UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

VITOR HUGO LELES FONSECA
Orientador: Prof. Dr. Reinaldo Silva Fortes

APRIMORAMENTOS DE USABILIDADE PARA O OPCODERS JUDGE

Ouro Preto, MG
2025

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS
DEPARTAMENTO DE COMPUTAÇÃO

VITOR HUGO LELES FONSECA

APRIMORAMENTOS DE USABILIDADE PARA O OPCODERS JUDGE

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Reinaldo Silva Fortes

Ouro Preto, MG
2025

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E BIOLOGICAS
DEPARTAMENTO DE COMPUTACAO

FOLHA DE APROVAÇÃO

Vitor Hugo Leles Fonseca

Aprimoramentos de usabilidade para o opCoders Judge

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 31 de Março de 2025.

Membros da banca

Reinaldo Silva Fortes (Orientador) - Doutor - Universidade Federal de Ouro Preto
Aline Norberta de Brito (Examinadora) - Doutora - Universidade Federal de Ouro Preto
Saul Emanuel Delabrida Silva (Examinador) - Doutor - Universidade Federal de Ouro Preto

Reinaldo Silva Fortes, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 31/03/2025.

Documento assinado eletronicamente por **Reinaldo Silva Fortes**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 01/04/2025, às 07:35, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0885589** e o código CRC **8D3C9A98**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: 3135591692 - www.ufop.br

Aos meus pais, que nunca mediram esforços para que eu chegasse até aqui.

Agradecimentos

Agradeço a Deus, por me guiar com sabedoria, força e fé durante este processo. Sua presença em minha vida foi essencial para superar os desafios e seguir em frente com determinação.

Aos meus pais, Tânia e Luiz, minha eterna gratidão. Pelo amor incondicional, pelo apoio constante e pela confiança que sempre depositaram em mim. Sem os ensinamentos, o incentivo e a dedicação de vocês, eu não teria chegado até aqui. Agradeço profundamente por terem aberto mão de tantas coisas para que eu tivesse uma boa educação. Vocês foram minha fonte de motivação e coragem, e não tenho palavras suficientes para expressar minha gratidão por tudo o que fizeram por mim.

Aos meus tios, tias, primos, primas e demais familiares, que sempre estiveram ao meu lado, oferecendo amor e apoio a cada momento dessa caminhada. O suporte de cada um de vocês foi fundamental para que eu chegasse até aqui.

À república Tabu, meu lar em Ouro Preto, lugar onde cresci como pessoa e fiz amizades que quero levar por toda a vida. Sou imensamente grato por cada momento compartilhado, pelas risadas, pelas conversas profundas e pelos desafios que enfrentamos juntos. Vocês não são apenas amigos, mas uma parte essencial da minha história, e tenho certeza de que, independentemente de onde a vida me levar, levarei cada um de vocês no meu coração. Obrigado por tudo, por estarem ao meu lado e por me mostrarem o verdadeiro valor da amizade.

Às amizades que fiz em Ouro Preto, minha eterna gratidão. Essa cidade, com sua história e cultura, me presenteou com pessoas incríveis que se tornaram uma parte fundamental da minha jornada. São amizades que sei que vou levar para toda a vida, não importa onde o destino nos leve. Obrigado por fazerem parte da minha história e por me acompanharem nesse caminho de crescimento e aprendizado.

Ao TerraLAB, laboratório onde iniciei meu desenvolvimento profissional. Agradeço especialmente ao Tiago, Rodrigo e Pedro, que foram fundamentais nessa jornada. Grande parte do que aprendi profissionalmente se deve a vocês.

Ao meu orientador, Reinaldo, meu sincero agradecimento pelo acompanhamento neste trabalho, pela paciência, disponibilidade e a boa vontade em me ajudar em todos os momentos em que precisei. Com a sua orientação, a caminhada até a conclusão deste trabalho se tornou mais leve e gratificante.

À UFOP e professores do DECOM, pelo ensino público, gratuito e de qualidade.

Agradeço imensamente a todos que, de alguma forma, contribuíram para que este trabalho se tornasse realidade.

Resumo

As plataformas educacionais revolucionam o ensino de programação, tornando o aprendizado mais acessível e cativante, ao fornecer correções automáticas e retorno imediato aos estudantes. Nesse contexto, este trabalho apresentou melhorias na usabilidade do *opCoders Judge*, uma ferramenta de correção automática de código para disciplinas de programação, para proporcionar uma experiência de uso mais fluida e eficiente. Para isso, foi primeiramente realizada uma análise das telas e funcionalidades da plataforma, identificando pontos que precisavam de ajustes importantes. A partir dessa avaliação, foram traçados objetivos claros, sendo o principal deles o de desenvolver uma interface mais organizada e fácil de usar. Entre os ajustes, destaca-se a inclusão de gráficos para acompanhamento do desempenho dos alunos, a reorganização das telas para melhorar a navegação e a criação de um sistema de recomendação de exercícios personalizado, com base no desempenho anterior dos alunos. Essas mudanças resultaram em uma interface mais clara e funcional, facilitando a organização das atividades por parte dos estudantes. Os resultados mostram que as melhorias tornarão a navegação mais simples e intuitiva, além de motivar os alunos a se dedicar mais aos estudos, promovendo um aprendizado mais eficaz. No contexto acadêmico, a pesquisa oferece uma contribuição significativa ao aplicar princípios de *design* centrados no usuário, criando uma experiência de aprendizado mais fluida e envolvente, essencial para manter os alunos engajados. As conclusões indicam que a nova versão da plataforma tem o potencial de estar mais bem-adaptada às necessidades dos usuários, com a expectativa de que, em etapas futuras de desenvolvimento, auxilie de forma mais eficaz na preparação dos alunos para atuar na área de tecnologia. Ainda falta concluir aspectos importantes do projeto, como o protótipo das interfaces do sistema de recomendação de exercícios. No entanto, ainda há etapas pendentes. Este trabalho focou na projeção das interfaces e funcionalidades, mas a implementação das novas funcionalidades e do sistema de recomendação de exercícios, por exemplo, será realizada em trabalhos futuros.

Palavras-chave: Usabilidade. Corretor de código-fonte. Prototipagem de interfaces. Aprendizado personalizado. Melhorias de usabilidade

Abstract

Educational platforms are revolutionizing the teaching of programming, making learning more accessible and engaging by providing automatic corrections and immediate feedback to students. In this context, this work presented improvements to the usability of *opCoders Judge*, an automatic code correction tool for programming subjects, to provide a more fluid and efficient user experience. The platform's screens and functionalities were first analyzed to identify points that needed adjustments. Based on this assessment, clear objectives were set, the main one being to develop a more organized and user-friendly interface. Among the adjustments, we highlight the inclusion of graphs to monitor student performance, the reorganization of screens to improve navigation, and the creation of a personalized exercise recommendation system based on student's previous performance. These changes have resulted in a more straightforward and functional interface, making it easier for students to organize their activities. The results show that the improvements will make navigation more concise and intuitive and motivate students to dedicate themselves more to their studies, promoting more effective learning. In the academic context, the research significantly contributes by applying user-centered design principles, creating a more fluid and engaging learning experience, which is essential for a more effective learning experience.

Keywords: Usability. Source code checker. Interface prototyping. Personalized learning. Usability improvements

Lista de Ilustrações

Lista de Abreviaturas e Siglas

LLM *Large Language Model*. 9

SAM *Self-Assessment Manikin*. 8, 9

SUS *System Usability Scale*. 5, 8, 9

UX *User Experience*. ix, 4

Sumário

1 Introdução

A evolução das plataformas educacionais tem sido fundamental para transformar o ensino de programação. Elas tornam o aprendizado mais acessível, interativo e dinâmico, permitindo que os alunos recebam correções automáticas e *feedback* instantâneo. No entanto, essa eficácia não é garantida. A qualidade da experiência de aprendizado depende da usabilidade da plataforma. Segundo Nielsen e Molich (1990), a usabilidade é fundamental para a satisfação do usuário; se a plataforma não for bem projetada, mesmo as melhores ferramentas podem não conseguir manter os alunos interessados, o que pode prejudicar seu desempenho acadêmico. Por isso, é essencial que as plataformas educacionais foquem na usabilidade durante seu desenvolvimento, criando um ambiente de aprendizado que seja eficaz e agradável.

Este trabalho aborda o aprimoramento da usabilidade da plataforma *opCoders Judge*, uma ferramenta utilizada para a correção automática de exercícios práticos de disciplinas de programação, uma vez que as dificuldades percebidas, como a navegação confusa, podem impactar negativamente a experiência de aprendizado dos alunos, dificultando a localização de recursos, a compreensão do fluxo de tarefas e levando a frustrações que reduzem a eficácia do aprendizado. Embora estudos anteriores, como os de Patrocínio (2023) e Cedraz (2023), tenham mostrado a importância e os benefícios de plataformas como o *opCoders Judge*, eles também identificaram problemas na interface e na interação dos usuários que afetam diretamente a eficiência da ferramenta e a experiência de aprendizado dos alunos. Além disso, a falta de personalização dificulta que a plataforma atenda às necessidades específicas de cada aluno, o que pode frustrar tanto aqueles que precisam de mais ajuda quanto os que buscam desafios maiores.

Implementar melhorias que priorizem a clareza na navegação e a personalização dos conteúdos pode trazer benefícios significativos, não apenas em termos de eficiência, mas também no engajamento dos alunos. Ao oferecer o acesso às informações e recursos, a plataforma permite que os estudantes se sintam mais confortáveis em explorar e utilizar suas funcionalidades, criando um ambiente propício para o desenvolvimento de habilidades essenciais na área de tecnologia.

As seções a seguir estão organizadas da seguinte forma. Na Seção 1.1, é apresentada a justificativa para a realização deste trabalho, destacando a importância de melhorias na usabilidade da plataforma *opCoders Judge* no contexto educacional. Na Seção 1.2, são esclarecidos os objetivos gerais e os objetivos específicos, que visam aprimorar a interação dos alunos com a plataforma por meio de práticas de *design* centradas no usuário. Na Seção 1.3, é apresentada a organização deste trabalho, com uma descrição dos capítulos que compõem a estrutura do estudo.

1.1 Justificativa

Com o avanço das tecnologias e o aumento do uso de plataformas digitais na educação, é essencial investigar não apenas melhorias de usabilidade, mas também o desenvolvimento de novas funcionalidades que aumentem a capacidade dessas ferramentas. Este trabalho foca no *opCoders Judge*, uma plataforma com grande potencial para contribuir significativamente na formação de novos profissionais na área de tecnologia, indo além dos estudos prévios de usabilidade ao propor a implementação de funcionalidades inovadoras.

Além de melhorar a usabilidade da plataforma, este estudo explora novas funcionalidades voltadas para personalizar a experiência de aprendizado, como o sistema de recomendação de exercícios, ajustado ao desempenho de cada aluno, e o editor de código-fonte, para deixar o processo de desenvolvimento mais rápido e bem organizado. Isso cria um plano de estudo mais flexível e individualizado, permitindo que os estudantes avancem no próprio ritmo, tanto aqueles que precisam de mais suporte quanto os que buscam desafios mais complexos. A implementação dessas funcionalidades torna o aprendizado mais eficaz e envolvente, otimizando não apenas a interface, mas também a maneira como o conteúdo é apresentado. Este estudo também abordou o aprimoramento de telas e funcionalidades já existentes, como Minhas Tarefas, Atividade Aberta, Perfil, Como Usar? e a tela inicial.

Esse avanço não só preenche lacunas deixadas por estudos anteriores, como também serve de referência para futuros desenvolvimentos na área de plataformas educacionais, garantindo que o *opCoders Judge* continue a evoluir em termos de inovação e impacto educacional. Ao unir melhorias de usabilidade com a criação de novas funcionalidades, este trabalho reflete um compromisso em contribuir para a formação de profissionais mais capacitados e prontos para enfrentar os desafios do mercado de trabalho atual.

1.2 Objetivos

O objetivo principal deste trabalho é aprimorar a usabilidade da plataforma *opCoders Judge*, propondo uma reformulação centrada no usuário que melhore a interação dos alunos com as ferramentas de envio e correção de código, promovendo uma experiência de aprendizado mais fluida e envolvente. Para isso, serão adotadas práticas de design centradas nas necessidades dos usuários, buscando resolver questões específicas de navegação e interação identificadas nas versões anteriores da plataforma.

