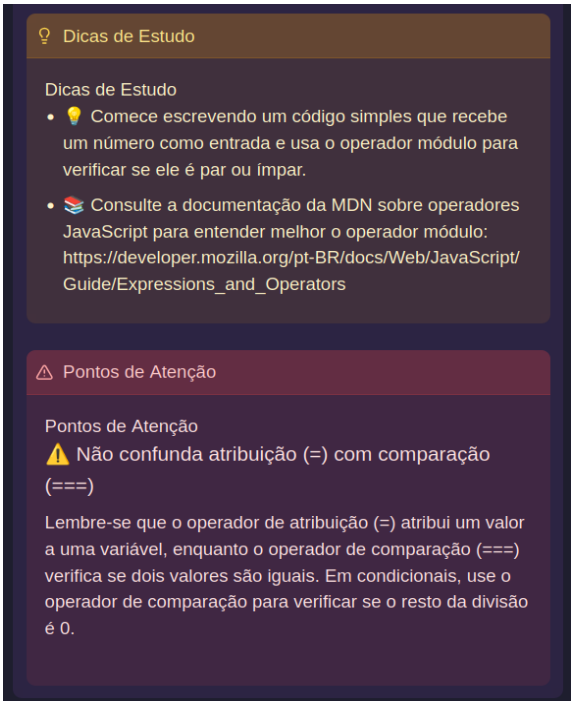


Figura 22 – *Feedback* com dicas de estudo e pontos de atenção sobre o código.

Fonte: Elaborado pelo autor.

A Tabela 1 apresenta um resumo dos principais requisitos funcionais definidos para a plataforma.

Tabela 1 – Requisitos funcionais da plataforma

Requisito	**Descrição**
Execução de códigos	Suporte a múltiplas linguagens e execução segura
Gestão de problemas	Cadastro, exibição e resolução de desafios
Feedback por IA	Sugestões e explicações automáticas
Gestão de usuários	Perfis, autenticação e progresso
Gamificação	Conquistas e XP

Os problemas exibidos na plataforma, como ilustrado na Figura 17, foram previamente cadastrados diretamente no banco de dados por meio de scripts internos de desenvolvimento, sem o uso de uma interface gráfica voltada para o gerenciamento por docentes. Embora a arquitetura do sistema tenha sido projetada para, futuramente, permitir o acesso de professores com perfis específicos para a criação e curadoria de exercícios, essa funcionalidade ainda não foi implementada devido às limitações de tempo durante o desenvolvimento da plataforma, mas será aplicado em trabalhos futuros. Assim, os dados apresentados — como enunciado, exemplos, restrições e gabaritos esperados — foram alimentados de forma manual e centralizada, com o objetivo de viabilizar testes e validações da plataforma durante a fase de prototipagem.

