Aplicativo móvel para ensino de computação

Bruna F. Silva[1], **Leandro F. Schweitzer**[1],
Fernando W. Albiero[1], **Morganna Giovanelli**[2]

[1]Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina (IFSC)
Rua Heitor Villa Lobos, 225 – São Francisco – 88506-400 – Lages – SC – Brasill

[2]BotCity Desenvolvimento de Sistemas LTDA
Rua Capitão Augusto Sales Pupo, 93 - Jardim Chapadão
Campinas - São Paulo - SP - Brasil

{bruna.f1991, leandro.f1998}@aluno.ifsc.edu.br,

fernando.albiero@ifsc.edu.br, morgcontato@gmail.com

Abstract. *This paper focuses on developing a mobile device as a support tool for applying theoretical and practical computing concepts through trails with multiple-choice questions and answers. Gamification elements were incorporated to engage and motivate users. At the end of the project development, a survey was carried out to evaluate the application, revealing satisfactory results.*

Resumo. *Este artigo concentra-se no desenvolvimento de um dispositivo móvel como uma ferramenta de apoio para a aplicação de conceitos teóricos e práticos de computação por meio de trilhas com perguntas e respostas de múltipla escolha. Elementos de gamificação foram incorporados para engajar e motivar os usuários. Ao final do desenvolvimento do projeto, foi realizada uma pesquisa para fins de avaliação do aplicativo, revelando resultados satisfatórios.*

1. Introdução

O Instituto Federal de Santa Catarina oferece diversos cursos em diferentes modalidades de ensino, incluindo o curso de graduação em ciência da computação, que forma profissionais na área de tecnologia da informação.

O curso apresenta um Projeto Pedagógico de Curso (IFSC, 2022) que engloba uma matriz curricular organizada de tal maneira que a aprovação em determinadas disciplinas é um requisito prévio para a progressão em outras. No Quadro 1, consta a referência das disciplinas. Além disso, o PPC também descreve que o processo de admissão no curso ocorre anualmente. Como resultado, a oferta de disciplinas segue o mesmo padrão anual, disponibilizando 40 vagas adicionais com uma carga horária total de curso de 3200 horas, divididas em 8 fases. Diante dessa configuração, caso um estudante seja reprovado em uma disciplina, ele terá que aguardar até o próximo ano para cursá-la novamente, o que pode acarretar atrasos na conclusão do curso e constituir um fator de evasão para o estudante.

Durante o percurso acadêmico, alguns estudantes enfrentam desafios no contexto da aprendizagem. Essas dificuldades podem surgir de várias fontes, abrangendo desde a falta de compreensão e interpretação lógica de determinado tema até a escassez de

experiências práticas por parte dos estudantes. Essas questões resultam em disciplinas com altas taxas de reprovação, que, por sua vez, levam à formação de turmas numerosas e contribuem para a escassez de vagas na matrícula dos estudantes. Conforme apontado por Gomes et al. (2008), as taxas de evasão e repetência em disciplinas introdutórias de programação são consideravelmente elevadas nos cursos da área de tecnologia.

A aprendizagem inicial de programação é considerada um processo no qual os estudantes a encaram como um verdadeiro teste de vocação. Os estudantes que têm pouco ou nenhum sucesso nesse processo geralmente o abandonam e começam a olhar para outras possibilidades, às vezes, até para a mudança de carreira (da Costa Chagas e De Oliveira, 2011).

	Nome
1	Introdução à programação
2	Programação Orientada a Objetos
3	Arquitetura de Computadores
4	Linguagens e Paradigmas de Programação
5	Desenvolvimento de Aplicações Orientada a Objetos
6	Estrutura de Dados
7	Grafos
8	Teoria da Computação

Quadro 1: Disciplinas de computação que são pré-requisito.

As Tecnologias de Informação e Comunicação (TICs), segundo Farias (2013), possuem a propósito de aprimorar o processo educacional, proporcionando facilidade nos os processos de interação, fomentar os recursos didáticos e auxiliar na inclusão digital.

A expansão do acesso aos dispositivos móveis em escala global tem ocasionado transformações na produção e compartilhamento do conhecimento, oferecendo diversas possibilidades para a aprendizagem, com base na mobilidade dos dispositivos, dos estudantes, dos conteúdos e no acesso ao conhecimento em qualquer momento e local (Rafaela e Breno, 2014).

Considerando que os estudantes possuem diferentes estilos de aprendizagem e que os dispositivos móveis estão cada vez mais presentes em seu dia a dia, um aplicativo móvel pode se tornar uma opção viável de metodologia ativa para o aprendizado de temas relacionados à computação, permitindo que o estudante assuma o papel de protagonista em seu processo de aprendizagem.

O objetivo geral deste projeto consiste no desenvolvimento de um aplicativo móvel que proporcione uma abordagem teórica e prática de tópicos específicos relacionados às disciplinas do curso de Ciência da Computação. Dessa maneira, os estudantes terão à disposição uma alternativa de estudo fora do ambiente de sala de aula, além da flexibilidade de poderem estudar em qualquer lugar e a qualquer momento.

Para solução do problema acima citados, foram estabelecidos os seguintes objetivos específicos:

• Identificar os tópicos que apresentam maiores desafios aos estudantes dentro do curso;

• Codificar um aplicativo móvel apresentando os tópicos em forma de questões objetivas;

• Disponibilizar o aplicativo móvel para estudantes do curso de Ciência da Computação;

• Avaliar o aplicativo móvel desenvolvido.

O processo de desenvolvimento deste trabalho consiste em quatro etapas principais. Na primeira etapa, serão conduzidos levantamentos no histórico do Campus, a fim de obter informações referentes ao número de estudantes que não obtiveram aprovação nas disciplinas pré-requisito para o curso de Ciência da Computação. Diante desses dados, será feita a escolha da disciplina alvo da aplicação. Na segunda etapa, será realizado o processo de prototipação e posterior desenvolvimento do aplicativo, empregando a metodologia ativa conhecida como gamificação para abordar os tópicos da disciplina selecionada. Na terceira etapa, o aplicativo será disponibilizado para o público-alvo, constituído pelos estudantes da área de computação, com o propósito de permitir que eles experimentem e utilizem o aplicativo, a fim de prosseguir para a próxima etapa. Na quarta e última etapa, será realizada uma pesquisa por meio de um formulário contendo perguntas pertinentes para avaliar o projeto. Este formulário será disponibilizado aos estudantes que fizeram uso do aplicativo, e as respostas deles poderão ser utilizadas para orientar melhorias futuras no software.

O trabalho é uma pesquisa aplicada, do ponto de vista de abordagem do problema é qualitativa, do ponto de vista de objetivo é exploratória e dos procedimentos técnicos é bibliográfica.

Este artigo está estruturado em quatro seções distintas. A primeira seção consiste em uma introdução que contextualiza o tema, abordando as possíveis dificuldades de aprendizagem por parte dos estudantes. A segunda seção apresenta o embasamento teórico, explorando pesquisas sobre os benefícios da aplicação de metodologias ativas no processo de ensino, revisando trabalhos similares e definindo os recursos empregados no desenvolvimento do aplicativo móvel. A terceira seção descreve detalhadamente o processo de desenvolvimento do aplicativo. Por fim, a última seção apresenta as considerações finais sobre este trabalho, incluindo os testes e avaliações realizados.

2. Referencial teórico

Esta seção apresenta temas que enfatizam a relevância do desenvolvimento deste projeto, sendo subdividida em três subseções distintas. A primeira subseção aborda a temática das metodologias ativas de aprendizagem. A segunda subseção trata do desenvolvimento de trabalhos relacionados. Na terceira subseção, são discutidas as tecnologias empregadas no desenvolvimento do presente trabalho.

2.1. Metodologias de aprendizagem ativas

Metodologias de aprendizagem ativas são abordagens pedagógicas que colocam o estudante no centro do processo de aprendizado, incentivando sua participação ativa e engajamento, em contraste com a abordagem passiva comumente adotada, na qual o aluno apenas consome conteúdos. Segundo Berbel (2011), essas metodologias têm como objetivo promover uma aprendizagem mais significativa, envolvendo o aluno de maneira prática e estimulando o desenvolvimento de habilidades cognitivas, sociais e emocionais.

O trabalho de revisão de literatura elaborado por Witt e Kemczinski (2020) aborda as metodologias ativas adotadas no ensino de computação. Com o objetivo de oferecer alternativas para questões como evasão e falta de motivação dos estudantes, foram analisados 46 estudos que aplicaram alguma forma de metodologia ativa. Entre os trabalhos analisados foram citadas as seguintes metodologias:

• Aprendizado Baseado em Problemas (ABP), no qual os estudantes são apresentados a um problema desafiador que requer a aplicação de conhecimentos e habilidades para resolvê-lo. O ABP estimula a autonomia do aluno, uma vez que ele precisa buscar e desenvolver as competências necessárias para solucionar o problema proposto.

• Gamificação, incorpora elementos típicos de jogos, como desafios, recompensas e competições, no ambiente de aprendizado. Por meio dela, o estudante é motivado e engajado, pois seu progresso é acompanhado, conquistas são reconhecidas e metas são estabelecidas. A gamificação cria um ambiente lúdico e estimulante, que aumenta a motivação e torna o processo de aprendizado mais envolvente.

A integração de educação e jogos pode gerar mais engajamento aos estudantes em aprender algo novo. Segundo Zhan et al. (2022), nos últimos anos, tem havido um interesse crescente em usar jogos como uma ferramenta educacional para ajudar os alunos a aprender e auxiliar os professores a ensinar.

Um jogo pode ser definido quer pelo objetivo que os seus jogadores procuram alcançar, quer pelo conjunto de regras que determinam o que podem ou não fazer esses jogadores (Guzzo, 2020). Desta forma é viável organizar o aplicativo móvel em um formato que possibilite a definição de objetivos alcançáveis pelos estudantes, por meio da aplicação de raciocínio lógico às questões apresentadas, promovendo a prática e fixação do conhecimento sobre um assunto.

Tendo em vista as metodologias acima citadas, optou-se pelo uso delas neste trabalho, aplicando a APB através da elaboração de questões objetivas apresentando um contexto, um problema e opções de solução, onde o estudante deve optar pela opção correta, e agregando elementos de gamificação pelos quais o estudante é motivado a engajar, acompanhar seu progresso, reconhecer conquistas e metas estabelecidas, criando um ambiente lúdico e estimulante, que aumenta a motivação e torna o processo de aprendizado mais envolvente.