Além dos aprimoramentos de usabilidade, este trabalho propõe a implementação de novas funcionalidades inovadoras, como uma tela demonstrando como exercícios podem ser recomendados com base no desempenho individual dos estudantes, e um editor de código-fonte integrado, tornando o ambiente de aprendizagem mais personalizado e adaptável a diferentes níveis de conhecimento e ritmo de aprendizagem.

Para alcançar o objetivo principal, serão abordados os seguintes objetivos específicos:

- Propor aprimoramentos nas interfaces das seções “Minhas Tarefas”, “Atividade Aberta”, “Perfil”, “Como Usar?” e na tela inicial.
- Propor o design inicial de um sistema de recomendação para adaptar os exercícios conforme o desempenho do aluno, apresentando uma tela que ilustra como o plano de estudo personalizado poderia ser gerado;
- Propor o design de um editor de código-fonte integrado à plataforma que auxilie os alunos na escrita, organização e depuração do código.

1.3 Organização do Trabalho

A organização do restante do trabalho está estruturada da seguinte forma:

Capítulo 2: apresenta uma revisão bibliográfica, contendo fundamentação teórica e trabalhos relacionados ao desenvolvimento do *opCoders Judge*, abordando aspectos como usabilidade, *design* de interface, desenvolvimento de interfaces responsivas, prototipação e melhorias implementadas na plataforma com base em trabalhos anteriores;

Capítulo 3: apresenta as telas projetadas para o *opCoders Judge*, destacando melhorias e novas propostas de interface com base em critérios de usabilidade, experiência do usuário e design, evidenciando ganhos em navegabilidade, clareza visual e organização da plataforma;

Capítulo 4: apresenta conclusões e trabalhos futuros.

2 Revisão Bibliográfica

Este capítulo apresenta uma revisão de literatura para o desenvolvimento desta monografia. A literatura relevante para o tema deste trabalho foi obtida por meio de portais de referências, tais como Google Acadêmico, Portal de periódicos CAPES, *CEEOL*, *Taylor e Francis*. Foram buscados autores vinculados aos temas analisados neste projeto, abrangendo o período específico de 2014 a 2024, admitindo-se a inclusão de trabalhos mais antigos, conforme necessário.

Sendo assim, este capítulo foi estruturado em algumas seções. A Seção 2.1 aborda a fundamentação teórica que embasa o desenvolvimento deste trabalho monográfico. Em seguida, na Seção 2.2, são apresentados os trabalhos relacionados.

2.1 Fundamentação Teórica

A fundamentação teórica do desenvolvimento do *opCoders Judge* apoia-se em diversos estudos e ferramentas que têm contribuído significativamente para a evolução das plataformas de ensino e correção automática de código. A integração de sistemas tecnológicos no ensino de programação é um tema amplamente discutido na literatura acadêmica, abordado sob diferentes perspectivas, como usabilidade, personalização e automação de conteúdo, e facilitação do aprendizado por meio de exercícios práticos para melhorar a experiência de aprendizado dos alunos e facilitar a gestão de atividades pelos professores.

Na Subseção 2.1.1 são explorados os princípios de usabilidade e experiência do usuário (UX), fundamentais para garantir que o *opCoders Judge* ofereça uma interface intuitiva, com navegação eficiente e adequada às expectativas dos usuários, utilizando heurísticas como as de Nielsen e Molich (1990). Já na Subseção 2.1.2 discute-se o design de interface, que combina aspectos visuais e de interação para criar uma experiência agradável e fluida, aumentando a satisfação e confiança dos usuários ao interagir com a plataforma. Na Subseção 2.1.3 aborda a prototipação, permitindo testar e refinar ideias de design antes da implementação final, assegurando que a plataforma esteja alinhada com as expectativas de seus usuários e otimizando a usabilidade durante o desenvolvimento. Por fim, a Subseção 2.1.4 apresenta a escolha do Figma como ferramenta para desenvolver protótipos de média fidelidade, destacando seus benefícios.

2.1.1 Usabilidade e Experiência do Usuário (UX)

A usabilidade está relacionada a quão fácil é para alguém usar um produto e alcançar seus objetivos de forma eficiente, eficaz e satisfatória, sem enfrentar dificuldades durante o uso que possam afetar negativamente a experiência. Nielsen e Molich (1990) definiram 10 heurísticas de usabilidade que servem como diretrizes para avaliar como as pessoas interagem com sistemas,

como garantir a consistência nas interfaces, dar *feedback* claro para as ações do usuário e a importância de oferecer opções de controle para que ele se sinta no comando das ações. Essas heurísticas são aplicadas em análises práticas, como no trabalho de Cedraz (2023), para identificar falhas que prejudiquem a experiência do usuário.

O Sistema de Usabilidade de Software (*System Usability Scale* (SUS)) é um método padronizado que ajuda a compreender como os usuários percebem a facilidade de uso e o nível de controle que têm ao interagir com o sistema. O SUS é uma escala simples de dez itens que fornece uma visão global de avaliações subjetivas de usabilidade (BROOKE, 1996).

Essa ferramenta permite obter detalhes que nem sempre são evidentes, como a sensação de frustração causada por uma navegação confusa ou a satisfação em utilizar uma interface clara e intuitiva, sentindo que cada clique faz sentido e que está no controle, podendo realizar as tarefas sem dificuldades.

Aplicar essas métricas ao *opCoders Judge* contribuiu para oferecer uma visão mais detalhada sobre quais elementos da plataforma facilitam o aprendizado e quais causam frustração ou desmotivação, seja por uma interface pouco intuitiva ou por dificuldades no acesso a funcionalidades importantes para alunos e professores. Isso permite identificar com mais clareza os pontos que realmente agregam valor e as áreas que precisam de melhoria para tornar a experiência mais fluida e eficaz.

2.1.2 Design de Interface

O *Design de Interface* envolve a criação de interfaces que tornam a interação entre usuários e sistemas mais eficiente. Quando bem feita, uma interface pode melhorar a experiência do usuário, aumentar a retenção e a confiança. Essa área combina elementos de *design* gráfico, psicologia e usabilidade. De acordo com Norman (2013), o *design* de interação é o projeto de como os usuários interagem com produtos e serviços, considerando suas expectativas e comportamentos.

Para ter bons resultados, é essencial focar na criação de elementos visuais e na organização do *layout*, projetando cuidadosamente componentes como botões, menus e ícones usando cores, formas e tipografia que sejam claras e agradáveis, para facilitar a interação entre o usuário e o *software*. A organização do *layout* é essencial para a navegação, ao auxiliar os usuários a encontrarem as informações de maneira intuitiva.

2.1.3 Prototipação

A prototipação é uma etapa essencial no *design* e desenvolvimento de interfaces, permitindo que designers e desenvolvedores testem e validem suas ideias antes da implementação final (DAM; SIANG, 2020). Essa técnica cria modelos interativos ou visuais que simulam como os usuários irão interagir com o sistema.

Existem diferentes tipos de protótipos, variando em fidelidade e complexidade. Os de **baixa fidelidade** incluem simples esboços ou *wireframes*, enquanto os de **alta fidelidade** oferecem uma simulação mais detalhada da interface final. Os protótipos de **média fidelidade**, utilizados na Seção 3 deste trabalho, combinam elementos de ambos: possuem um *layout* mais elaborado e oferecem funcionalidades básicas, mas não têm o nível completo de interatividade dos protótipos de alta fidelidade. Eles são ideais para testar fluxos de navegação e funcionalidades de forma prática, sem exigir tanto tempo e recursos como seriam necessários para desenvolver uma versão completamente funcional.

Entre as ferramentas mais usadas para criar protótipos estão o *Sketch*, *Adobe XD*, *Adobe Illustrator* e *Figma*. O *Sketch* é conhecido por ser rápido e eficiente no *design* de interfaces, mas só funciona em computadores com *macOS*. O *Adobe XD* se conecta bem com outros programas da *Adobe* e é uma boa opção para quem já usa essa suíte de programas, sendo ótimo para criar protótipos interativos. O *Adobe Illustrator*, sendo mais usado para desenhos e gráficos, também pode ser usado para *design* de interfaces, mas é um pouco mais complicado de usar para isso. Já o *Figma* se destaca por funcionar em qualquer sistema, permitir que várias pessoas trabalhem colaborativamente e ser fácil de usar; por isso, foi escolhido.

Ao integrar a prototipação no aprimoramento da usabilidade do *opCoders Judge*, é possível identificar problemas de forma mais eficiente, explorar diferentes soluções e garantir que a plataforma seja intuitiva e atenda às expectativas dos usuários de maneira mais eficaz.

2.1.4 Ferramenta utilizada

Para o desenvolvimento dos protótipos deste trabalho, foi escolhido o Figma, uma ferramenta amplamente utilizada na área de design de interfaces digitais. A decisão foi tomada com base em suas funcionalidades completas, na facilidade de uso, e na sua acessibilidade. Por ser uma plataforma online, o Figma não precisa de instalações locais e pode ser acessado diretamente pelo navegador, o que facilita bastante o trabalho colaborativo e o acesso remoto, especialmente em contextos acadêmicos.

Os protótipos foram desenvolvidos em média fidelidade, para representar de forma clara e objetiva a estrutura da interface e os principais fluxos de navegação. Embora não incluíssem todos os detalhes visuais finais, esse nível de fidelidade se mostrou adequado para comunicar a proposta do projeto com clareza, além de servir como base para orientar as próximas etapas do desenvolvimento. Também foi essencial para identificar possíveis melhorias na experiência do usuário ainda nas fases iniciais, o que possibilitou realizar ajustes rápidos antes de avançar para os protótipos de alta fidelidade.

2.2 Trabalhos Relacionados

Diversos trabalhos relacionados e ferramentas de diferentes autores desempenharam um papel importante para o desenvolvimento do projeto *opCoders Judge*. A seguir, são discutidos alguns desses trabalhos, para contextualizar suas contribuições e destacar a relevância de cada um no processo de desenvolvimento do *opCoders Judge*.

O primeiro trabalho a ser citado é a monografia de Brito (2019), que surgiu como resposta à abundância de desistências e reprovações nas disciplinas de programação, resultado da dificuldade dos alunos em lidar com o alto nível de abstração exigido. A proposta principal foi testar o uso de ferramentas computacionais, como um corretor automático de código, para auxiliar os alunos a compreenderem melhor os conceitos por meio de exercícios práticos e diretos.

Ao final do estudo, foi feita uma análise com 31 alunos que utilizaram a ferramenta, buscando entender onde estavam os desafios e quais melhorias poderiam ser feitas para torná-la ainda mais útil. Os resultados mostraram que os alunos que se dedicaram às atividades práticas conseguiram um desempenho superior também nas avaliações teóricas. Além disso, muitos relataram que as aulas práticas foram decisivas para o sucesso na disciplina, reforçando o valor de uma abordagem mais aplicada. Esses dados sugerem que as ferramentas computacionais e a metodologia adotada realmente ajudaram a tornar o aprendizado mais concreto e acessível, melhorando o desempenho dos alunos.

Entre as limitações identificadas, uma questão importante foi a necessidade de expandir a ferramenta para um ambiente *online*, permitindo que mais alunos pudessem utilizá-la de forma prática e colaborativa, além de facilitar atualizações rápidas e melhorias contínuas na plataforma.

Patrocínio (2023) deu continuidade ao projeto iniciado por Brito (2019), visando aprimorar o módulo *web* do *opCoders Judge*. As melhorias implementadas incluem a criação de um módulo administrativo completo e a expansão do suporte a múltiplas linguagens de programação. Além disso, Patrocínio (2023) aprimorou o módulo *web* do *opCoders Judge* para torná-lo mais eficiente e funcional, visando auxiliar no processo de aprendizagem de programação dos alunos, oferecendo correções rápidas das atividades submetidas.