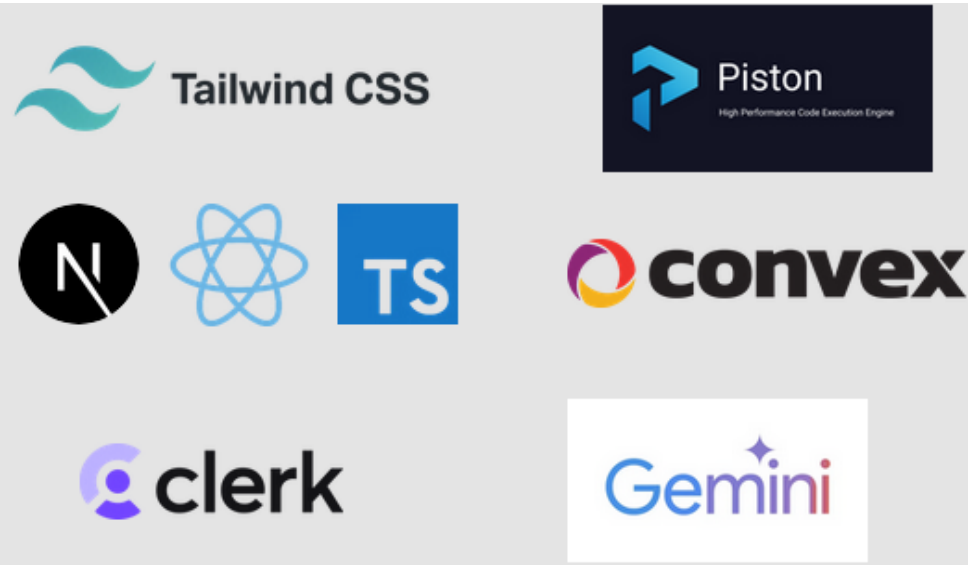


Figura 23 – Principais tecnologias utilizadas no desenvolvimento da plataforma Educ-dev.

4.2 Etapa 2 – Desenvolvimento e Arquitetura da Plataforma

O desenvolvimento da plataforma foi dividido em *Frontend* e *Backend*, visando modularidade e escalabilidade. A arquitetura geral é ilustrada na Figura 25. A Figura 23 apresenta uma visão geral das principais tecnologias empregadas, como *Next.js*, *React*, *Tailwind CSS*, *Piston*, *Convex*, *Clerk* e *Gemini API*, que compõem a arquitetura do sistema.

4.2.1 Frontend

O *Frontend* foi desenvolvido utilizando **Meta (2011)** com **Vercel (2016)**, proporcionando renderização eficiente e navegação fluida. A estilização foi realizada com **Tailwind Labs (2017)**, garantindo responsividade e design moderno. Os principais componentes implementados incluem:

- **Editor de código:** Baseado no *Microsoft (2015)*, com suporte a múltiplas linguagens, ajuste de fonte, seleção de tema e compartilhamento de código.
- **Painel de saída:** Exibe resultados da execução, erros e *feedbacks* da IA de forma clara e interativa.
- **Painel de problemas:** Lista desafios categorizados por dificuldade e estatísticas de progresso.
- **Assistente de IA:** Modal de chat integrado, permitindo interação direta com o modelo de IA para dúvidas e sugestões.
- **Gamificação:** Exibição de XP, nível, sequência de resolução e conquistas.

As Figuras 14 e 24 ilustram as principais telas do *Frontend*.

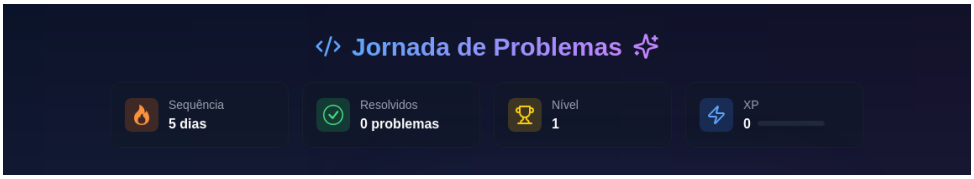


Figura 24 – Tela de exibição de XP, nível, sequência de resolução e conquistas

Fonte: Elaborado pelo autor.

4.2.2 Backend

O *Backend* foi implementado em *Node.js* utilizando a plataforma **Convex, Inc. (2024)**, que oferece funções *serverless* para manipulação de dados, autenticação e integração com *APIs* externas. O banco de dados integrado ao *Convex, Inc. (2024)* armazena informações de usuários, problemas, submissões e progresso. Para a autenticação de usuários, foi integrada a ferramenta **clerk**, que fornece uma solução completa de autenticação baseada em identidade como serviço. O Clerk permite o gerenciamento seguro de sessões, controle de acesso e integração com provedores externos, como Google e GitHub. Na plataforma, sua principal função é garantir que apenas usuários autenticados possam acessar funcionalidades como submissão de códigos, acompanhamento de progresso e interação com o assistente de IA (Clerk Inc., 2024). A execução e correção automática dos códigos são realizadas por meio de integração com o *Google Gemini API*. O fluxo de submissão e *feedback* é o seguinte:

1. O usuário submete o código pelo editor.
2. O *backend* executa o código e valida os testes.
3. O código, o enunciado do problema e o histórico de interações são enviados à *API* de IA.
4. A IA retorna *feedback* detalhado, sugestões de melhoria, explicações de conceitos e dicas de estudo.
5. O *feedback* é apresentado ao usuário de forma visual e interativa.