2.2. Trabalhos relacionados

O trabalho desenvolvido por da Silva e de Oliveira (2022), nomeado **Desenvolvimento de uma Plataforma para Geração de Simulados**, oferece uma maneira complementar de ensino de matemática através de uma plataforma móvel e site. Nele encontramos a metodologia ativa de ensino, utilizando a técnica de gamificação, onde os estudantes podem

realizar *quizzes* com múltiplas escolhas referente a um conteúdo específico disponível na base de dados do aplicativo. A cada resposta, os estudantes recebem um retorno de acerto ou erro da questão, pontuação e ranking, trazendo dinamicidade para a atividade. As avaliações feitas em uma escola pública, por parte dos estudantes foi bastante positiva. De maneira abrangente foi apontado que o aplicativo é amigável e fácil de utilizar, auxilia no aprendizado e na produtividade, facilita e enriquece a compreensão dos conteúdos, além de tornar a aula mais dinâmica. Porém também foi relatado que tanto estudantes quanto professores, dentro do contexto avaliado, nem sempre tem alcance a esse tipo de tecnologia.

O segundo projeto com título **Ensino e aprendizagem de vocabulário da língua inglesa de forma dinâmica: desenvolvimento de aplicativo para dispositivos móveis**, busca por alternativas de atrair o interesse de estudantes e minimizar suas dificuldades com uma língua estrangeira, o trabalho desenvolvido por Gomes e de Castro (2013) propõe explorar a língua inglesa dentro do contexto da área de computação e informática através de um aplicativo móvel. O aplicativo permite escolher o nível de dificuldade e os temas (Fundamentos da computação, Linguagens de programação, Redes, Sistemas Operacionais e Banco de Dados), em seguida apresenta funcionalidades nos seguintes tópicos:

• Introdução: a qual apresenta conceitos básicos sobre um tema escolhido através de imagens e textos;

• *Quiz* com perguntas de múltipla escolha;

• *Everyday* apresenta situações do cotidiano de profissionais da área com perguntas e respostas.

Como terceiro exemplo, a ferramenta colaborativa desenvolvida por Machado et al. (2018), **Uma ferramenta colaborativa para apoiar a aprendizagem de programação de computadores** tem a intenção de reduzir as dificuldades encontradas pelos estudantes no aprendizado de programação. Além da abordagem colaborativa, há investigação de meios de tornar o processo de aprendizado mais flexível, acessível e inclusivo. Com o objetivo de tentar auxiliar no ensino de programação, o desenvolvimento de um aplicativo móvel que facilita a construção de um ambiente virtual, no qual possam ser aplicadas técnicas de metodologia ativa, os tipos de atividades disponíveis são: questões de múltipla escolha, quebra-cabeça de algoritmo e problemas abertos. Outra opção importante é a possibilidade de troca de mensagens entre os usuários, facilitando a comunicação e colaboração na construção do conhecimento. Os resultados do uso da ferramenta por uma turma apontou que o processo de aprendizado se tornou mais leve, facilitou o acesso de informações e comunicação. Por outro lado apresentou algumas limitações como falta de sinal Wi-Fi em alguns ambientes, dispositivos não compatíveis e interface inadequada.

O trabalho desenvolvido por Fernando Andrade Matos et al. (2019), como quarto exemplo, tem o objetivo de utilizar a metodologia de aprendizagem gamificação para o apoio do ensino de disciplinas iniciais em cursos de saúde na Universidade Federal de Sergipe. A ferramenta, com o título **LabMorfoQuiz**, foi planejada em forma de *quiz*, com 176 questões de quatro alternativas separadas por disciplina, com tempo de 30 segundos para resposta. Para sua avaliação utilizou o modelo MEEGA+ que fornece um kit pronto

para avaliação de jogos educativos, e a avaliação do jogo foi realizada com 30 discentes do Campus. Os discentes consideraram a experiência positiva, com o jogo intuitivo e fácil de usar. Fornecendo também uma boa fonte de conteúdo auxiliar para a aprendizagem, entendendo-se por fim que o aplicativo pode ser um bom recurso para engajar e motivar os estudantes.

Por fim, o **Aplicativo Exatas: Ferramenta de apoio ao aprendizado móvel** implementado no trabalho de Mocbel e Farias (2018) visa utilizar o *Mobile Learning* para estudantes do ensino médio com conteúdo de exatas. O aplicativo conta com videoaulas, textos expositivos, questões e gráficos de desempenho. Avaliado por uma turma de graduação, foi apontado que mais de 87% dos entrevistados concordam que o software satisfaz o que é proposto e concordam em indicar o mesmo para outras pessoas, respectivamente. Como consideração para trabalhos futuros, tem-se a sugestão de criar uma área do professor dentro do aplicativo.

No quadro 2 é possível visualizar uma análise de comparações de funcionalidades em comum entre este trabalho, chamado de IFSComp dentro do quadro, e os demais trabalhos estudados nessa seção. Visto que o segundo e o terceiro não tinham nome, foi utilizado o nome de seus autores dentro do quadro.

	Atividades em grupo	Vídeo aulas	Visualização do Desempenho	Ranking	Tipo de questões
Simulados	Apresenta	Não apresenta	Quantificado	Apresenta	Objetivas
Gomes	Não apresenta	Não apresenta	Não apresenta	Não apresenta	Objetivas
Machado	Apresenta	Apresenta	Não apresenta	Não apresenta	Diversas
LabMorfo	Apresenta	Não apresenta	Quantificado	Apresenta	Objetivas
Exatas	Apresenta	Apresenta	Quantificado e Gráfico	Apresenta	Objetivas
IFSComp	Não apresenta	Não apresenta	Quantificado	Não apresenta	Objetivas

Quadro 2: Análise Comparativa do IFSComp com demais trabalhos relacionados.

2.3. Tecnologias utilizadas

A seguinte subseção tem como propósito descrever as tecnologias selecionadas para o desenvolvimento do aplicativo móvel. Nesse contexto, foi optado, preferencialmente, ferramentas de uso gratuito, sendo que estas estão exibidas na Figura 1 e serão descritas nas próximas subseções.

2.3.1. React Native

React Native[1] é um *framework* para o desenvolvimento de aplicações móveis, o qual foi escolhido para o desenvolvimento do *frontend* deste trabalho. Utiliza como base o Javascript[2], lançado em 2015. Seu maior diferencial é a capacidade de escrever o código apenas uma vez, sem precisar adaptar individualmente para plataformas como iOS ou Android. Segundo StackOverflow (2022), é o *framework* mais utilizado no desenvolvimento

[1]https://reactnative.dev

[2]https://developer.mozilla.org/pt-BR/docs/Web/JavaScript

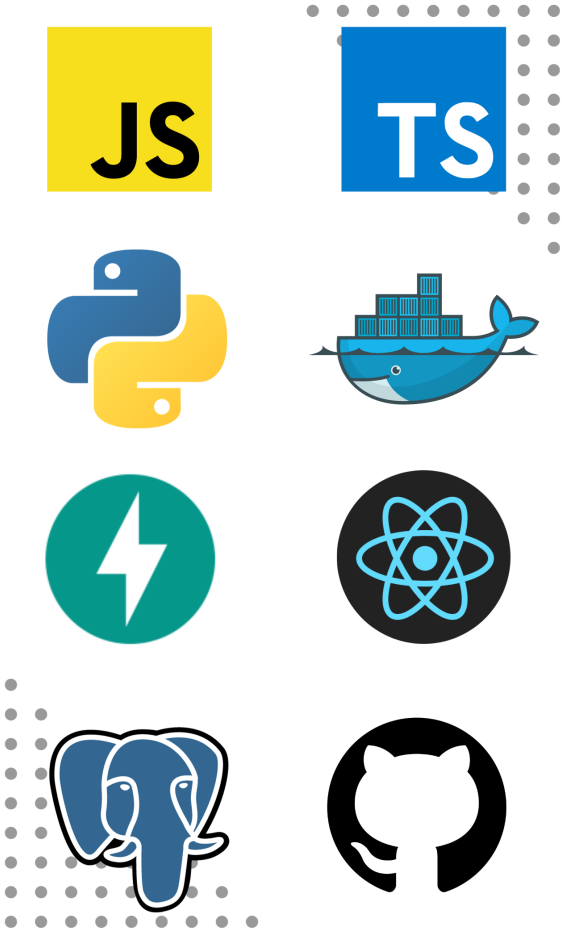

Figura 1. Logos das tecnologias utilizadas.

de aplicativos, o que o torna interessante para o desenvolvimento deste trabalho, tendo em vista também seu foco em agilidade e responsividade.

Em conjunto a ele, será utilizado a biblioteca de componentes Tamagui[3], que facilita a estilização, fornece componentes prontos para o uso que podem ser mais personalizados, além de controle de estilo e sistema de design tipado.

2.3.2. FastAPI

O FastAPI[4] é um *framework* de desenvolvimento de *Application Programming Interface* (API) *RESTful* em Python[5] que se destaca pela sua alta performance e facilidade de uso. Ele permite a criação rápida e eficiente de serviços web, fornecendo uma interface amigável para a construção de *endpoints* com suporte a tipos de dados estáticos, validação automática, documentação interativa e geração automática de esquemas.

Como código aberto, possui uma vasta gama de bibliotecas de terceiros para diversos usos específicos. Também contém diversos recursos para o aumento de produtividade.

[3]https://tamagui.dev
[4]https://fastapi.tiangolo.com
[5]https://docs.python.org/pt-br/3

Esse recurso será utilizado para o desenvolvimento do *backend* do aplicativo móvel.

2.3.3. PostgreSQL

O PostgreSQL[6] é um sistema de gerenciamento de banco de dados relacional e orientado a objetos, de código aberto. Possui recursos em comum a bancos de dados de grande porte. Neste trabalho, será a ferramenta responsável pela camada de persistência de dados do aplicativo, guardando informações sobre os usuários.

2.3.4. GitHub

O Github[7] é um serviço de hospedagem baseado em nuvem que utiliza o Git, uma tecnologia para controle de versão e gerenciamento de código-fonte. Através do *GitHub*, é possível gerenciar informações como os autores do código, as datas de modificação e, além disso, potencializa a colaboração por meio de recursos como o uso de *branches*. Esses *branches* permitem a existência de múltiplas versões do mesmo código, que posteriormente podem ser mescladas utilizando a ferramenta, a qual também registra as alterações realizadas e destaca possíveis conflitos. Dessa forma, o GitHub desempenhará um papel fundamental no versionamento e histórico de desenvolvimento do aplicativo.