A transição para a versão *online* do *OpCoders Judge* representa um avanço significativo em relação à versão *offline* anteriormente desenvolvida. Com a implementação da versão *online*, tanto alunos quanto professores passaram a ter acesso a uma plataforma *web* mais prática, ágil e interativa para corrigir e gerenciar atividades de programação. A versão *online* facilita não apenas a correção das atividades pelos alunos, mas também a gestão e correção pelas turmas grandes dos professores, reduzindo a carga de trabalho relacionada às numerosas tarefas e exercícios.

Essa evolução representa uma melhoria substancial no processo de correção e gestão de exercícios de programação, tornando-o mais acessível, dinâmico e eficaz. Com isso, busca-se não apenas aprimorar a interface do usuário, oferecendo mais recursos para os alunos e um módulo administrativo mais completo para os professores, mas também aumentar a motivação de todos

os envolvidos nas disciplinas de programação, essenciais para as engenharias e outros campos do conhecimento.

Cedraz (2023) realizou uma avaliação de usabilidade do *opCoders Judge* para entender se os usuários acham a ferramenta fácil de usar e se conseguem realizar suas tarefas de maneira satisfatória e eficiente. O estudo avaliou a qualidade da interface do sistema, verificando se ela realmente oferece uma boa experiência em termos de facilidade, rapidez e satisfação dos usuários.

No trabalho, foram realizados testes de usabilidade com alunos de uma disciplina introdutória de programação. Os testes incluíram métodos de avaliação como a análise das 10 Heurísticas de Nielsen e a aplicação de questionários como o *System Usability Scale* (SUS) e *Self-Assessment Manikin* (*Self-Assessment Manikin* (SAM)). Esses métodos ajudaram a identificar problemas específicos na interface e entender como os alunos se sentiam ao usar o sistema, incluindo se eles se sentiam motivados e no controle.

Além disso, os testes envolveram a análise de um protótipo mais avançado do *opCoders Judge*, onde foram verificadas as correções feitas após a identificação dos problemas. O foco foi saber se essas melhorias realmente facilitaram a vida dos usuários, medindo o quanto eles acertavam nas tarefas, o número de problemas encontrados, o tempo gasto em cada atividade e como a interface se comportava no geral.

Mendonça (2023) aborda o desenvolvimento de uma plataforma de gestão de questões dinâmicas para o ensino de programação de computadores. O trabalho visa atender à necessidade de criar questões distintas para cada aluno, aprimorando a plataforma existente com um módulo capaz de gerar uma infinidade de questões únicas. A importância de questões inéditas e contextualizadas é destacada para melhorar a compreensão dos conceitos e promover autonomia nos alunos, proporcionando variedade e personalização nas avaliações. A evolução do banco de dados e a modelagem conceitual foram fundamentais para o desenvolvimento do módulo, contribuindo para uma experiência educacional enriquecedora, mais eficaz e com variedade e personalização nas questões.

Silva (2023) abordou em seu trabalho o aprimoramento do *opCoders Judge*, com foco na melhoria da interface, migração de tecnologias e remodelagem do banco de dados. Foram realizados testes de usabilidade, que revelaram a necessidade de mudanças na interface para torná-la mais atraente e intuitiva. Além disso, foram identificados problemas de usabilidade na versão atual da ferramenta, principalmente relacionados à tela de envio das tarefas.

O aprimoramento da ferramenta com base em testes de usabilidade envolveu a realização de testes com voluntários, incluindo alunos de diferentes cursos da disciplina de Programação de Computadores I, para identificar e corrigir possíveis problemas na interface e na experiência do usuário. Foram aplicados questionários como SUS e SAM para avaliar a efetividade, eficiência e satisfação dos participantes.

Os testes de usabilidade permitiram coletar *feedback* direto dos usuários, avaliando

aspectos como facilidade de uso, eficiência e satisfação geral com a ferramenta. Com base nos resultados desses testes, foram implementadas melhorias na interface, migração de tecnologias e remodelagem do banco de dados, visando aprimorar a usabilidade e a experiência do usuário ao utilizar o *opCoders Judge*.

Zimerman (2024) realizou experimentos que envolveram a exploração de estratégias de interação com as ferramentas de chatbots baseados em *Large Language Model* (LLM), como o *ChatGPT*, *ChatPDF*, *PopAI* e *Sider*, explorando estratégias de interação e o uso de *prompts* para direcionar a geração de questões de programação de computadores de forma mais precisa e personalizada. Essa abordagem mostrou-se eficaz na automação e diversificação do processo de criação de questões, evidenciando a viabilidade do uso de *chatbots* para esse fim. Os experimentos destacaram a capacidade dessas tecnologias em gerar questões relevantes e diversificadas, contribuindo para melhorar o ensino de programação.

Os trabalhos de Cedraz (2023) e Silva (2023) têm uma relação direta com o foco deste trabalho monográfico, que visa aprimorar a usabilidade do *opCoders Judge*. O trabalho de Cedraz (2023) é relevante por abordar a avaliação da usabilidade da ferramenta, utilizando métodos como as 10 Heurísticas de Nielsen e questionários como o SUS e SAM, amplamente utilizados em pesquisas para avaliar a experiência e a resposta emocional de usuários em relação a produtos ou interfaces. Esses métodos forneceram uma boa visão sobre os problemas de usabilidade enfrentados pelos alunos. No entanto, enquanto Cedraz (2023) se concentrou em testes com usuários e ajustes na interface a partir de problemas detectados, o presente trabalho foca em uma etapa posterior, implementando funcionalidades que vão além de correções de interface. Entre essas funcionalidades estão a melhoria na hierarquia tipográfica e a criação de um sistema de recomendação personalizado para os alunos.

O trabalho de Silva (2023), por sua vez, teve como foco a remodelagem do banco de dados e melhorias na interface, principalmente na tela de envio de tarefas. Em comparação, o presente trabalho propõe também melhorias interativas na tela de correção de código e a criação de protótipos responsivos, adaptados para diferentes resoluções de dispositivos. Além disso, enquanto os trabalhos anteriores destacaram a importância de uma interface intuitiva, o presente trabalho amplia essa visão ao incorporar elementos dinâmicos e automatizados, como a tela de questões geradas automaticamente, além de estender as características das questões para incluir áreas de conhecimento, assuntos de interesse, tópicos relacionados à disciplina e graus de dificuldade, integrando as funcionalidades dos trabalhos desenvolvidos por Mendonça (2023) e Zimerman (2024) à interface do *opCoders Judge*.

3 Desenvolvimento

Neste capítulo, serão apresentadas as telas projetadas para o *opCoders Judge*, incluindo tanto melhorias em relação às versões anteriores da plataforma quanto novas propostas de interface. A análise será feita com base em critérios de usabilidade, experiência do usuário e *design* de interface, destacando as melhorias em termos de navegabilidade e clareza visual. Sempre que aplicável, será realizada uma comparação entre as versões anteriores e as novas propostas, evidenciando as melhorias implementadas. Serão discutidos os benefícios das mudanças implementadas, como a melhoria da estrutura e organização da interface. Para as novas telas propostas, será discutida a justificativa de sua criação e os benefícios que trazem à organização e estrutura geral da plataforma.

A Seção 3.1 apresenta uma visão geral da estrutura da plataforma. A Seção 3.3 apresenta o novo layout da tela “Minhas Tarefas”. A Seção 3.4 apresenta um sistema de recomendação de exercícios personalizados. A Seção 3.5 apresenta o *header* reformulado. A Seção 3.6 apresenta o *footer* reformulado. A seção 3.2 apresenta melhorias na organização da interface da tela inicial. A Seção 3.7 apresenta a tela “Perfil”. A Seção 3.8 apresenta a tela “Atividade Aberta”. A Seção 3.9 descreve várias melhorias na interface do sistema para aprimorar a experiência do aluno.

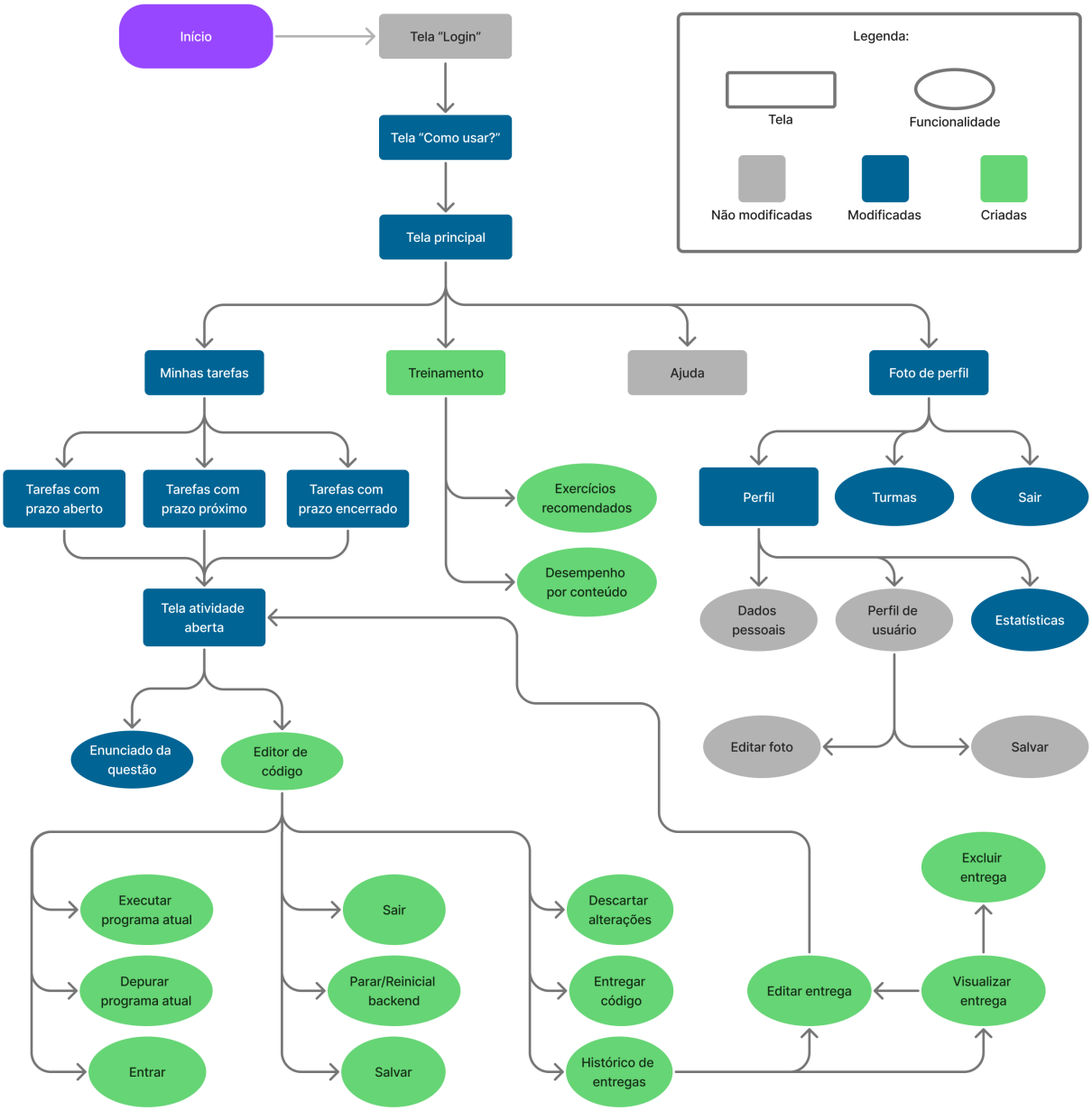
3.1 Mapa da plataforma

Para proporcionar uma visão clara das mudanças propostas, esta seção apresentará o mapa da plataforma, evidenciando sua estrutura geral e destacando as telas que foram modificadas ou criadas. Além do mapa visual, será feito um breve panorama das principais modificações realizadas, oferecendo um resumo do que foi desenvolvido. Em seguida, cada tela será detalhada em seções específicas, onde serão abordadas as alterações prototipadas, os novos componentes introduzidos e os critérios adotados para as decisões de design.

O fluxograma utiliza as cores cinza, azul e verde para indicar, respectivamente as telas e funcionalidades que não foram modificadas, as que foram modificadas e as que foram criadas.