Para garantir que o assistente de IA respeitasse princípios pedagógicos e não entregasse diretamente a solução dos exercícios, foi elaborado um *prompt* cuidadosamente estruturado, enviado junto aos dados de cada submissão. Esse prompt inclui instruções explícitas sobre as condutas proibidas (como fornecer respostas diretas ou corrigir totalmente o código) e orientações claras sobre como auxiliar o estudante de forma educativa. Entre as diretrizes estão: explicar conceitos relevantes, sugerir caminhos de raciocínio, apontar erros sem oferecer a correção e fazer perguntas que estimulem a reflexão. O conteúdo é formatado em HTML e adaptado à linguagem de programação escolhida, permitindo uma apresentação didática, interativa e padronizada na interface da plataforma. Dessa forma, o modelo de IA atua como um tutor inteligente, guiando o processo de aprendizagem sem comprometer o desafio proposto.

O sistema de juiz online desenvolvido nesta plataforma funciona através de um mecanismo de execução e validação automatizada de código. Quando o aluno submete sua solução,

o sistema utiliza uma *Application Programming Interface (API)* externa *Engineer Man (2024)* para executar o código em um ambiente isolado e seguro. A execução dos códigos submetidos pelos estudantes é realizada por meio da integração com o *Engineer Man (2024)*, uma *engine* de execução de código de uso geral, de alto desempenho e projetada para lidar com código potencialmente não confiável de maneira segura e eficiente.

O *Piston* é a principal ferramenta utilizada no *backend* da aplicação para compilar e executar os códigos em tempo real, sendo compatível com dezenas de linguagens de programação. Quando um estudante submete uma solução para um problema proposto, a plataforma envia uma requisição HTTP *(Hypertext Transfer Protocol)* para a *API* do *Engineer Man (2024)*, informando a linguagem, a versão desejada e o conteúdo do código. A *API* executa o código dentro de um ambiente isolado, criado via *Docker* e *sandboxing* com uso de *namespaces* Linux, usuários não privilegiados e *cgroups*, retornando uma resposta estruturada contendo a saída padrão, erros de execução, tempo de *CPU*, uso de memória e códigos de status. Com base na comparação entre a saída produzida e a saída esperada, a aplicação determina se a resposta está correta, funcionando como um juiz online. A robustez do *Piston* permite isolar completamente as execuções, garantindo segurança, desempenho e confiabilidade no ambiente educacional desenvolvido (Engineer Man, 2024).

O processo de avaliação é baseado na comparação entre a saída gerada pelo código do aluno e as saídas esperadas predefinidas para cada caso de teste. Cada problema possui exemplos de entrada e saída que servem como testes automáticos, permitindo que o sistema verifique se a solução está correta de forma objetiva e rápida. O *feedback* é apresentado de maneira visual e detalhada, mostrando ao aluno exatamente quais testes passaram e quais falharam, incluindo as entradas utilizadas, saídas esperadas e saídas geradas. Quando todos os testes são aprovados, o sistema automaticamente marca o problema como resolvido e permite o acesso ao próximo exercício. Esta abordagem representa uma evolução dos juízes online tradicionais, pois além da correção automática, oferece orientação educacional através de inteligência artificial, proporcionando um ambiente de aprendizado mais completo e interativo para o desenvolvimento de habilidades em programação.

A Figura 25 ilustra o fluxo de comunicação entre *frontend*, *backend* e IA.

4.3 Etapa 3 – Avaliação da Plataforma e Métricas de Uso

A avaliação da plataforma foi realizada por meio de testes com usuários reais, que utilizaram a plataforma para resolver desafios de programação e interagir com o assistente de IA. Foram coletadas métricas como:

- Número de usuários cadastrados e ativos.
- Quantidade de problemas resolvidos por usuário.
- Frequência de uso das funcionalidades de *feedback* por IA.