2.3.5. Docker

O Docker[8] é uma plataforma de código aberto que é utilizada para implementar aplicativos em *containers* virtuais. Essa técnica de conteinerização viabiliza a execução de diversos aplicativos em ambientes distintos e complexos. Essa tecnologia terá a função de criar o ambiente adequado para a implantação do aplicativo.

2.3.6. Render

O Render[9] é uma plataforma que oferece serviços de hospedagem de aplicativos e *websites* em nuvem, com recursos variados, entre eles abrange as tecnologias utilizadas neste trabalho. Entre as vantagens destaca-se a interface intuitiva, a facilidade no processo de implantação quando vinculada ao *GitHub* e disponibilidade de recursos gratuitos. Será utilizado para a hospedagem do aplicativo e disponibilização do mesmo.

3. Desenvolvimento

Nesta seção serão abordados os tópicos referentes a modelagem e desenvolvimento do aplicativo móvel. Divididas em subseções, a subseção 1 trata dos requisitos, subseção 2 da modelagem do sistema, a subseção 3 apresenta o projeto de banco de dados, a subseção 4 a prototipagem, a subseção 5 o desenvolvimento da *User Interface* (UI) e por fim a subseção 6 aborda o desenvolvimento da *Application Programming Interface* (API).

[6]https://www.postgresql.org
[7]https://github.com
[8]https://www.docker.com
[9]https://render.com

3.1. Requisitos

Dentre as diversas metodologias disponíveis para o levantamento de requisitos de software, optou-se pela utilização da técnica conhecida como *User Stories*. Segundo Pontes e Arthaud (2018), ela é composta por um único parágrafo sucinto que aborda a funcionalidade do sistema, e é concebida de forma a ser apresentada em cartões (*Story Cards*) para facilitar a manipulação. A redação dos cartões deve ser caracterizada por simplicidade, clareza e concisão.

Com isso contamos com três elementos fundamentais para entender a história do usuário: a persona, a ação e o resultado. Por meio desses elementos, é possível capturar as principais necessidades e requisitos do software sob a perspectiva do usuário, além de definir o nível de prioridade associado a cada funcionalidade descrita. A utilização dos cartões auxilia na manutenção do foco nas demandas dos usuários, possibilita o refinamento e detalhamento das funcionalidades, facilita a organização e a priorização das mesmas, bem como contribui para um desenvolvimento mais eficiente e eficaz do software, (Lopes et al., 2019).

3.1.1. Cartões

Entrar no sistema	
Como um	usuário
Eu quero	fazer login no sistema
Para que	possa acessar as funcionalidades
Prioridade	alta

Quadro 3: Requisito entrar no sistema.

Gerenciar acesso	
Como um	usuário
Eu quero	gerenciar meu acesso ao sistema
Para que	possa fazer mudanças quando necessário
Prioridade	baixa

Quadro 4: Requisito gerenciar acesso.

Acesso ao conteúdo	
Como um	usuário
Eu quero	ter acesso às explicações do conteúdo
Para que	possa aprender a teoria
Prioridade	alta

Quadro 5: Requisito acesso ao conteúdo.

Questões práticas	
Como um	usuário
Eu quero	exercitar com questões práticas
Para que	possa testar o conhecimento e fixar o que foi aprendido
Prioridade	alta

Quadro 6: Requisito questões práticas.

Verificação de respostas	
Como um	usuário
Eu quero	verificar se acertei ou errei as questões
Para que	possa fixar melhor o conteúdo
Prioridade	alta

Quadro 7: Requisito correção de respostas.

Pontuação no aplicativo	
Como um	usuário
Eu quero	verificar minha pontuação no aplicativo
Para que	possa me sentir motivado a continuar usando o aplicativo
Prioridade	baixa

Quadro 8: Requisito pontuação no aplicativo.

3.2. Prototipagem

A prototipação de um aplicativo é uma etapa crucial no processo de desenvolvimento, pois permite que as ideias e conceitos sejam testados e refinados antes da implementação final.

Ao criar os protótipos, é importante levar em consideração a experiência do usuário. Isso significa que os designers devem se concentrar em tornar a navegação e a interação com o aplicativo o mais intuitivas e eficientes possível. Segundo de Abreu Cybis et al. (2015) as interfaces se assemelham a ferramentas que consistem em áreas de trabalho, controles e comandos, sendo que o desenvolvimento deve priorizar a maneira como o usuário utiliza essa ferramenta.

Os protótipos podem incluir telas diferentes e fluxos de trabalho para representar as várias funcionalidades do aplicativo, também ajudam a visualizar o layout do aplicativo, incluindo a disposição dos elementos na tela, as cores, as fontes e os ícones utilizados. Esses aspectos visuais são importantes para criar uma identidade visual coesa e atraente para o aplicativo. Além disso, os protótipos podem mostrar como os diferentes elementos interagem entre si, como botões, menus e campos de entrada de dados, permitindo que os usuários tenham uma compreensão clara de como utilizar as diferentes funcionalidades. (Pressman e Maxim, 2021)

A criação de protótipos ajuda a identificar possíveis problemas e desafios antes da implementação do aplicativo. Por exemplo, os protótipos podem revelar fluxos de

trabalho complexos ou confusos, problemas de usabilidade, falta de consistência visual ou interações ineficientes. Ao identificar esses problemas precocemente, a equipe de desenvolvimento pode realizar ajustes e melhorias para garantir que o aplicativo final ofereça uma experiência de alta qualidade. (Soares e Resende, 2017)

Na Figura 22, é apresentada a tela inicial do aplicativo, que geralmente é a primeira tela que os usuários veem ao abrir o aplicativo. Nessa tela, são exibidos os botões de cadastro de usuário e login, que são elementos essenciais para a entrada no sistema.

Ao clicar no botão de cadastro de usuário, os usuários são redirecionados para a tela de cadastro, onde são solicitadas informações como nome de usuário, e-mail e senha. Esses campos servem para coletar os dados necessários para criar uma conta de usuário no aplicativo. Ao preencher essas informações e confirmar o cadastro, os usuários podem utilizar as funcionalidades oferecidas pelo aplicativo.

Já na tela de login, os usuários precisam fornecer seu e-mail e senha para acessar sua conta. Essa tela de login é uma medida de segurança para garantir que apenas usuários autorizados possam acessar os recursos e informações do aplicativo.

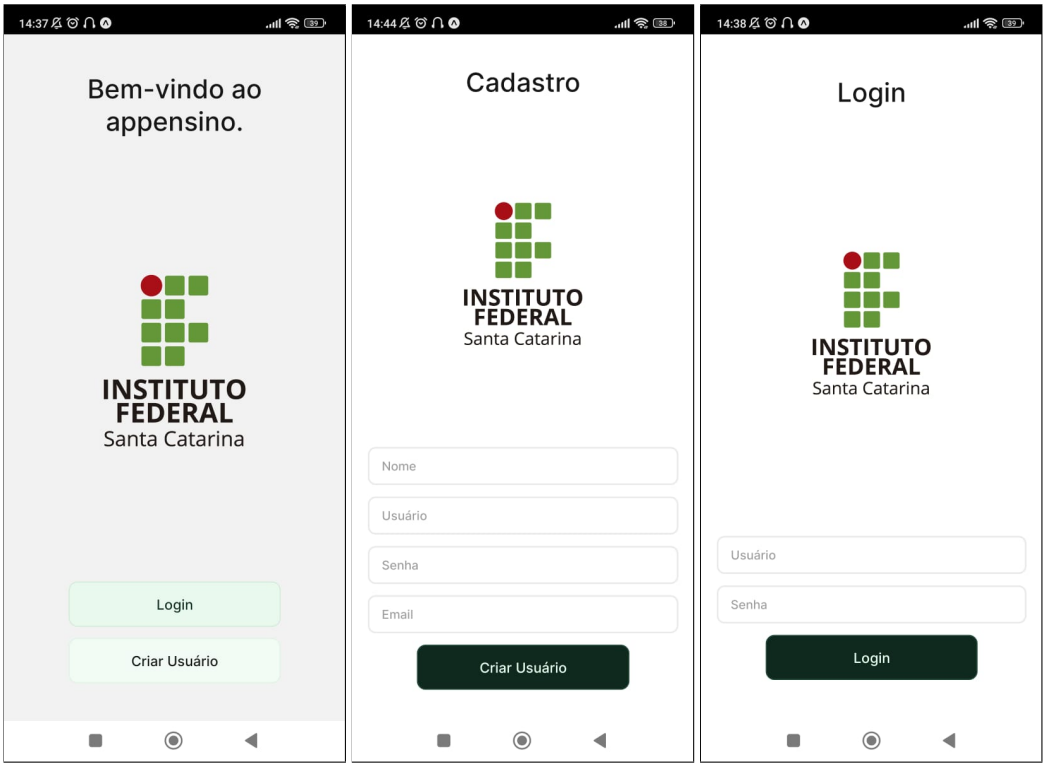

Figura 2. Protótipos de tela inicial, cadastro e login.

Na Figura 3, é apresentada a tela na qual o aluno pode escolher qual das trilhas de ensino deseja começar ou continuar. Essas trilhas são exibidas em cards, que contêm um título e uma breve descrição para cada uma delas. Essa tela tem como objetivo fornecer aos alunos uma visão geral das opções disponíveis e permitir que eles selecionem a trilha de ensino de seu interesse.

Além disso, a Figura 3 também apresenta uma tela de exemplo de conteúdo e uma

tela de exemplo de questão. Essas telas demonstram como o conteúdo do aplicativo é apresentado aos usuários e como as questões são exibidas para avaliar o conhecimento adquirido.

A tela de exemplo de conteúdo pode conter informações como texto, imagens, vídeos ou outros recursos relevantes para o aprendizado. Essa tela oferece aos alunos acesso ao material didático necessário para compreender e assimilar os conceitos ensinados na trilha selecionada.

Já a tela de exemplo de questão permite que os alunos testem seu conhecimento através de perguntas ou exercícios relacionados ao conteúdo estudado. Essa interação promove a participação ativa do aluno no processo de aprendizagem e permite que ele avalie seu progresso. Os botões de voltar à seleção de trilhas e a barra de progresso ajudam a navegar e acompanhar o progresso do usuário dentro do aplicativo.