1. **Tela inicial**
    a. **Exibe:**
        i. Texto: “Seja bem-vindo”;
        ii. Botão: “Login com e-mail institucional Google”.
2. **Ação:**
    a. Usuário faz login

Figura 3.1 – Fluxograma da plataforma.



i. Redireciona para tela "Como usar?"

3. **Tela Como usar?**

a. **Exibe:**

i. Explicação de como utilizar o site

ii. Botão: "Não mostrar esta mensagem novamente"

a. **Ação:**

i. Usuário escolhe uma das opções do *Header*

4. **Tela principal**

a. **Header com opções:**

i. Minhas tarefas

ii. Treinamento

iii. Ajuda

iv. Perfil

a. **Opções de navegação**

i. **Minhas tarefas**

A. **Exibe 3 cards:**

I. Tarefas com prazo aberto

II. Tarefas com prazo próximo

III. Tarefas com prazo encerrado

B. **Ação:**

I. **Usuário clica em um card**

1. **Exibe lista de exercícios correspondentes**

a. **Ação:**

i. Usuário clica em um exercício

A. **Exibe:**

I. Enunciado da questão

II. Editor de código

1. **Opções:**

a. Executar programa atual

b. Depurar programa atual

c. Entrar (*debug*)

d. Sair (*debug*)

e. Parar/Reiniciar *backend*

f. Salvar

g. Descartar alterações

h. Histórico de entregas

i. Entregar código

ii. **Treinamento**

A. **Exibe recomendações de exercícios divididas em:**

I. Desempenho por conteúdo

II. Exercícios recomendados

5. **Ajuda**

a. Exibe mensagens de ajuda para cada tema da disciplina

6. **Perfil**

   a. **Exibe opções:**

      i. Dados pessoais

      ii. Perfil de usuário

      iii. Estatística

         A. Gráfico: Média das notas

         B. Gráfico: Média da turma

3.2 Tela Inicial

Na tela inicial antiga (ver Figura 3.2), a seção **Como usar?** estava integrada à tela **Minhas Tarefas**, o que gerava uma poluição visual na interface, dificultando a clareza e a organização da tela. Para resolver essa questão, agora existe uma tela dedicada exclusivamente a essa finalidade (ver Figura 3.3), que será exibida automaticamente após o login. Caso o aluno não queira mais ver a mensagem, haverá uma opção para marcar e não exibir novamente. Ao clicar em fechar, o aluno será direcionado para a tela Minhas tarefas. Essa mudança permite que os alunos tenham acesso a orientações claras e diretas sobre como interagir com a plataforma, sem as distrações relacionadas às suas tarefas pendentes.

Figura 3.2 – Tela inicial antiga.

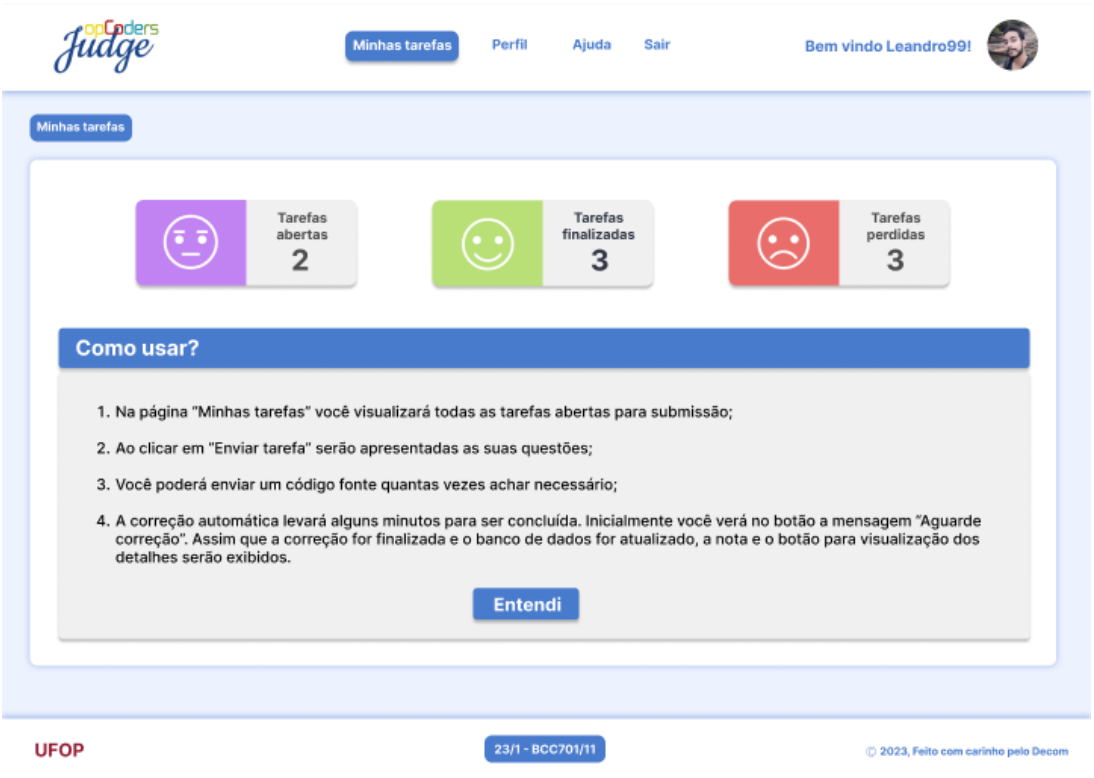
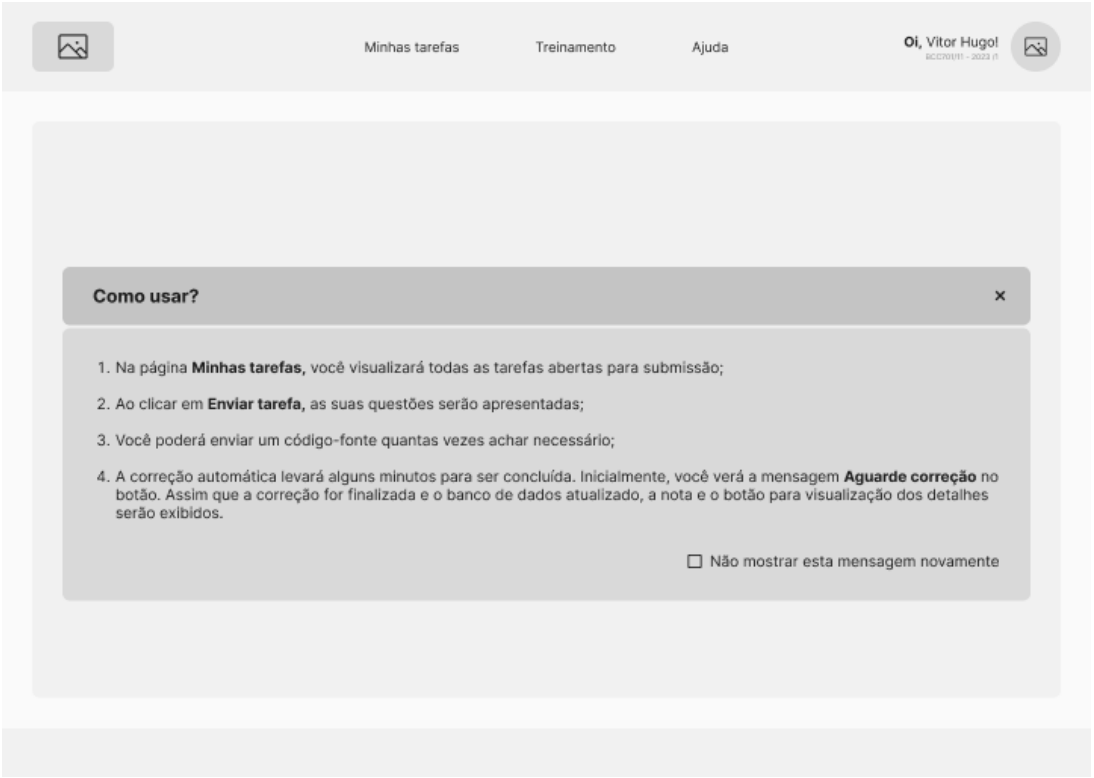


Figura 3.3 – Tela Como usar?.



3.3 Tela Minhas tarefas

Em comparação com a antiga tela (ver Figura 3.4), na tela minhas tarefas do novo *layout* (ver Figura 3.5), foram adicionados dois gráficos com o intuito de melhorar a experiência do usuário em relação à gestão de suas atividades: um de progresso das tarefas e um de prazo, trazendo vantagens tanto em termos de clareza quanto de organização das informações.

O gráfico de progresso exibe visualmente a evolução do usuário em relação às suas atividades, mostrando o número total de questões de cada tarefa, as que já foram resolvidas e as pendentes. Com isso, o aluno pode visualizar rapidamente seu desempenho, identificando quantas questões já foram concluídas e quantas ainda precisam de atenção.

O gráfico de prazos complementa a experiência ao fornecer uma visão clara das tarefas com prazos próximos de encerrar, utilizando cores para priorizar as atividades conforme o tempo restante para a entrega:

- Vermelho para tarefas com menos de 3 dias;
- Amarelo para aquelas com prazos entre 3 e 7 dias;
- Verde para tarefas com mais de 7 dias de prazo.

A principal vantagem desse gráfico é permitir que o usuário priorize facilmente as atividades mais urgentes, identificando rapidamente as tarefas em vermelho, que precisam de

Figura 3.4 – Tela minhas tarefas antiga.