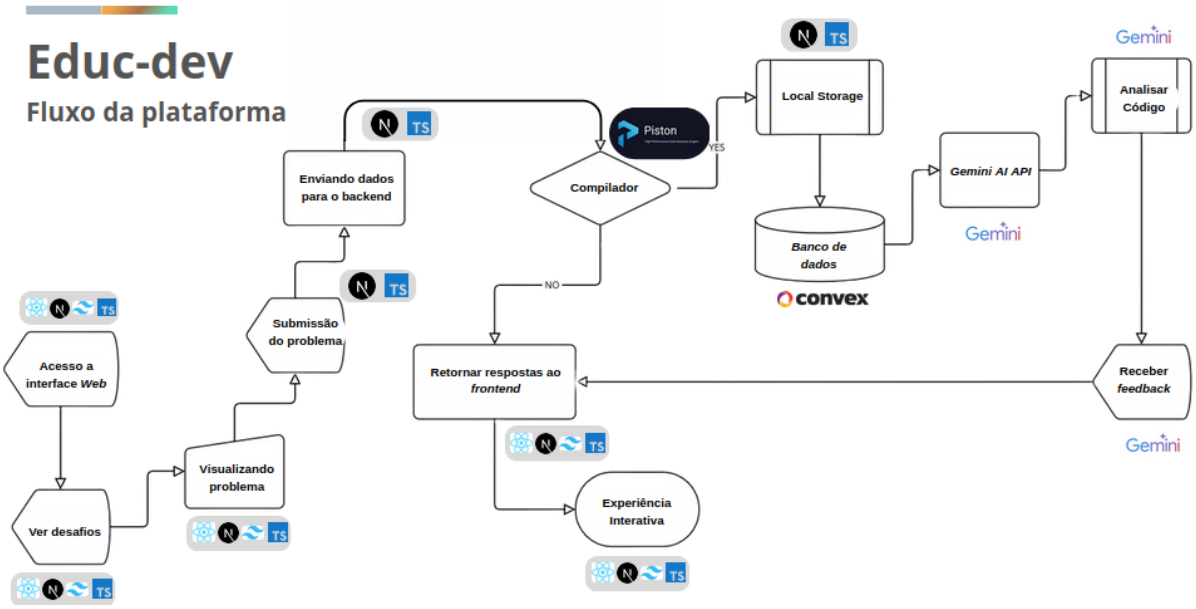


Figura 25 – Fluxo de comunicação entre *frontend*, *backend* e IA

Fonte: Elaborado pelo autor.

- Evolução do ranking e conquistas desbloqueadas.

A avaliação através dos formulários respondidos pelos alunos permitiu validar a usabilidade, a efetividade do *feedback* automatizado e o impacto da gamificação no engajamento dos estudantes. A próxima seção apresenta os principais resultados obtidos durante a avaliação da plataforma.

5 RESULTADOS

Nesta seção, são apresentados os dados coletados por meio de um formulário aplicado a uma turma da disciplina de Introdução à Programação, do curso de Ciência da Computação no Instituto Federal do Ceará - CAMPUS Aracati. A avaliação foi realizada no mês de junho de 2025, contando com a participação de 14 estudantes da turma. Antes da aplicação do formulário, os alunos foram orientados a acessar e utilizar a plataforma *Educ-Dev* por meio de um *link* disponibilizado em sala. A atividade consistiu em explorar a plataforma livremente durante uma aula prática, resolvendo pelo menos 3 desafios de programação da trilha introdutória e testando o assistente de *feedback* com inteligência artificial. Após a experiência, os alunos responderam ao questionário. O objetivo foi avaliar aspectos como usabilidade, clareza dos desafios, engajamento com a gamificação, utilidade do *feedback* da IA e impacto na aprendizagem. A Tabela 2 mostra a média das notas atribuídas pelos usuários em cada critério avaliado, numa escala de 1 (muito ruim) a 5 (muito bom).