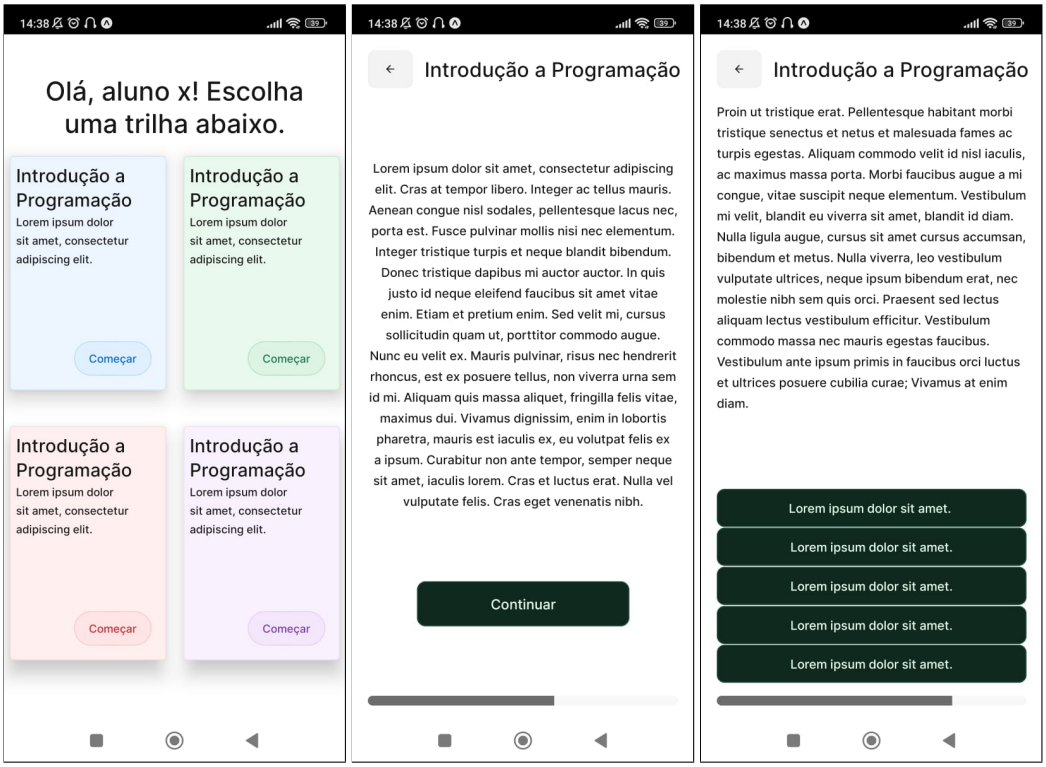

Figura 3. Protótipos de tela de trilhas, introdução ao conteúdo.

3.3. Projeto de banco de dados

Com base nos requisitos e prototipagem, foi definida a persistência de alguns dados dos usuários, necessários para o ele criar, logar e acompanhar o seu progresso dentro do aplicativo. Os dados de e-mail e senha são utilizados para login no sistema, o nome é utilizado como uma forma de identificação o usuário no aplicativo, os pontos são somados durante a realização do *quiz* e são armazenados juntamente com um identificador da trilha que foi realizada.

O banco de dados utilizado foi o PostgreSQL, banco relacional para armazenar

informações estruturadas, como mostra na Figura 4. A comunicação é feita através da API desenvolvida com FastAPI.

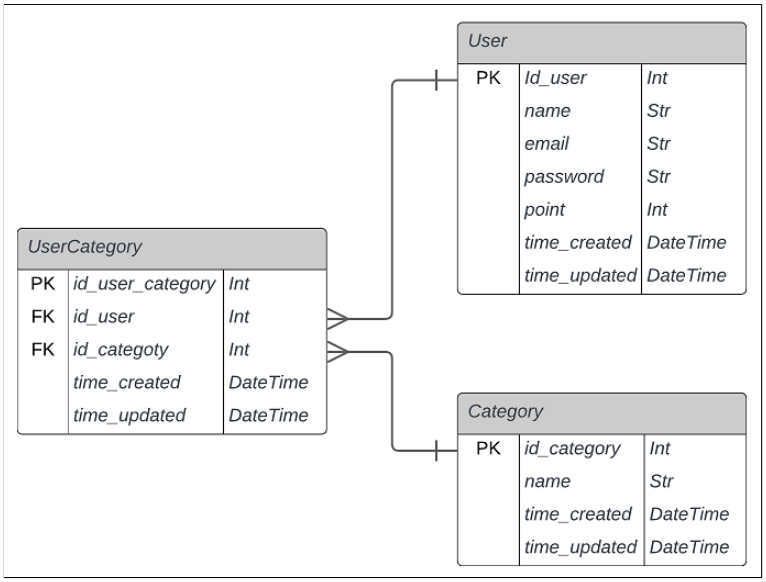

Figura 4. Banco de dados.

3.4. Desenvolvimento da *User Interface*

A *User Interface* (UI) deste projeto foi construída com base no *framework* React Native, pela necessidade de criar uma experiência de usuário rica e nativa em dispositivos móveis, tanto para iOS quanto para Android.

Optou-se por incorporar o Expo[10] ao projeto devido às diversas simplificações que ele oferece. O Expo é uma plataforma que facilita a criação de aplicativos React Native, eliminando muitas das complexidades de configuração que geralmente estão associadas ao desenvolvimento móvel.

Para gerenciar a navegação entre as páginas da aplicação de forma intuitiva, foi adotado o Expo-Router[11]. Essa biblioteca simplifica a criação de sistemas de navegação, garantindo que os usuários possam transitar facilmente entre as diferentes telas do aplicativo, proporcionando uma experiência de usuário coesa e amigável.

Também a escolha da Context API[12] para o gerenciamento de estado global foi motivada pela necessidade de compartilhar informações críticas entre os componentes da UI sem a complexidade de passar manualmente propriedades.

Portanto, a combinação de React Native como base, o Expo para simplificar o desenvolvimento, o Expo-Router para navegação intuitiva e a Context API para o gerenciamento de estado global, foi uma escolha estratégica que permitiu a criação de uma UI robusta, eficiente e repleta de recursos.

[10]https://expo.dev
[11]https://docs.expo.dev/routing/introduction
[12]https://react.dev/reference/react/useContext

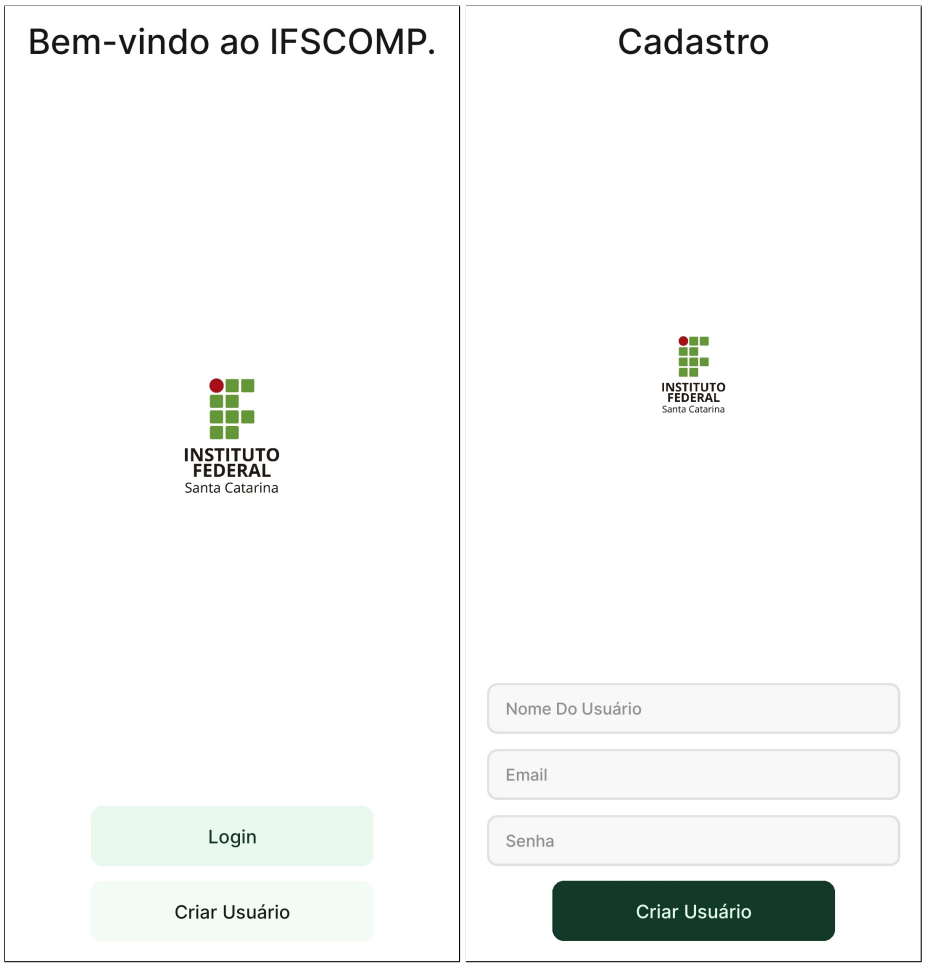

Figura 5. Tela inicial, cadastro.

3.4.1. Autenticação

No processo de cadastro de usuário, o sistema realiza uma verificação para garantir que não exista previamente um usuário com o mesmo endereço de e-mail. Esse processo ocorre ao enviar uma requisição com as informações do usuário que deseja se cadastrar, o dado de e-mail é verificado nos registros do banco de dados, retornando uma resposta de validação. Se não houver nenhuma correspondência nos registros, um novo usuário é criado e suas informações são armazenadas no banco de dados, caso contrário, é emitido um alerta de e-mail já cadastrado.

Já no momento do login, o sistema verifica a coincidência entre o e-mail e a senha fornecidos pelo usuário com os dados previamente registrados no banco de dados. Se houver uma correspondência exata, o usuário é autenticado com sucesso e tem acesso aos recursos autorizados, caso contrário é emitido um alerta informando que os dados de e-mail ou de senha estão incorretos.

A abordagem adotada para o cadastramento protege o sistema contra múltiplos registros com as mesmas credenciais. O sistema de login com a tupla de e-mail e senha, ajuda a garantir a integridade das informações e restringir o acesso do usuário apenas aos

registros dele.