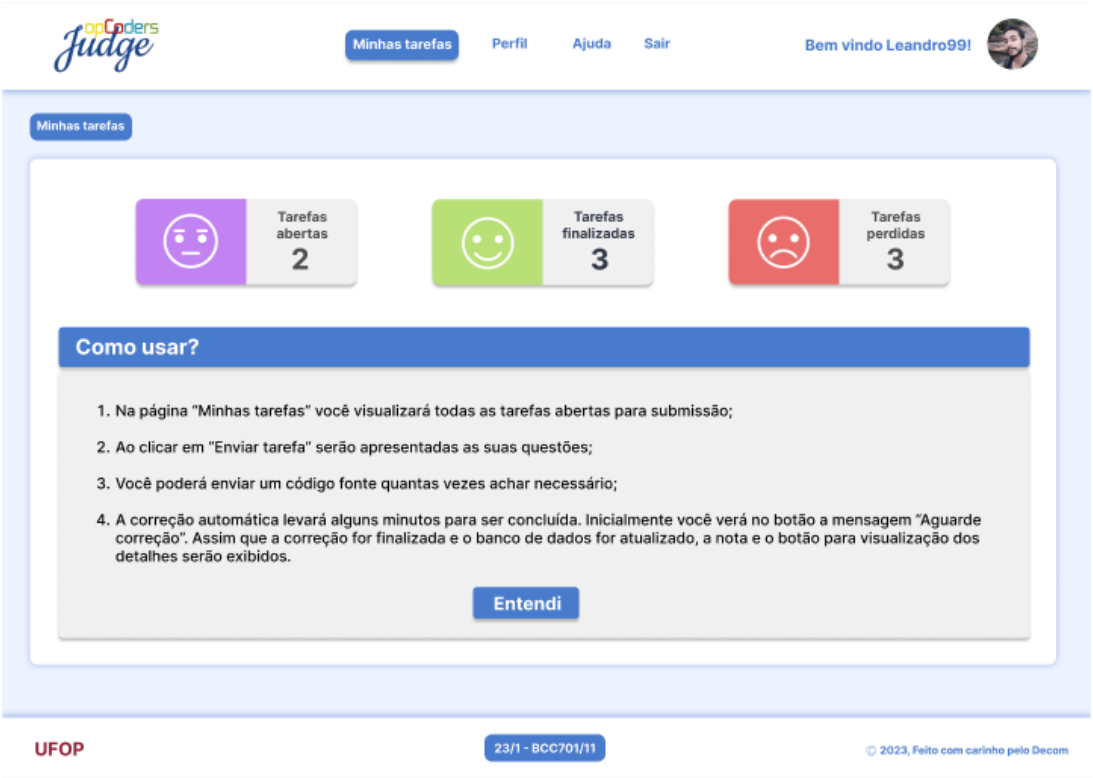
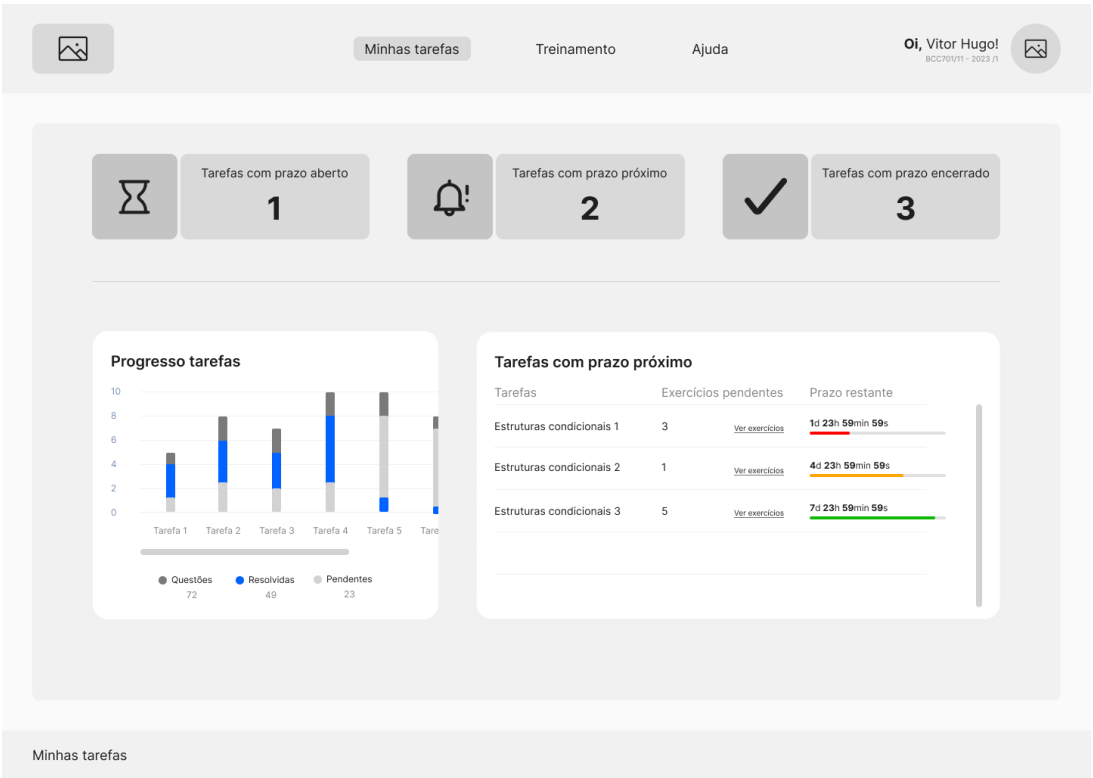


Figura 3.5 – Proposta de tela minhas tarefas.



maior atenção. Além disso, ele ajuda no controle do tempo, evitando o risco de perder prazos, garantindo que o usuário se concentre nas atividades mais urgentes.

3.4 Tela Treinamento

Esta nova tela apresenta um sistema de recomendação de exercícios que utilizará o desempenho do aluno em atividades anteriores, com foco nas notas obtidas em questões, para sugerir exercícios personalizados. A interface incluirá a visualização dessas sugestões, baseada em um algoritmo que analisará o histórico de respostas e identificará os conteúdos em que ele teve um desempenho abaixo de um limite, como uma nota mínima predefinida. Vale ressaltar que o desenvolvimento do algoritmo em si está fora do escopo deste trabalho, sendo o foco voltado para o projeto da interface que suportará essa funcionalidade.

O sistema avaliará as notas que o aluno recebeu nas questões de cada tópico, e caso ele não tenha tido um bom desempenho em algum deles (por exemplo, uma média abaixo de 60%), priorizará exercícios que abordem esse conteúdo específico. Além disso, também considerará a urgência das tarefas relacionadas, recomendando exercícios de reforço para tópicos cujos prazos estão próximos de encerrar.

O principal objetivo é fornecer um plano de estudo personalizado que auxilie o aluno a melhorar seu desempenho nas áreas de maior dificuldade, oferecendo *feedback* contínuo sobre seu progresso. Dessa forma, o sistema não apenas tornará o estudo mais eficiente, mas também apoiará o aluno em sua busca por um melhor rendimento geral na plataforma.

Figura 3.6 – Tela treinamento.

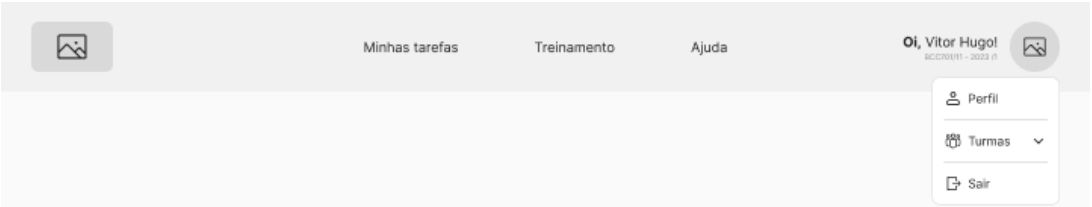
3.5 *Header*

O *header*, componente fixo no topo da interface que permite a navegação entre as páginas, passou por mudanças significativas. No protótipo anterior (ver Figura 3.7), incluía as opções de menu **Perfil** e **Sair**, que contribuíam para a poluição visual da interface. No novo protótipo (ver Figura 3.8), essas opções foram realocadas para aparecerem ao clicar na foto de perfil.

Além disso, foi acrescentada a opção **Treinamento**, que permite aos alunos praticarem com exercícios personalizados, conforme as dificuldades identificadas durante a execução das atividades. O filtro de tarefas por turma também foi reposicionado no *header*, localizado logo abaixo do nome do usuário e ao lado da foto de perfil - anteriormente, essa função ficava no *footer*. Essa nova localização facilita o acesso a essa funcionalidade, especialmente para os alunos que precisam alternar entre diferentes turmas e tarefas.

Figura 3.7 – *Header* antigo.

Figura 3.8 – *Header* novo.



3.6 *Footer*

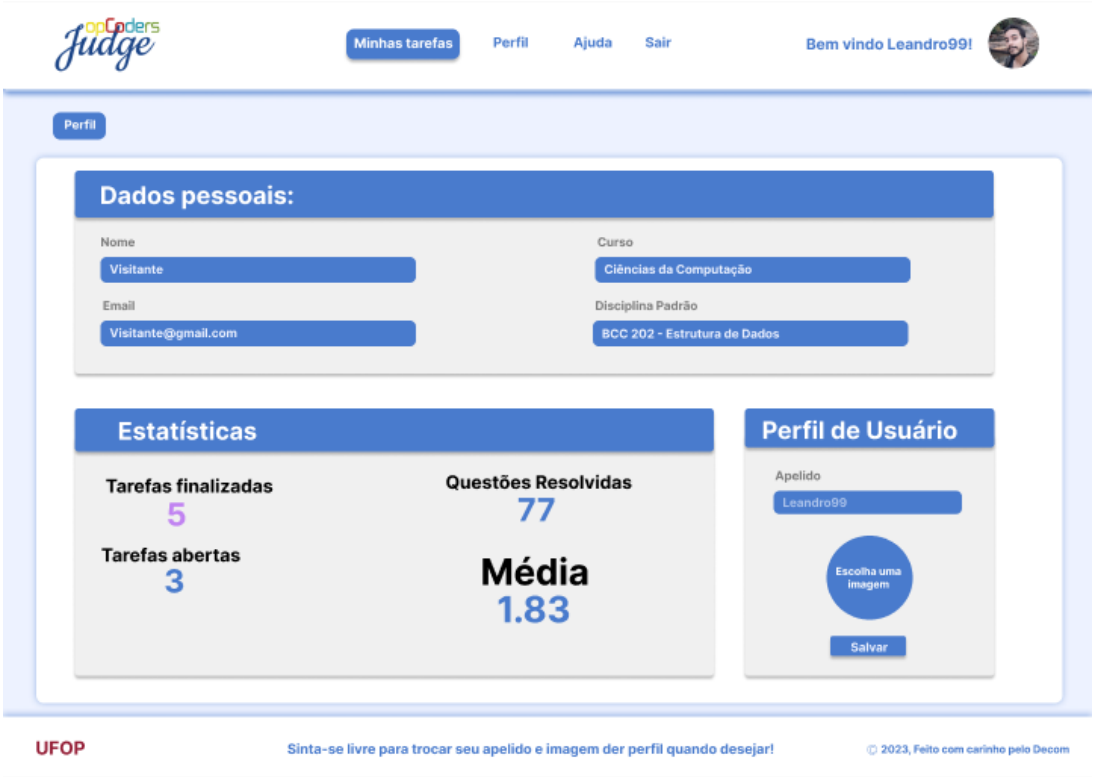
No *footer* antigo, como citado anteriormente, havia a opção de filtrar tarefas por turma, transferida para o *header*, otimizando a organização da plataforma. Além disso, o histórico de navegação, que antes ficava logo abaixo do *header*, foi movido para o *footer* (ver Figura 3.10). Essa mudança não só ajudou a otimizar o espaço na parte superior da tela, deixando o layout mais limpo e funcional, como também resultou em uma interface mais equilibrada. Vale destacar que o *footer*, assim como o *header*, será fixo na página, permitindo um acesso mais rápido e constante às funcionalidades de navegação, independentemente do local onde o usuário esteja na interface. Isso garante uma experiência de uso mais eficiente e facilita a interação contínua com a plataforma.

Figura 3.9 – *Footer* antigo.

Figura 3.10 – *Footer* novo.

3.7 Tela Perfil

Figura 3.11 – Tela Perfil antiga.



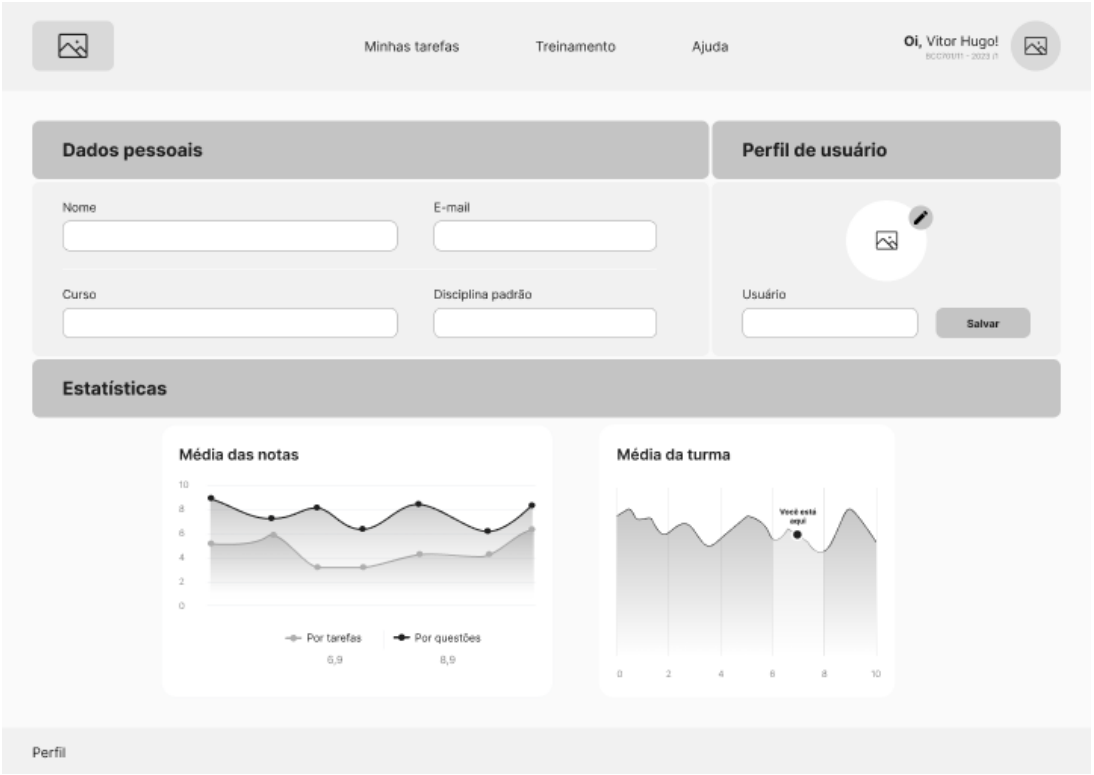
Na nova versão da tela Perfil (ver Figura 3.12), foram adicionados dois gráficos que proporcionam ao aluno uma visão detalhada do seu desempenho e de sua posição em comparação com a turma.