A seguir, são apresentados os gráficos referentes às principais questões avaliadas no formulário:

De maneira geral, os dados obtidos por meio do formulário indicam uma boa aceitação da plataforma pelos alunos. No entanto, apesar das notas elevadas em critérios como "reco-

Tabela 2 – Média das avaliações dos usuários

Critério Avaliado	**Média**
Facilidade de uso da plataforma	4.31
Adequação dos desafios ao nível do usuário	4.69
Engajamento com a gamificação	4.25
Utilidade do feedback da IA	4.56
Aprendizado prático proporcionado	4.44
Motivação após uso da plataforma	4.38
Funcionamento técnico da aplicação	4.63
Recomendação da plataforma	4.88
Nota geral da experiência (1 a 10)	**8.88**

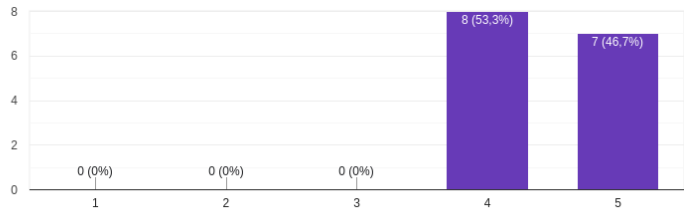


Figura 26 – Avaliação dos usuários sobre a facilidade de uso da plataforma.

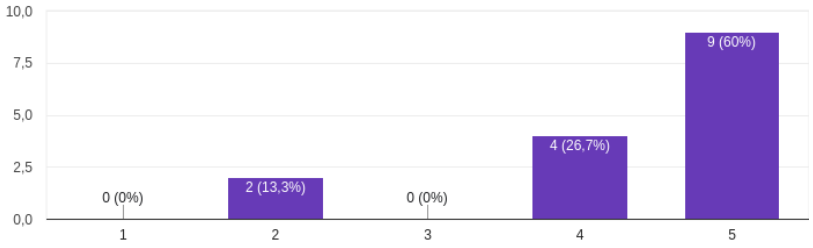


Figura 27 – Avaliação dos usuários sobre a utilidade do *feedback* da IA.

mendação da plataforma"(4.88) e "adequação dos desafios"(4.69), foi possível observar que o "engajamento com a gamificação"(4.25) apresentou a média mais baixa entre os itens avaliados, o que sugere oportunidades de aprimoramento nessa dimensão. As respostas abertas também apontaram sugestões relevantes, como a possibilidade de adicionar campos para anotações e refinar a interface visual. A experiência de aplicar o formulário demonstrou que os estudantes compreenderam as funcionalidades e utilizaram a plataforma de forma autônoma, validando o modelo proposto. Contudo, futuras avaliações com um público maior e por períodos mais longos podem oferecer análises ainda mais robustas sobre a eficácia da solução.

Além dos dados numéricos, as respostas abertas mostraram aspectos valorizados pelos estudantes, como a facilidade de uso e a interatividade da plataforma (Figura 26), além da utilidade do *feedback* da IA (Figura 27). Algumas sugestões incluíram melhorias na interface e adição de campos de anotação.

Os resultados demonstram que a plataforma Educ-Dev cumpriu seu propósito inicial de auxiliar na aprendizagem prática e personalizada de programação, sendo bem recebida pela

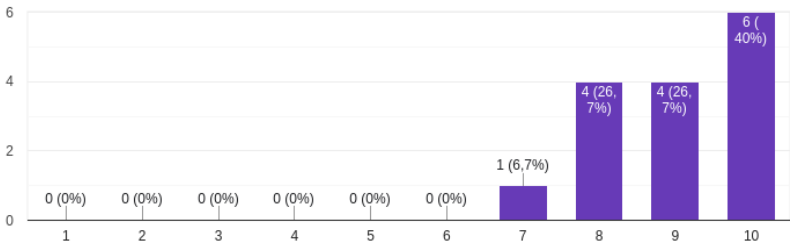


Figura 28 – Nota geral atribuída pelos usuários à experiência na plataforma.

maioria dos usuários testados.