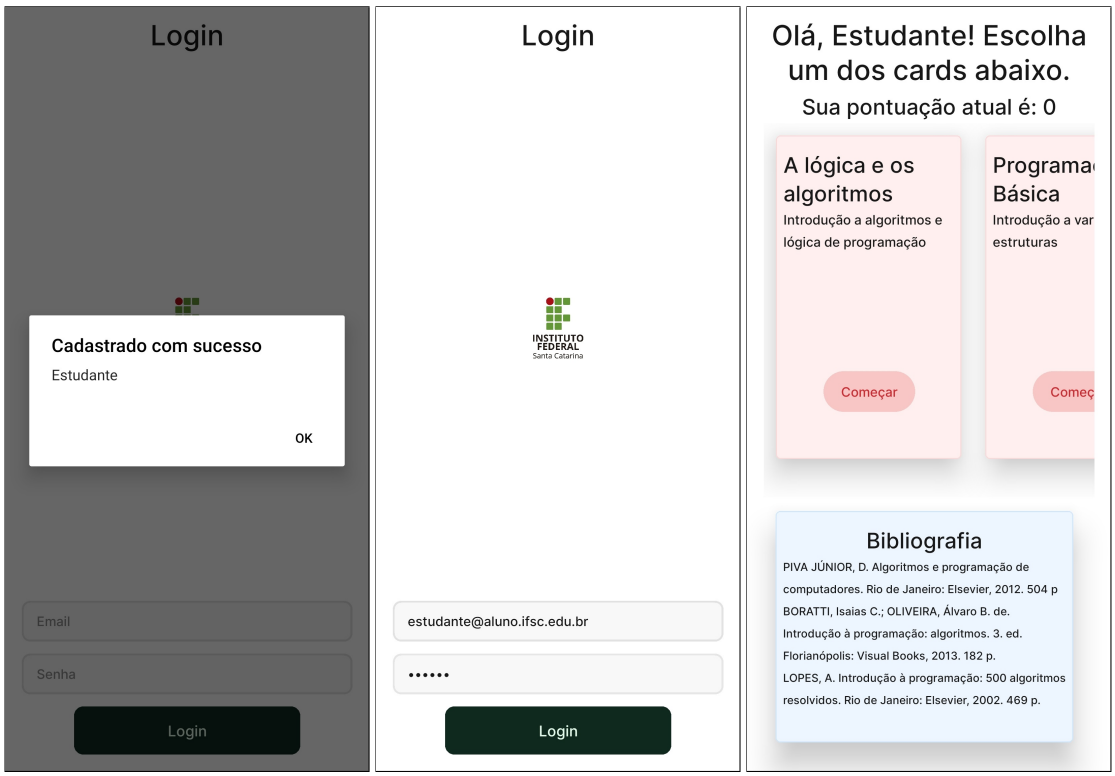

Figura 6. Fluxo de telas de cadastro com sucesso, *login* e trilhas.

3.4.2. Visão Geral

Após o processo de autenticação bem-sucedido, o usuário é redirecionado para a tela de visão geral, onde pode visualizar sua pontuação atual e selecionar entre dois *cards* que representam diferentes *quizzes* disponíveis. Cada *card* corresponde a um *quiz* específico, e o usuário pode escolher qual deseja iniciar a qualquer momento. Isso proporciona ao usuário a flexibilidade de escolher os tópicos ou desafios que mais lhe interessam, tornando a experiência mais personalizada e interativa.

A Figura 7 apresenta um exemplo de código que implementa a renderização de um cartão no aplicativo e a própria tela de visão geral. Na primeira vez em que o usuário acessa o aplicativo, os cartões são exibidos com um fundo de cor vermelha. Após o usuário concluir um *quiz*, o cartão muda para a cor verde, indicando que a categoria correspondente foi concluída. Além disso, o cartão exibe uma mensagem de finalização e o número de pontos recebidos pelo usuário. A bibliografia utilizada para a formulação das perguntas do aplicativo também é exibida na parte inferior da tela.

```
<Card key={index}
  theme="green"
  elevate
  animation="bouncy"
  size="$2"
  width={200}
  height={350}
  margin="$3"
  scale={0.9}
  pressStyle={{ scale: 0.925 }}
  bordered>
  <Card.Header padded marginTop="$2">
    <H3>{card.categoryTitle}</H3>
    <Paragraph fontSize={15}  >{card.categoryInfo}</Paragraph>
  </Card.Header>
  <Card.Footer
    padded
    theme="alt2">
    <YStack>
      <Button
        borderRadius="$10"
        marginLeft="$6"
        marginTop="$4"
        onPress={() => escolherCategoria(card)}
      >
        Começar
      </Button>
      <Paragraph color="blue"
        marginTop={10}
        marginLeft="$6"
      >Trilha completa</Paragraph>
    </YStack>
  </Card.Footer>
  <Card.Background />
</Card>
```

Figura 7. Exemplo de código do *card* e tela de trilhas.

Tendo em vista os altos índices de reprovação nas fases iniciais do curso de ciência da computação levantados no campus Lages, reforçando aqueles mostrados nas pesquisas apresentadas na seção anterior, os temas escolhidos para os *quizzes* foram referentes a algoritmos básicos e introdução a programação.

3.4.3. Questões

O início de cada trilha é marcado por uma introdução ao tema escolhido, preparando o usuário para o *quiz*. Então cada questão é exibida uma a uma, com o contexto, a pergunta e três alternativas, para então o usuário testar seu conhecimento. As perguntas do *quiz* são armazenadas em um arquivo de texto estruturado para transmissão de dados entre aplicações como cliente-servidor chamado *JavaScript Object Notation*[13] (JSON), demonstrado na Figura 8.

[13]https://www.json.org/json-pt.html

```json
[
    {
        "categoryTitle": "A lógica e os algoritmos",
        "categoryInfo": "Introdução a algoritmos e lógica de programação",
        "categoryDetail": "Lógica e algoritmos são os pilares da computação, onde a lógica representa a capacidade de pensar de forma estruturada e organizada, enquanto os algoritmos se traduzem em sequências de passos meticulosamente definidos, destinados a resolver problemas e executar tarefas. Um algoritmo é, essencialmente, um conjunto ordenado de instruções finitas e bem definidas que descrevem como realizar uma tarefa ou resolver um problema.\n\nO estudo da teoria de algoritmos é essencial para a compreensão de como criar, analisar e aprimorar essas sequências de passos, desempenhando um papel central na ciência da computação e na arte de desenvolver software.\n\nO alicerce da computação moderna repousa sobre a lógica sólida e a eficácia dos algoritmos, que possibilitam a automação de processos, tomada de decisões e o funcionamento suave de inúmeras aplicações tecnológicas.",
        "categoryId": 1,
        "questions":[
            {
              "context": "Continuamente realizamos escolhas, efetuamos cálculos e solucionamos desafios, ainda que muitas vezes de maneira automática, guiados por um conjunto específico de etapas ou uma forma particular de pensamento. Isso está intrinsecamente ligado ao conceito de lógica, pois seguimos padrões lógicos. É comum ouvirmos a importância do 'raciocínio lógico' ao criar algoritmos. A lógica é uma parte integral de nosso cotidiano. Para programarmos um computador com sucesso, é essencial possuir conhecimento e compreensão dos algoritmos, bem como a capacidade de aplicá-los para definir a sequência de ações requeridas na resolução de problemas específicos. Em termos mais simples, isso significa descobrir a solução ideal ou mais eficaz para posterior implementação em uma linguagem de programação.",
              "question": "Para a computação, um algoritmo pode ser definido como:",
              "options": [
                "Um tipo de software utilizado para criar gráficos.",
                "Conjunto das regras e procedimentos lógicos perfeitamente definidos que levam à solução de um problema em um número finito de etapas.",
                "Uma unidade de armazenamento em um computador."
              ],
              "correctAnswer": "Conjunto das regras e procedimentos lógicos perfeitamente definidos que levam à solução de um problema em um número finito de etapas.",
              "explanation": "A segunda opção, 'Conjunto das regras e procedimentos lógicos perfeitamente definidos que levam à solução de um problema em um número finito de etapas' é a resposta correta porque define de forma sucinta o que é um algoritmo na computação. Em resumo, um algoritmo é uma sequência de instruções lógicas claramente definidas que, quando seguidas, levam à solução de um problema em um número finito de passos, sendo essencial na programação e na resolução de desafios computacionais."
            },
```


Figura 8. JSON com questões.

A gestão das tentativas de acerto do usuário é exibida na Figura 9. Ao iniciar a trilha escolhida, o usuário passa por uma introdução ao tema e em seguida começam as perguntas. Nesta etapa, ao escolher a alternativa correta, o usuário recebe um ponto e é direcionado para a próxima pergunta do *quiz*, mas caso escolha uma alternativa incorreta para aquela questão, é apresentada uma nova tela com o alerta de resposta incorreta, juntamente com a explicação do motivo do erro, então é direcionado para a próxima questão sem pontuar. Esse fluxo é exibido na Figura 11, ele ocorre até que as perguntas acabem.

```jsx
import { useContext, useState, useEffect } from "react";
import { Alert, Pressable, TouchableOpacity } from "react-native";
import { ArrowLeft } from "@tamagui/lucide-icons";
import { useRouter } from "expo-router";
import {
  Button,
  H3,
  Paragraph,
  Progress,
  ScrollView,
  Text,
  Theme,
  XStack,
} from "tamagui";

import { MyStack } from "../../components/MyStack";
import { UserContext } from "../../contexts/UserContext";

export default function Questao() {
  const { user, setPoints, setCompletedCategories} = useContext(UserContext);
  const questions = user.currentCategory.questions
  const router = useRouter();
  const [currentQuestion, setCurrentQuestion] = useState(0);
  const [score, setScore] = useState(0);
  const progressPercentage = (currentQuestion / questions.length) * 100;


  const handleAnswer = async (answer) => {
    const isCorrect = answer === questions[currentQuestion].correctAnswer;
    if (isCorrect) {
      setScore(score + 1);
    }
    else{
        Alert.alert("Resposta incorreta", `${questions[currentQuestion].explanation}`);
    }

    if (currentQuestion + 1 === questions.length) {

      const updatedScore = score;
      const updatedPoints = user.points + updatedScore;

      setPoints(updatedPoints);


    }
    setCurrentQuestion(currentQuestion + 1);
  };

```

Figura 9. Código de função de gerenciamento.

```jsx
  <Paragraph fontSize="$5">Pontuação Atual {score}</Paragraph>
  <Paragraph>{questions[currentQuestion].context}</Paragraph>
  <H3 fontSize="$6" marginTop="$4" textAlign="left">{questions[currentQuestion].question}</H3>

  <Theme name="dark_green_alt1">
    <Pressable>
      {questions[currentQuestion].options.map((option, index) => (
        <Button
          key={index}
          onPress={() => handleAnswer(option)}

          height="$11"
          borderWidth={2}
          borderColor="transparent"
          margin="$1"
        >
          <Text>{option}</Text>
        </Button>
      ))}
    </Pressable>
  </Theme>
```

Figura 10. Código de exibição de alternativas.