O primeiro gráfico apresenta a média das notas das tarefas e questões. As notas das tarefas consideram o prazo de entrega, penalizando envios fora do tempo estipulado, enquanto as notas das questões não são afetadas pelo prazo, pois o aluno pode submetê-las e ter a correção mesmo após o encerramento do tempo. Esse gráfico oferece ao aluno uma visão clara de como está se saindo em diferentes tópicos da disciplina, ajudando a identificar áreas em que ele está indo bem e outras em que pode melhorar.

O segundo gráfico mostra a média de notas da turma e posiciona o aluno em relação aos seus colegas. Essa visualização ajuda o aluno a entender como está seu desempenho comparado ao restante da turma, destacando se ele está acima, na média ou abaixo da média geral, oferecendo uma motivação extra, ao permitir que o aluno veja onde se encontra no ranking geral, facilitando a compreensão de seu progresso e incentivando melhorias, caso sua média esteja abaixo da média geral.

Além dos gráficos, foi adicionado um ícone sobre a foto de perfil, permitindo que o aluno edite sua foto de forma rápida e intuitiva, tornando a experiência mais personalizada e interativa.

Figura 3.12 – Tela Perfil nova.



3.8 Tela Atividade aberta

Foram realizadas mudanças importantes na tela de atividade aberta (ver Figura 3.15) para proporcionar uma experiência mais clara, organizada e reduzir a poluição visual. Anteriormente (ver Figura 3.13), as questões eram organizadas lado a lado, em formato de guia, semelhante à navegação em abas de um navegador de internet. No entanto, à medida que a quantidade de questões aumentasse, a leitura dos títulos ficaria prejudicada, pois não haveria espaço suficiente para exibir o texto completo. Agora, com a nova disposição por paginação, cada questão é apresentada de forma mais limpa, facilitando a navegação e o foco em uma questão por vez.

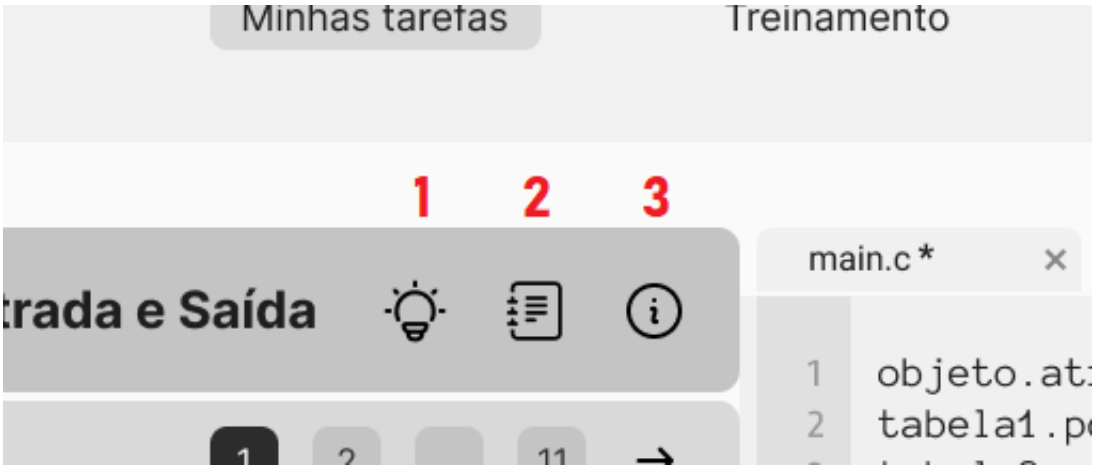
A paginação numérica segue um modelo simples e intuitivo, onde o usuário clica diretamente nos números da questão ou usa os botões de "anterior" e "próximo", neste trabalho representados por setas. Além disso, os números das questões serão combinados com indicador visual de cor para mostrar o *status* da questão (respondida, pendente, etc.) para oferecer uma visão mais clara do progresso.

Outra melhoria foi a substituição dos botões de dicas e ajuda por ícones, para otimizar o espaço e tornar o acesso às funcionalidades mais intuitivo e visualmente agradável. Também foi

Figura 3.13 – Tela Tarefa aberta antiga.

incluído um novo ícone de informações, que oferece orientações adicionais para a realização dos exercícios.

Figura 3.14 – Ícones para auxílio durante as tarefas.



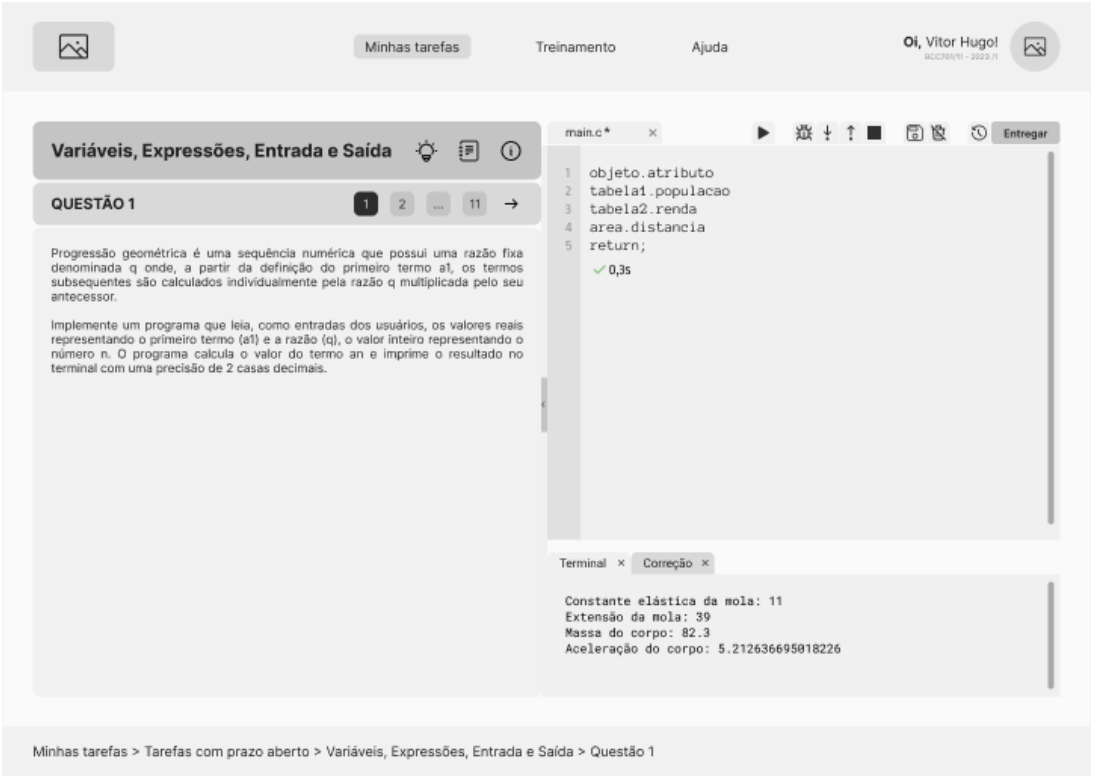
Abaixo, uma legenda numerada corresponde a cada ferramenta, explicando sua finalidade:

1. **Dicas:** organiza as informações e oferece ajuda de forma rápida ao aluno, sem ser necessário sair da tela das tarefas;
2. **Ajuda:** organiza e fornece as informações necessárias para auxiliar o aluno na realização da tarefa;

3. **Informações:** detalha quais capítulos do conteúdo são necessários para resolver as atividades e recomenda a revisão das aulas teóricas, garantindo uma preparação mais completa.

Pequenas alterações no texto das seções e dos botões também foram realizadas, para deixá-los mais diretos e alinhados com o novo design da plataforma.

Figura 3.15 – Tela Tarefa aberta nova.



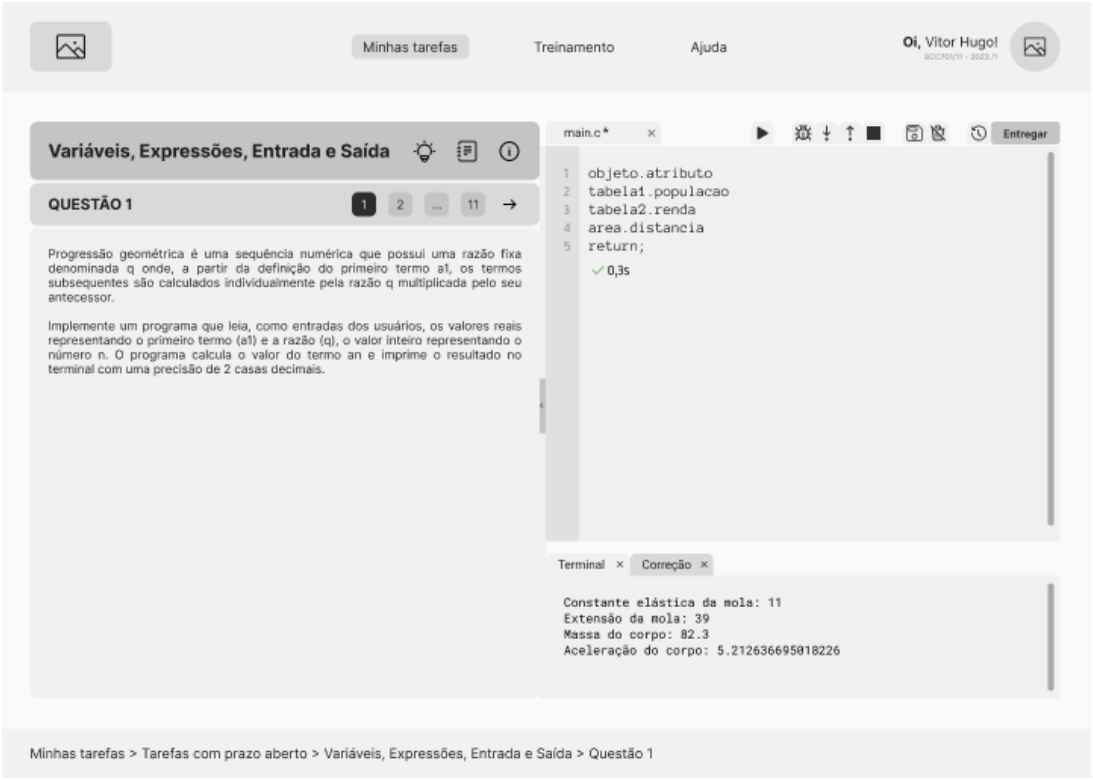
3.9 Editor de código-fonte

No desenvolvimento de *software*, um editor de código-fonte eficiente é essencial para garantir uma experiência de aprendizado produtiva e agradável dos programadores. Ele oferece diversas funcionalidades que auxiliam na escrita, organização e depuração do código, tornando o fluxo de trabalho mais ágil e estruturado. Pensando nisso, foi prototipado um editor de código-fonte no *opCoders Judge*, como pode ser visto na Figura 3.16, com diversas funcionalidades, com foco em tornar o processo de programação mais fluido, organizado e intuitivo.

Ao acessar uma questão, o editor será aberto junto ao enunciado, com ambas as seções dividindo igualmente a tela, ou seja, cada uma ocupando metade do espaço disponível. Dessa forma, o aluno poderá visualizar o enunciado e escrever seu código simultaneamente, sem a necessidade de alternar entre telas ou perder o foco da tarefa.

O objetivo principal foi oferecer um editor que não apenas permitisse a escrita de código, mas também auxiliasse os alunos no acompanhamento de suas versões, facilitasse a depuração e

Figura 3.16 – Tela com o editor de código.



incentivasse boas práticas de desenvolvimento. Para isso, foram incorporadas ferramentas que otimizam desde a execução e teste dos programas até o controle de alterações e a organização da interface.