6 CONCLUSÃO

Este trabalho apresentou o desenvolvimento de uma plataforma gamificada para o ensino de programação, integrando correção automatizada por Inteligência Artificial. A pesquisa envolveu desde a definição dos requisitos e modelagem do sistema até a implementação de funcionalidades que visam facilitar o aprendizado prático, promover o engajamento dos estudantes e oferecer feedback personalizado. A aplicação foi disponibilizada para uma turma de Introdução à Programação, sendo avaliada positivamente quanto à usabilidade, utilidade do feedback da IA e impacto da gamificação no processo de aprendizagem. Espera-se que esta plataforma contribua para a modernização do ensino de programação, servindo de referência para o desenvolvimento de novas soluções educacionais que aliem tecnologia, interatividade e personalização do aprendizado. Como trabalhos futuros, pretende-se expandir a plataforma com a oferta de conteúdos voltados ao ensino de tecnologias como *React.js*, *Next.js*, *TailwindCSS*, *convex*, e etc consolidando-a como uma ferramenta educacional completa. Além disso, propõe-se a utilização da plataforma como um sistema de avaliação de desempenho discente, permitindo a geração de rankings que possam auxiliar instituições na identificação de estudantes com maior destaque, contribuindo em processos de concessão de bolsas de estudo e outras iniciativas acadêmicas. Além disso, planeja-se realizar avaliações com um número maior e mais diversificado de usuários, bem como integrar recursos colaborativos e de acompanhamento docente, ampliando o potencial de uso da plataforma em diferentes contextos educacionais.

REFERÊNCIAS

BEECROWD. **The Ecosystem where Innovative Companies find the best Tech Talent**. 2025. Disponível em: <https://beecrowd.com/>.

BOMMASANI, R. et al. On the opportunities and risks of foundation models. **arXiv preprint arXiv:2201.06939**, 2022.

BRAZIL, A.; BARUQUE, L. Gamificação aplicada na graduação em jogos digitais. ***Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)*** , p. 677, 2011. Doi: 10.5753/cbie.sbie.2015.677.

BRITO, A.; MADEIRA, C. Xp & skills: gamificando o processo de ensino de introdução a programação. ***Anais dos Workshops do Congresso Brasileiro de Informática na Educação***, 2015. Doi: 10.5753/cbie.wcbie.2015.1124,.

Clerk Inc. **Clerk – Authentication and User Management for Modern Applications**. 2024. <https://clerk.com>. Acesso em: 29 jul. 2025.

Convex, Inc. **Convex – The backend for the modern web**. 2024. Acessado em: 29 jul. 2025. Disponível em: <https://www.convex.dev>.

DETERDING, S. et al. From game design elements to gamefulness: defining "gamification". ***Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments***, p. 9–15, 2011. Doi: 10.1145/2181037.2181040.

DICHEVA, D. et al. Gamification in education: A systematic mapping study. **Educational Technology & Society**, 2015.

DUOLINGO. **Duolingo**. 2011. <https://www.duolingo.com/>.

Engineer Man. **Piston - A high performance general purpose code execution engine**. 2024. <https://github.com/engineer-man/piston>. Acesso em: 21 jul. 2025.

EVANS, E. **Domain-Driven Design: Tackling Complexity in the Heart of Software**. Boston: Addison-Wesley, 2003.

FORBELLONE; VILLAR, A. L. **Lógica de Programação - 4.ed.: A Construção de Algoritmos e Estruturas de Dados com Aplicações em Python**. [S.l.]: São Paulo: Makron Books, 2005.

FRANCISCO, R. E.; JúNIOR, C. X. P.; AMBRóSIO, A. P. Juiz online no ensino de programação introdutória - uma revisão sistemática da literatura. In: **Anais do XXVII Simpósio Brasileiro de Informática na Educação (SBIE 2016)**. [S.l.: s.n.], 2016.