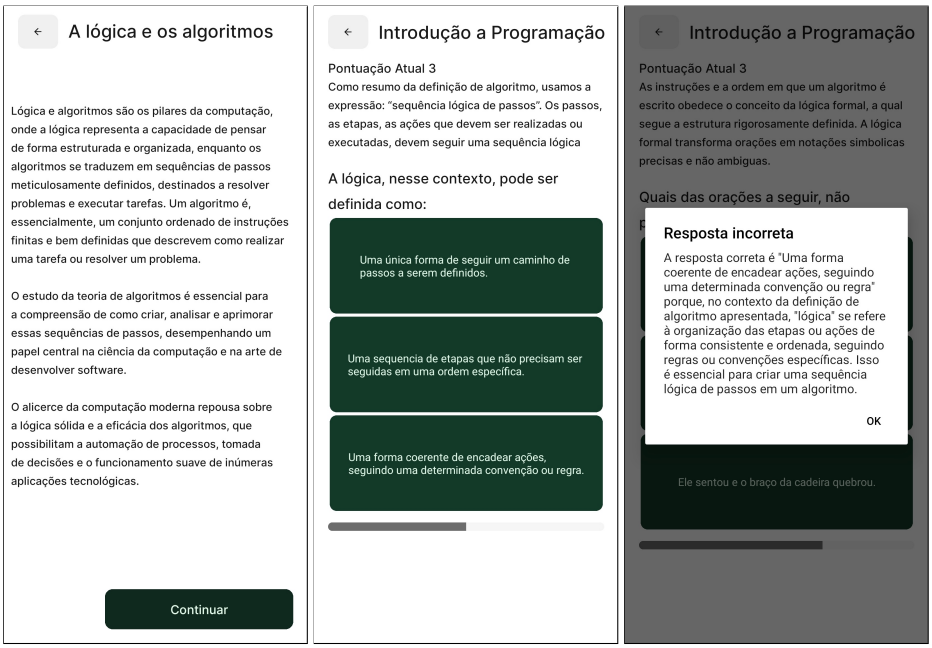

Figura 11. Fluxo de telas ao responder uma pergunta de forma incorreta.

3.5. Desenvolvimento da API

A API foi desenvolvida utilizando o *web framework* FastAPI, na arquitetura cliente-servidor. Tem a responsabilidade de receber requisições do *frontend*, fazer as validações necessárias e enviar a resposta adequada sobre elas. Também faz a comunicação com o banco de dados através de um conjunto de funcionalidades permitindo criar, ler, atualizar e deletar registros referentes aos usuários e as questões do aplicativo.

3.5.1. Modelos e Esquemas

Os modelos são classes que representam os dados do aplicativo. Eles são utilizados para interagir com o banco de dados, definindo as tabelas, os campos, e os relacionamentos entre as tabelas. Na Figura 12 é apresentado o modelo de criação da tabela de usuário, onde é definido o nome da tabela, seguido de suas colunas e especificações de tipos e atributos.

```
8   class User(Base):
9       __tablename__ = "user_app"
10
11      id_user = Column(Integer, primary_key=True, index=True)
12      name = Column(String)
13      email = Column(String)
14      password = Column(String)
15      point = Column(Integer)
16      time_created = Column(DateTime(timezone=True), server_default=func.now())
17      time_updated = Column(DateTime(timezone=True), onupdate=func.now())
18
19      user_category = relationship("UserCategory", back_populates="user")
```

Figura 12. Código de modelo de usuário.

Os esquemas definem como os dados devem ser formatados, quais são seus tipos, quais são obrigatórios ou não, validando e mantendo a consistência na troca de informações. A Figura 13 apresenta o esquema para ações com o usuário, passando pela configuração de *Object-Relational Mapping* (ORM) que permite mapear objetos Python para tabelas de banco de dados, facilitando a integração entre essas ferramentas.

```python
class UserBase(BaseModel):
    name: str
    email: str
    point: int

class UserCreate(UserBase):
    password: str

class User(UserBase):
    id_user: int

    class Config:
        orm_mode = True
```

Figura 13. Código de esquema de usuário.

3.5.2. Operações

Dentro da plataforma temos a opção de realizar algumas operações com as entidades. As operações são divididas em *Create, Read, Update, Delete* (CRUD), são fundamentais para a interação com o banco de dados.

• *Create*

A operação *Create* tem a função de criar um novo usuário. No caso deste projeto, exige algumas informações do usuário, como nome, e-mail e senha, descritos na Figura 14. Esses dados são registrados no banco de dados e utilizados para o acesso do usuário ao aplicativo.

```python
def create_user(db: Session, user: schemas.UserCreate):
    password = cript(user.password)
    db_user = models.User(
        name=user.name,
        email=user.email,
        point=0,
        password=password
        )
    print(db_user)
    db.add(db_user)
    db.commit()
    db.refresh(db_user)
    return db_user

def cript(password: str):
    return hashlib.sha256(password.encode()).hexdigest()
```

Figura 14. Código de função criar novo usuário.

• *Read*

A operação *Read* é responsável por fazer a leitura de dados do banco de dados, conforme a requisição recebida. A Figura 15 apresenta duas formas de leitura, a primeira através do identificador único do usuário e a segunda através do e-mail do usuário, ambas retornando o objeto do banco de dados.

```
24  def get_user(db: Session, user_id: int):
25      return db.query(models.User).filter(models.User.id_user == user_id).first()
26
27  def get_user_by_email(db: Session, email: str):
28      return db.query(models.User).filter(models.User.email == email).first()
```

Figura 15. Código de função ler usuário.

• *Update*

A operação *Update* faz a atualização de dados do usuário no banco de dados. A Figura 16 apresenta a atualização de pontos do usuário.

```
35  def update_user_point(db: Session, user_id: int, user: schemas.UserCreate):
36      db_user = db.query(models.User).filter(models.User.id_user == user_id).first()
37      db_user.point = user.point
38      db.commit()
39      db.refresh(db_user)
40      return db_user
```

Figura 16. Código de função atualizar pontuação usuário.

• *Delete*

A operação *Delete* permite que o usuário exclua seus dados do aplicativo e banco de dados. A Figura 17 apresenta essa operação.

```
43  def delete_user(db: Session, user_id: int):
44      db_user = db.query(models.User).filter(models.User.id_user == user_id).first()
45      db.delete(db_user)
46      db.commit()
47      return db_user
```

Figura 17. Código de função deletar usuário.

3.5.3. Rotas

As rotas ou *endpoints* servem para definir comportamentos e ações durante as interações com a interface de usuário. Baseado no protocolo *Hypertext Transfer Protocol* (HTTP), podemos utilizar os métodos *POST, GET, UPDATE e DELETE*.

• Rota criar novo usuário

Essa rota definida com o método *POST* recebe as informações do formulário pre-enchido pelo usuário no *frontend*, baseado no formato JSON Schema, faz a validação de e-mail e retorna uma resposta para o *frontend*, confome a Figura 18.

```python
@app.post("/users/", response_model=schemas.User)
def create_user(user: schemas.UserCreate, db: Session = Depends(get_db)):
    db_user = crud.get_user_by_email(db, email=user.email)
    if db_user:
        raise HTTPException(status_code=400, detail="email already registered")
    return crud.create_user(db=db, user=user)
```

Figura 18. Código de função criar novo usuário.

A requisição deve chegar formatada conforme mostra a Figura 19. A partir dela é feita a validação do e-mail, caso o e-mail esteja cadastrado, ele se torna inválido para novo cadastro, retornando como resposta a Figura 20. Caso seja um e-mail novo, o cadastro é válido e como retorna como resposta a Figura 21, para que o *frontend* armazene as informações de nome, e-mail, pontos e identificador único do usuário na sessão.

```json
{
    "name": "Estudante",
    "email": "estudante@email.com",
    "point": 0,
    "password": "12345"
}
```

Figura 19. JSON com requisição criar usuário.

```json
{
    "detail": "email already registered"
}
```

Figura 20. JSON com resposta erro na criação.

```json
{
    "name": "Estudante",
    "email": "estudante@email.com",
    "point": 0,
    "id_user": 5
}
```

Figura 21. JSON com resposta criação com sucesso.

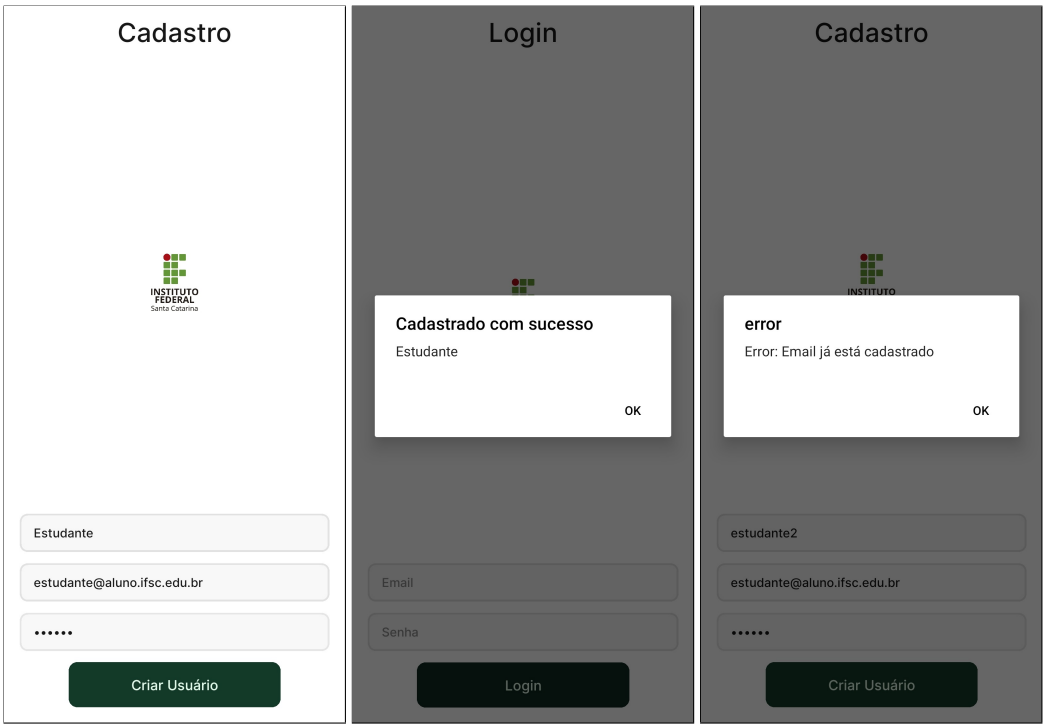

Figura 22. Fluxo das telas de cadastro com retorno de sucesso e de erro.