Serão detalhadas cada uma dessas funcionalidades, apresentando como elas foram integradas ao editor e de que forma contribuem para um fluxo de trabalho mais eficiente. As subseções a seguir explicam, em detalhes, as principais funcionalidades:

Na subseção 3.9.1, será descrito como o aluno pode, com simples cliques ou arrastando as divisórias entre as áreas, reorganizar o espaço conforme sua preferência, favorecendo tanto a leitura do enunciado, do código e a análise dos resultados.

Na subseção 3.9.2, serão apresentados os principais comandos e recursos disponíveis na interface, como execução e depuração do código, controle de versões, além das opções para salvar e entregar atividades.

Na subseção 3.9.3, será explicado como o sistema organiza e disponibiliza o histórico de entregas, permitindo que o aluno acompanhe sua evolução ao longo do tempo.

Na subseção 3.9.4 serão descritos os mecanismos para evitar perda de progresso e envios desnecessários na plataforma.

Na subseção 3.9.5, serão abordados os mecanismos implementados para que o aluno tenha maior controle sobre as modificações feitas no código, incluindo opções para salvar, descartar ou recuperar versões anteriores.

3.9.1 Tela redimensionável

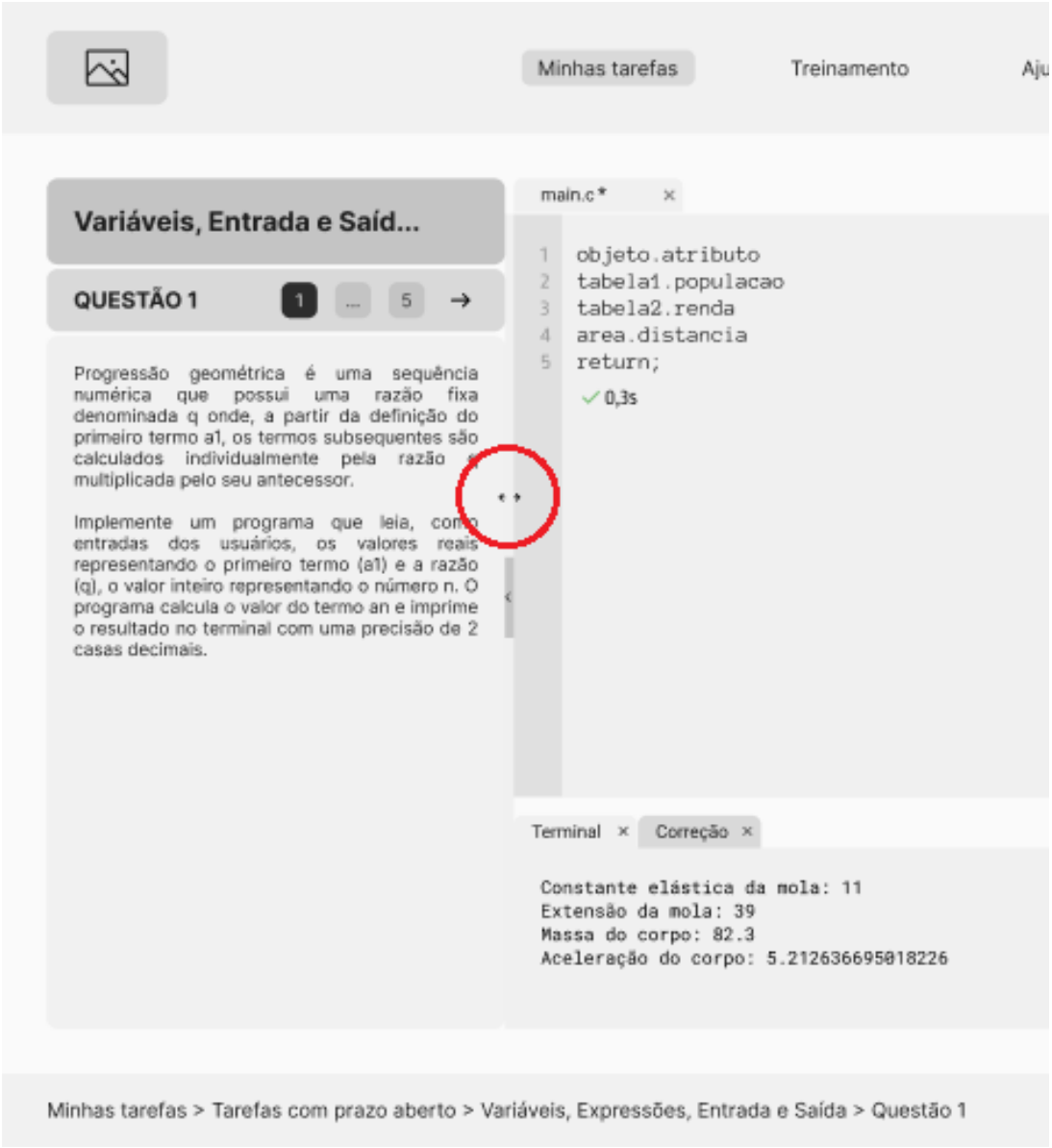
O sistema contará com novas funcionalidades pensadas para tornar a experiência do aluno mais fluida e intuitiva, oferecendo uma interface flexível e adaptável às suas necessidades. Entre as melhorias, será possível minimizar ou maximizar tanto a seção de enunciado quanto a área de código com apenas um clique na coluna que as separa (ver Figura 3.17). Além disso, o aluno poderá redimensionar essas áreas facilmente: basta posicionar o cursor do *mouse* entre as duas seções, clicar e arrastar para direita ou para esquerda (ver Figura 3.18).

Figura 3.17 – Opção para redimensionar com clique.

Outra novidade é a possibilidade de ajustar o tamanho da área de código e da saída de resultados também clicando e arrastando o ícone (ver Figura 3.19), tornando a interface ainda mais adaptável. Com essa funcionalidade, cada aluno poderá distribuir o espaço conforme suas necessidades: quem precisa focar mais na programação poderá ampliar a área do código, enquanto aqueles que desejam analisar a saída dos testes com mais clareza terão a opção de aumentar essa seção. Essa flexibilidade proporciona um ambiente mais confortável para o desenvolvimento e leitura do código, independentemente do dispositivo utilizado.

Para manter a interface limpa e organizada, o ícone de redimensionamento ficará oculto por padrão e será exibido apenas quando o aluno posicionar o cursor sobre a área correspondente. Essas melhorias permitirão um controle maior sobre a disposição dos elementos na tela, facilitando

Figura 3.18 – Redimensionar arrastando entre enunciado e área de código.



a organização do ambiente de estudo.

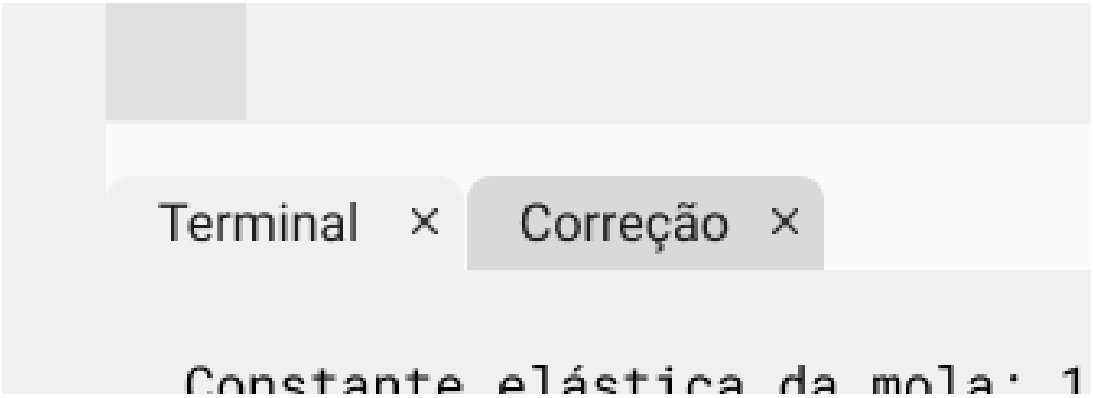
Além disso, um novo ícone será adicionado para permitir que o aluno feche a área de saída sempre que necessário (ver Figura 3.20), oferecendo ainda mais controle sobre a interface. Essa melhoria será útil em dispositivos com telas menores, onde cada centímetro da interface faz diferença para a experiência do aluno. Dessa forma, o sistema se tornará ainda mais organizado e eficiente, permitindo que o aluno adapte o ambiente de estudo da melhor forma possível para seu aprendizado.

3.9.2 Ferramentas do editor

Diversas ferramentas e comandos presentes nos editores de código-fonte facilitam a vida dos programadores, ajudando no gerenciamento do código, na correção de erros e no

Figura 3.19 – Redimensionar arrastando entre área de código e saída de resultados.

Figura 3.20 – Opção para fechar a saída.



acompanhamento do processo de desenvolvimento. Entre as funcionalidades mais usadas, estão aquelas que permitem testar um programa, corrigir problemas, salvar alterações e acompanhar o histórico de modificações.

A Figura 3.21 mostra uma série de ícones correspondente a cada ferramenta, cada um

identificado por um número.

Figura 3.21 – Ferramentas do editor de código-fonte.

Abaixo, uma legenda numerada corresponde a cada ferramenta, explicando seu funcionamento na nova tela de edição de código-fonte. Além disso, são detalhadas suas funcionalidades e como elas contribuem para uma experiência mais intuitiva, organizada e eficiente no desenvolvimento dos programas.

1. **Executar programa atual:** Essa funcionalidade inicia a execução do código sem depuração, permitindo rodar o programa diretamente para verificar sua saída e comportamento. O programa é executado normalmente, e o aluno pode verificar sua saída e comportamento conforme esperado.

2. **Depurar programa atual:** Ao invés de simplesmente rodar o código, essa opção executa o programa em modo de depuração, permitindo que o aluno acompanhe a execução passo a passo, identifique erros, inspecione variáveis em tempo real e analise o fluxo do programa, facilitando a correção de *bugs*.

3. **Entrar:** Durante a depuração, essa funcionalidade permite entrar na execução de uma função ou método chamado na linha atual do código. Ao encontrar uma chamada de função, o depurador não apenas executa essa linha, mas entra na função para mostrar cada uma de suas instruções, ajudando o aluno a entender melhor seu funcionamento.

4. **Sair:** Durante a depuração, essa funcionalidade permite que o aluno continue a execução do código até que a função ou método atual seja finalizado, retornando ao nível anterior da pilha de chamadas. Isso é útil para passar para a próxima etapa sem analisar cada linha da função, quando já se compreendeu o comportamento dela.

5. **Parar/reiniciar *backend*:** essa opção interrompe a execução do *backend* e o reinicia, o que pode ser necessário para fazer alterações de código ou corrigir falhas inesperadas sem precisar fechar e reabrir todo o ambiente de desenvolvimento.

6. **Salvar:** Essa funcionalidade grava as alterações feitas no código ou documento atual, garantindo que as modificações não sejam perdidas caso o sistema seja fechado ou ocorra alguma falha inesperada.

7. **Descartar alterações:** Quando o aluno fizer modificações no código, mas decidir que deseja voltar à versão anterior, essa opção permite reverter todas as mudanças feitas desde o último salvamento, retornando o arquivo à versão anterior.

8. **Histórico de entregas:** Essa funcionalidade exibe um registro das versões enviadas ao longo do tempo, permitindo que o aluno acompanhe as modificações, compare versões anteriores, e se necessário, recupere alguma versão antiga para reverter alterações indesejadas.

9. **Entregar:** A opção de entrega permite enviar a versão atual do código para o sistema de controle de versões, corrigindo a solução entregue e atribuindo uma nota, além de garantir que as mudanças fiquem registradas e disponíveis para consulta posterior, tornando esta versão a mais recente no histórico de versões.

3.9.3 Histórico de versões

A funcionalidade de histórico de versões do *opCoders Judge* apresentada na Figura 3.22, permite que o aluno acompanhe seu progresso detalhadamente. Cada submissão de código é registrada como uma nova tentativa, nomeada com data e hora, e associada a um histórico de versões.