GOMES, A.; MENDES, A.; (2014). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. ***Frontiers in Education Conference (FIE), 2014 IEEE***, IEEE, p. 1–8, 2014. Doi: 10.1109/FIE.2014.7044086.

GOOGLE. **Gemini is here: Google's most capable AI model**. 2023. Acesso em: jun. 2025. Disponível em: <https://blog.google/technology/ai/google-gemini-ai/>.

JR, D. P.; CORTELAZZO, A. L. Sala de aula invertida, ambientes de aprendizagem e educação online: a junção de três métodos para potencialização do ensino de algoritmos. **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**, 2015.

LAHTINEN, E. et al. A study of the difficulties of novice programmers. ***Acm Sigcse Bulletin and (Vol. 37, No. 3, pp. 14-18)***, 2005. Doi: 10.1145/1067445.1067453.

LEETCODE. **LeetCode - Improve Coding Skills**. 2015. <https://leetcode.com/>.

MARQUES, M. F. P. Trabalho de Conclusão de Curso, **Desenvolvimento de uma ferramenta educacional baseada em aprendizagem móvel para apoio ao ensino de lógica de programação**. Cidade - Estado: [s.n.], 2023. Orientador: PROF. DRA. REGINA CÉLIA COELHO.

MEIRELES; FERNANDES, T. Desenvolvimento de um objeto de aprendizagem de matemática usando o scratch: da elaboração à construção. ***Curitiba, 2017. 217 f.Dissertação (Mestrado em Educação) –Universidade Federal do Paraná, Curitiba, 2017***, 2017.

Meta. **React – A JavaScript library for building user interfaces**. 2011. Acessado em: 29 jul. 2025. Disponível em: <https://react.dev>.

Microsoft. **Monaco Editor – The code editor that powers VS Code**. 2015. Acessado em: 29 jul. 2025. Disponível em: <https://microsoft.github.io/monaco-editor>.

MICROSOFT. **GitHub Copilot – Your AI pair programmer**. 2023. Acesso em: jun. 2025. Disponível em: <https://github.com/features/copilot>.

OPENAI. **Introducing ChatGPT**. 2022. Acesso em: jun. 2025. Disponível em: <https://openai.com/blog/chatgpt>.

RESENDE, E. S. Trabalho de Conclusão de Curso, **The Avaliator: uma ferramenta web para avaliação de qualidade de código-fonte de aprendizes de programação em Python**. Uberlândia - MG: [s.n.], 2024.

RICHARDS, C.; THOMPSON, C. W.; GRAHAM, N. Beyond designing for motivation: the importance of context in gamification. **in Proceedings of the First ACM SIGCHI Annual Symposium on Computer-human Interaction in Play, New York, NY, USA**, 2014.

RUSSELL, S. J.; NORVIG, P. **Inteligência Artificial**. 3. ed. Rio de Janeiro: Elsevier, 2013. Tradução de Regina Célia Simille. ISBN 9788535237016.

SEABORN, K.; FELS, D. I. Beyond designing for motivation: the importance of context in gamification. **International Journal of Human-Computer Studies**, 2015.

SIQUEIRA, B. A. W. de. Trabalho de Graduação, **Uma ferramenta para a elaboração de feedbacks de atividades iniciais de programação integrada a um árbitro virtual**. Recife - PE: [s.n.], 2024. Orientador: Ricardo Massa Ferreira Lima.

Tailwind Labs. **Tailwind CSS – Rapidly build modern websites without ever leaving your HTML**. 2017. Acessado em: 29 jul. 2025. Disponível em: <https://tailwindcss.com>.

TURING, A. M. Computing machinery and intelligence. **Mind**, Oxford University Press, v. 59, n. 236, p. 433–460, 1950.

Vercel. **Next.js – The React Framework**. 2016. Acessado em: 29 jul. 2025. Disponível em: <https://nextjs.org>.