• Rota ler

As rotas de leitura utilizam o método *GET* para retornar os dados contidas no banco de dados. A Figura 23 apresenta a leitura de dados de um usuário através do seu identificador único, é feita a validação se existe esse identificador informado, retornando como resposta a Figura 24 caso não exista.

```
66  @app.get("/users/{user_id}", response_model=schemas.User)
67  def read_user(user_id: int, db: Session = Depends(get_db)):
68      db_user = crud.get_user(db, user_id=user_id)
69      if db_user is None:
70          raise HTTPException(status_code=404, detail="User not found")
71      return db_user
```

Figura 23. Código de função ler usuário.

```
1  {
2      "detail": "User not found"
3  }
```

Figura 24. JSON com leitura de usuário inválido.

• Rota atualizar

Rotas de atualização de informações no banco de dados utilizam o método *PUT*. As Figuras 25 e 26 apresentam a atualização dos pontos do usuário ao finalizar uma trilha, fazendo a validação através do seu identificador único.

```python
@app.put("/users/{user_id}/point", response_model=schemas.User)
def update_user_point(user_id: int, user: schemas.UserCreate, db: Session = Depends(get_db)):
    db_user = crud.get_user(db, user_id=user_id)
    if db_user is None:
        raise HTTPException(status_code=404, detail="User not found")
    return crud.update_user_point(db=db, user_id=user_id, user=user)
```

Figura 25. Código de função atualizar usuário.

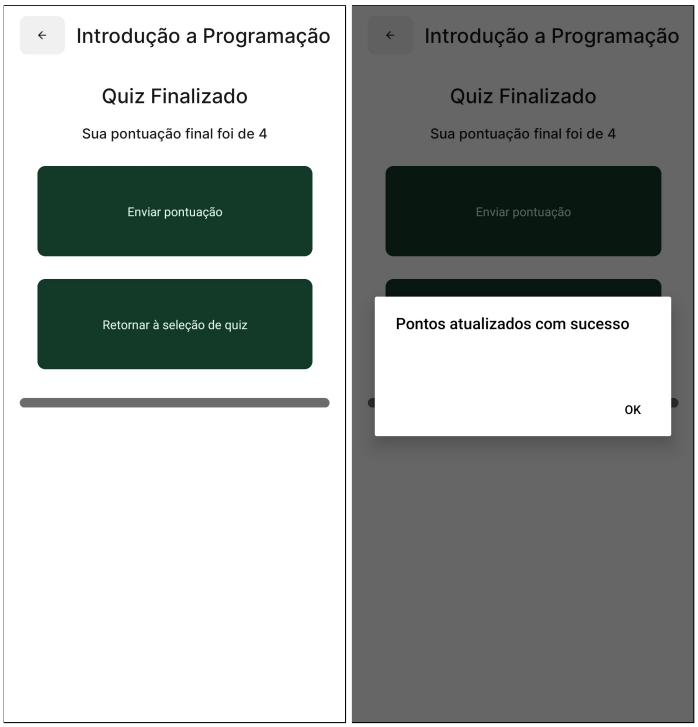

Figura 26. Tela finalização do *quiz* e atualização de pontos.

• Rota deletar

A rota apresentada na Figura 27 utiliza o método *DELETE*, oferecendo a opção para o usuário deletar seus dados através do seu identificador único.

```python
@app.delete("/users/{user_id}", response_model=schemas.User)
def delete_user(user_id: int, db: Session = Depends(get_db)):
    db_user = crud.get_user(db, user_id=user_id)
    if db_user is None:
        raise HTTPException(status_code=404, detail="User not found")
    return crud.delete_user(db=db, user_id=user_id)
```

Figura 27. Código de função deletar usuário.

3.5.4. Implantação

Para a implantação da API no ambiente de produção, criou-se um arquivo *dockerfile* o qual contem as instruções necessárias para criar o ambiente virtual e executar os comandos de

instalação e inicialização do código. Todo esse processo é feito na plataforma Render, que deixa um link disponível para o consumo da API.

```
1   FROM python:3.11-slim
2   ENV POETRY_VIRTUALENVS_CREATE=true
3
4   RUN pip install poetry
5
6   COPY . .
7
8   RUN poetry config installer.max-workers 10
9   RUN poetry install --no-interaction --no-ansi
10
11  EXPOSE 8000
12  CMD [ "poetry", "run", "uvicorn", "--host", "0.0.0.0", "api.main:app" ]
```

Figura 28. Dockerfile.

4. Avaliação

Esta seção tem como objetivo apresentar e discutir sobre a avaliação do aplicativo móvel realizada com o público a cerca de aspectos visuais, técnicos e educativos. A avaliação foi aplicada através de uma apresentação sobre o projeto e posteriormente o aplicativo foi disponibilizado para que o usuário fizesse o cadastro, login e acessasse as trilhas de aprendizagem. Cada trilha conta com uma sequência de perguntas de múltipla escolha, adicionando pontuação a cada resposta certa ou mostrando uma explicação a cada resposta errada. Ao final de cada trilha, o usuário pode guardar seus pontos no banco de dados ou reiniciar a trilha.

Após o uso do aplicativo, foi disponibilizado um questionário para coletar informações referentes a experiência do usuário, foi solicitado o consentimento dos usuários para a participação da pesquisa e as respostas de 13 usuários foram coletadas de forma anonima.

A métrica de avaliação adota a escala baseada na de Likert, a qual foi concebida pelo educador e psicólogo norte-americano Rensis Likert (Bermudes et al., 2016). A pesquisa disponibiliza um conjunto de alternativas de resposta relacionadas a um tópico, sendo empregada como um elemento em perguntas de escolha múltipla, juntamente com um espaço aberto para receber sugestões gerais.

Sobre o perfil de conhecimento prévio sobre os assuntos abordados no aplicativo, obtivemos as respostas contidas da Figura 29. Observa-se que 92,3% dos usuários já haviam passado pela fase de algoritmos básicos de programação.

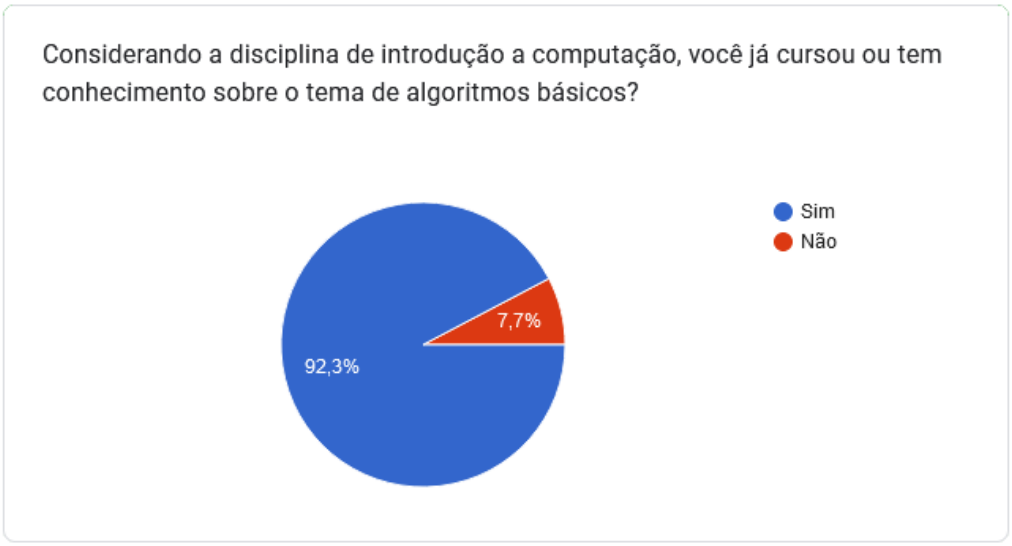

Figura 29. Conhecimentos prévios de algoritmos básicos.

Sobre a frequência que o entrevistados utilizam aplicativos educativos, onde **0** representa *não utiliza* e **5** representa *utiliza diariamente*, obtivemos as respostas da Figura 30. Observa-se que todos os entrevistados utilizam aplicativos educativos com alguma frequência.

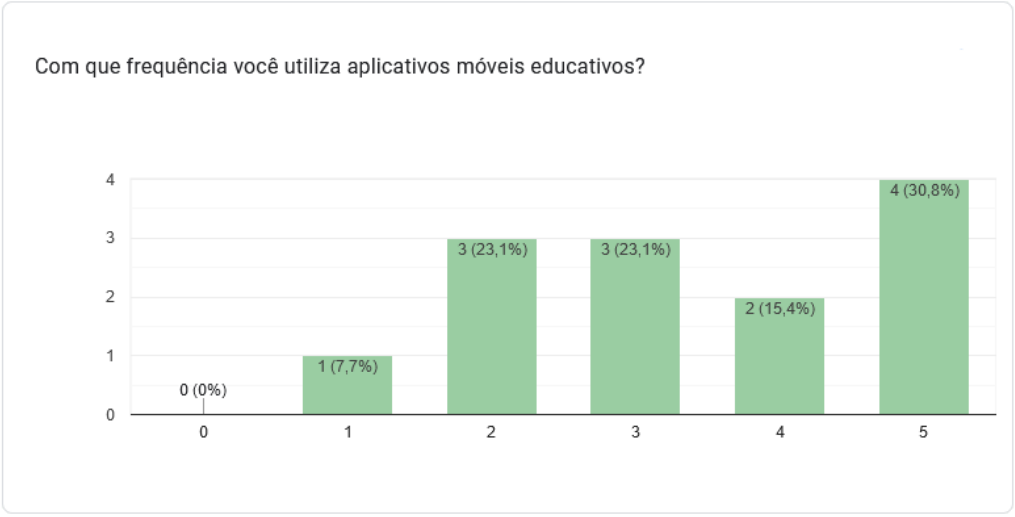

Figura 30. Frequência de uso de aplicativos educativos.