As submissões são organizadas cronologicamente, da mais recente para mais antiga. A versão mais recente é destacada em negrito com a indicação "versão atual" entre parênteses, facilitando a localização rápida da versão mais recente. Com isso, fica simples revisar tentativas anteriores e acompanhar as mudanças feitas ao longo do tempo.

Figura 3.22 – Histórico de entregas fora/dentro do prazo.

A tela da funcionalidade é dividida em uma ou duas seções, dependendo do *status* prazo de envio.

1. **Enquanto o tempo de envio estiver aberto:** A tela exibe uma única seção, onde todas as tentativas de submissão são registradas.

2. **Após o encerramento do tempo de envio:** A tela passa a exibir duas seções caso o aluno tente fazer algum envio:

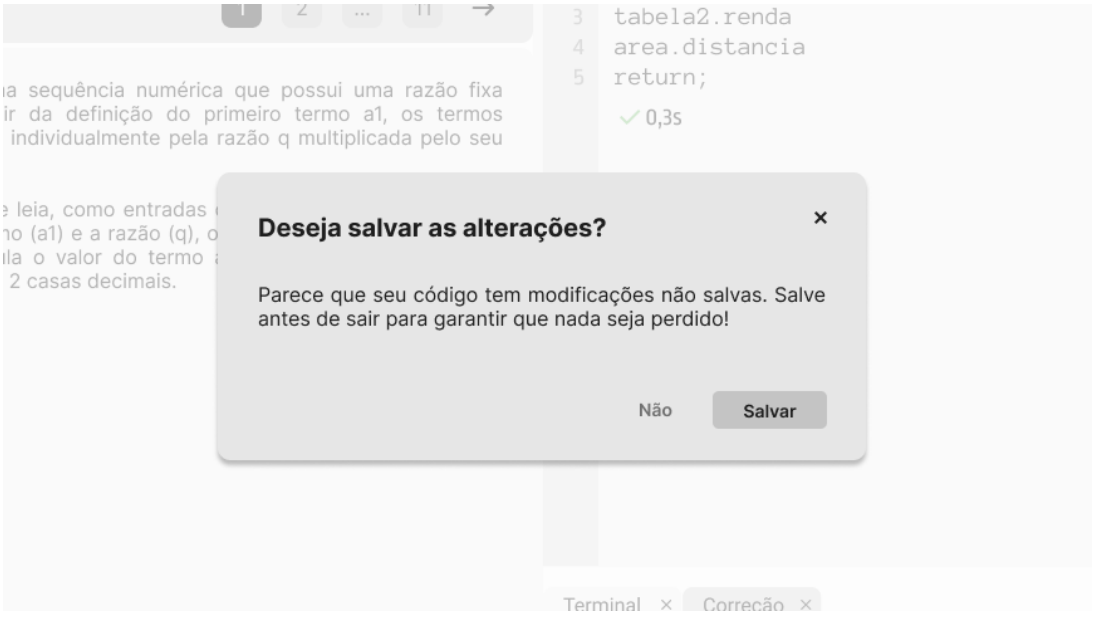
    - **Parte inferior:** Contém todas as submissões feitas dentro do prazo, sendo a última submissão dentro desse período é a que será considerada para pontuação oficial.
    - **Parte superior:** Permite que o aluno continue enviando novas versões do código, recebendo correções e feedback, mas essas novas versões não influenciam na nota final, apenas servindo para aprendizado e aperfeiçoamento contínuo, possibilitando que o aluno experimente melhorias sem a pressão da avaliação.

Além disso, o aluno pode comparar versões anteriores com a atual, facilitando a identificação de melhorias, regressões ou padrões de erro. Esse recurso é vantajoso tanto para o aluno, que acompanha sua evolução e experimenta diferentes abordagens, quanto para professores e tutores, que podem utilizar esse histórico para ter uma visão mais precisa da trajetória do aluno e de sua capacidade de resolver problemas progressivamente.

3.9.4 Envio de atividade e prevenção a perda de progresso

O botão "Entregar" só fica ativo se houver alterações no código desde a última submissão para evitar envios desnecessários, como, por exemplo, o aluno submeter tentativas repetidas que não adicionam nada de novo ao histórico da atividade. Quando o aluno clica em "Entregar" a atividade é salva e enviada automaticamente, tornando-se a versão atual. Se preferir continuar depois, há a opção de apenas salvar o progresso sem enviar. Caso o aluno tente sair da plataforma ou abrir uma versão anterior sem salvar, aparece uma mensagem de confirmação, como pode ser visto na Figura 3.23, para garantir que nada seja perdido por acidente.

Figura 3.23 – Caixa de diálogo com ações para salvar ou não o código.



Na caixa de diálogo, a ação principal está diretamente relacionada à tarefa a ser realizada, enquanto a ação secundária funciona como uma medida preventiva para evitar erros, caso o aluno tenha clicado por engano ou apenas por curiosidade

3.9.5 Edição e controle de alterações

O recurso de edição e controle de alterações melhora a experiência do aluno ao escrever, modificar e organizar seu código antes de submetê-lo, reduzindo erros e melhorando o fluxo de trabalho dentro da plataforma. Para isso, a interface mostra indicações visuais simples e eficazes, como um asterisco que aparece ao lado do identificador do código sempre que houver modificações não salvas, como pode ser visto na Figura 3.24 lembrando o aluno de salvar antes de continuar, e com isso, evitar a perda de progresso e tornar o processo mais organizado.

Figura 3.24 – Identificador indicando modificações não salvas.

O código pode ser editado diretamente na interface da plataforma antes de ser enviado para uma nova correção, sem a necessidade de copiar e colar trechos em outro lugar. Caso o aluno mudar de ideia ou quiser desfazer as mudanças feitas após a última submissão, basta clicar no ícone de descartar alterações, sem o risco de que nenhuma mudança fique salva por engano.

Esse recurso oferece mais controle sobre as edições, e mais autonomia no processo de aprendizado. O aluno pode revisar o código com calma antes da submissão, evitando envios apressados e retrabalho. Além disso, elimina a necessidade de reescrever trechos ou recuperar códigos perdidos, tornando a experiência mais prática e eficiente.

4 Considerações Finais

Este trabalho traz uma contribuição importante ao melhorar a usabilidade da plataforma, aplicando princípios de *design* centrado no usuário para tornar a interação mais simples e o aprendizado mais eficiente. Entre as melhorias, destaca-se a projeção da interface para um futuro sistema de recomendação de exercícios, que será capaz de se adaptar ao desempenho dos alunos. Além disso, foi projetado um editor de código-fonte que não apenas permite a escrita de código, mas também auxilia no acompanhamento de versões, facilita a depuração e incentiva boas práticas de desenvolvimento. As telas foram reorganizadas para melhorar a navegação, e os gráficos nas telas de "Minhas Tarefas", "Atividade Aberta" e "Perfil" auxiliam os alunos a entender melhor o seu progresso e a organizar suas atividades de forma mais eficiente. Essas mudanças tornam a experiência mais agradável e incentivam a continuidade no uso da plataforma, criando um ambiente de aprendizado mais motivador e dinâmico.

A Seção 4.1 apresenta uma conclusão sobre o trabalho desenvolvido e a Seção 4.2 apresenta sugestões de trabalhos futuros.

4.1 Conclusão

Este trabalho alcançou uma série de objetivos específicos, explorados ao longo do desenvolvimento da plataforma. A reestruturação das telas e a inclusão de gráficos informativos foram projetadas para melhorar a organização visual e a clareza das informações. Embora ainda seja necessário avaliar se os usuários percebem essas melhorias de fato, as mudanças visam tornar o processo de aprendizado mais eficiente. Além disso, foi proposta a interface para um sistema de recomendação de exercícios, que se baseia no desempenho dos alunos, cuja implementação e avaliação de eficácia serão etapas futuras no desenvolvimento da plataforma. Também foi prototipado um editor de código-fonte com funcionalidades que otimizam desde a execução e o teste dos programas até o controle de versões e a organização da interface, tornando o processo de programação mais fluido, estruturado e intuitivo.

Os resultados demonstram que as mudanças implementadas não apenas tornarão a navegação mais fluida, mas também facilitarão a compreensão do progresso dos alunos, aumentando sua motivação e comprometimento com os estudos. Isso cria um ciclo positivo de aprendizado, onde os alunos se sentirão mais confiantes e engajados. As novas interfaces permitem uma visualização mais intuitiva do desempenho, com o sistema de recomendação de exercícios auxiliando na personalização do aprendizado. Além disso, o editor de código-fonte facilita a escrita, organização e depuração do código, proporcionando um fluxo de trabalho mais ágil e tornando o aprendizado mais claro e acessível.

Além disso, as melhorias na usabilidade podem resultar em um aumento considerável na retenção dos alunos, uma vez que um ambiente de aprendizado mais acolhedor reduz a frustração e a desmotivação. Ao simplificar a navegação na plataforma, a experiência de aprendizado torna-se mais agradável e envolvente. Assim, o objetivo principal deste trabalho foi atingido: a nova versão da plataforma não só conseguirá atender melhor às expectativas dos usuários, mas também contribuir para a formação de profissionais mais bem preparados e comprometidos na área de tecnologia. Essa evolução também abre caminho para inovações futuras, garantindo que a plataforma se mantenha adaptável às constantes mudanças do cenário educacional.

4.2 Trabalhos Futuros

Para trabalhos futuros, sugerimos:

- **Assistente virtual em tempo real**: irá fornecer suporte imediato, auxiliando os alunos com dúvidas durante o estudo, proporcionando um ambiente mais interativo e dinâmico.
- **Banco de questões para os professores**: auxiliará na criação de avaliações mais variadas e eficazes, otimizando o processo de ensino.
- **Elementos de gamificação**: propostas para tornar a experiência de aprendizado mais interativa, estimulando a participação ativa dos alunos e tornando o processo de aquisição de conhecimento mais divertido e eficaz.
- **Implementação da nova interface**: a nova interface projetada deve ser implementada em uma nova versão do *opCoders Judge*.
- **Testes com usuários**: embora as telas tenham sido revisadas e novas funcionalidades tenham sido adicionadas, a confirmação do sucesso dessas mudanças depende de avaliações futuras e do *feedback* dos usuários. Assim, os impactos dessas melhorias na clareza e organização da interface ainda precisam ser analisados com mais profundidade.

Referências

BRITO, P. S. S. *O uso de ferramentas computacionais para o ensino de programação para alunos de engenharia.* 2019. Monografia de Bacharelado - Universidade Federal de Ouro Preto.

BROOKE, J. Sus: A quick and dirty usability scale. *Usability Evaluation in Industry*, 1996.

CEDRAZ, V. F. *Uma avaliação de usabilidade do corretor de exercícios de introdução à programação opCoders Judge*. 2023. Monografia de Bacharelado - Universidade Federal de Ouro Preto.

DAM, R. F.; SIANG, T. Y. Design thinking: Get started with prototyping. *Interaction Design Foundation*, 2020.

MENDONçA, T. S. *Projeto e desenvolvimento de uma plataforma de gestão de questões dinâmicas para o ensino de programação de computadores.* 2023. Monografia de Bacharelado - Universidade Federal de Ouro Preto.

NIELSEN, J.; MOLICH, R. Heuristic evaluation of user interfaces. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. [S.l.: s.n.], 1990. p. 249–256.

NORMAN, D. *The design of everyday things: Revised and expanded edition*. [S.l.]: Basic books, 2013.

PATROCíNIO, J. A. d. *OpCoders Judge: uma versão online para o corretor automático de exercícios de programação do projeto opcoders.* 2023. Monografia de Bacharelado - Universidade Federal de Ouro Preto.

SILVA, L. L. M. *OpCoders Judge: aprimorando o corretor automático de exercícios de programação com base em testes de usabilidade*. 2023. Monografia de Bacharelado - Universidade Federal de Ouro Preto.

ZIMERMAN, F. E. *Explorando chatbots baseados em LLM para criação automática de questões práticas de programação de computadores*. 2024. Monografia de Bacharelado - Universidade Federal de Ouro Preto.