Sobre a navegação e as instruções contidas no aplicativo, onde **0** representa *complexo* e **5** representa *fácil*, obtivemos as seguintes respostas nas Figuras 31 e Figura 32. Observa-se que na maioria dos casos a navegação e instruções estão satisfatórias.

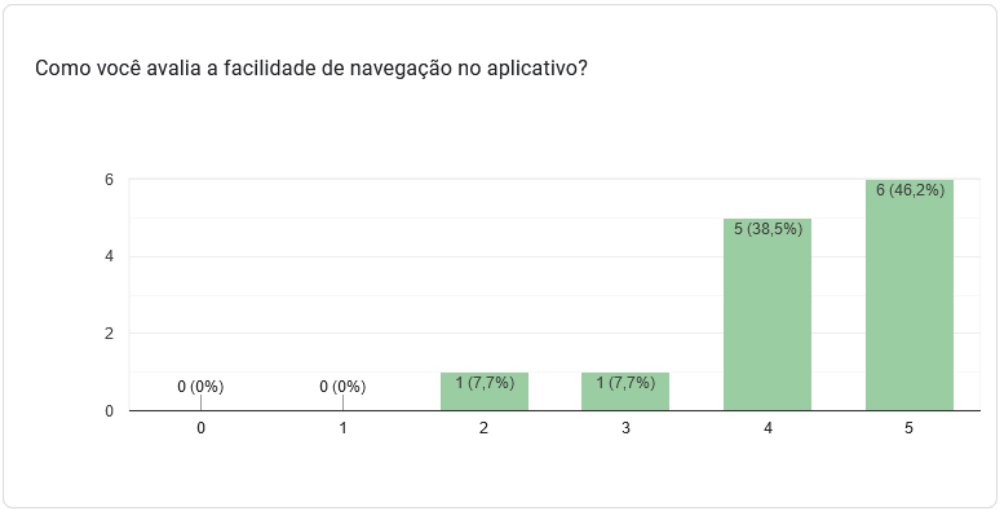

Figura 31. Navegação do aplicativo.

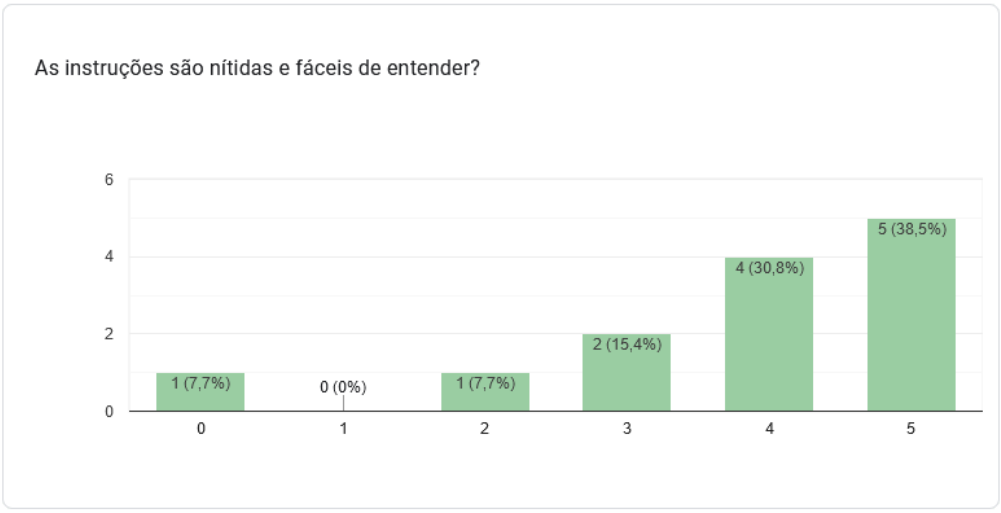

Figura 32. Instruções de uso do aplicativo.

Sobre as explicações de contexto apresentado antes das questões, onde **0** representa *não ajuda* e **5** representa *ajuda muito*, obteve-se as respostas apresentadas na Figura 33. Observa-se que na maioria dos casos a explicação e instruções estão ajudando a responder a pergunta.

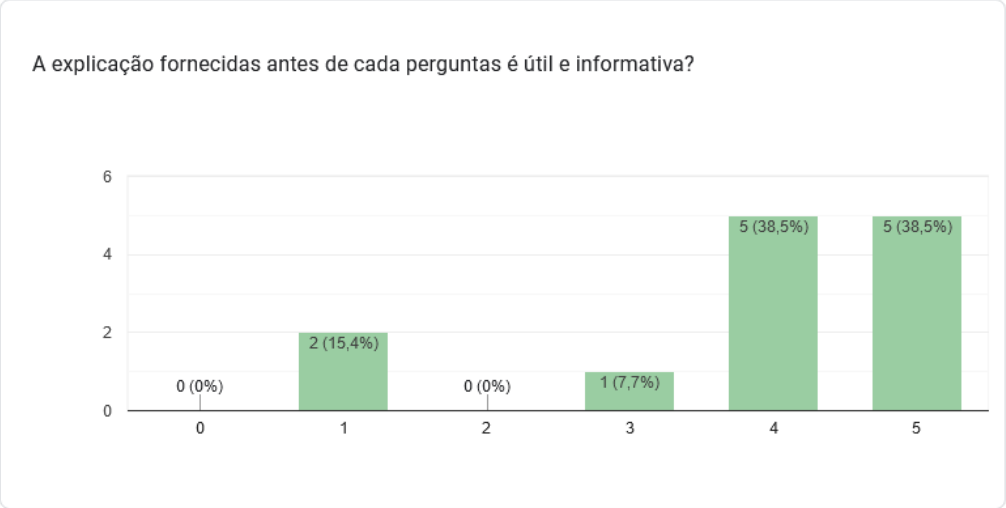

Figura 33. Explicações antes das perguntas.

Sobre o quanto o aplicativo facilita na compreensão de conteúdos de computação, onde **0** representa *não facilita* e **5** representa *facilita muito*, obteve-se as respostas na Figura 34. Observa-se que na maioria dos casos o uso de aplicativo facilita na compreensão de conteúdos de computação.

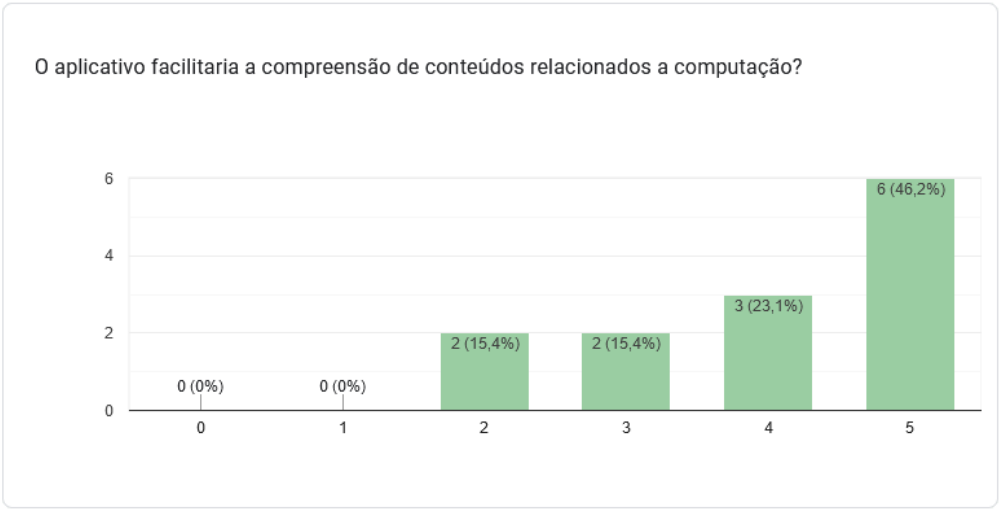

Figura 34. Facilidade de compreensão de conteúdos de computação.

Sobre a recomendação do aplicativo para praticar temas relacionados a programação, na Figura 35. Observa-se que 84,6% dos usuários recomenda o uso do aplicativo para praticar programação.

Figura 35. Recomendação do aplicativo.

No espaço aberto de sugestões,pode ser encontrado as respostas recebidas na figura 36 encontrada dentro do apêndice. Obteve-se principalmente apontamentos em relação a grande quantidade de texto presentes tanto no contexto quanto nas questões, deixando-as extensivas e cansativas, além de alguns apontamentos de ambiguidade nas questões de programação básica. Algumas sugestões quanto a usabilidade também surgiram, apontando a falta de acessibilidade para pessoas com visão limitada e a barra de progresso aparenta uma barra de rolagem, causando confusões na navegação.

5. Conclusões

O uso de metodologias ativas no processo de aprendizagem pode despertar no estudante a curiosidade e motivação para estudar.

Neste trabalho, foi aplicada a metodologia de gamificação, engajando os estudantes a responder perguntas relacionadas a algoritmos computacionais e programação básica, desta maneira estimulando o raciocínio lógico e a busca por conhecimento.

Algumas dificuldades foram encontradas durante a implementação do aplicativo móvel. Houve a necessidade de estudos diversos para o entendimento do funcionamento do framework utilizado para o desenvolvimento da API *backend* (FastAPI), além disso, a fase de implementação na plataforma de hospedagem (Render) exigiu o uso de uma tecnologia não apresentada durante o curso, a plataforma de criação de ambiente virtual (Docker). Também houve o desafio na integração da persistência de dados com a interface de usuário, na adaptação do formato de dados transmitidos e recebidos entre as partes envolvidas. A etapa de elaboração de questões exigiu bastante pesquisa em livros didáticos, para encontrar materiais compatíveis com os temas iniciais propostos. Os desafios e as dificuldades foram superadas, dando seguimento no desenvolvimentos do aplicativo móvel.

Como trabalhos futuros há a possibilidade de aumentar os temas abordados para praticar no aplicativo, além de algoritmos iniciais e programação básica; a possibilidade é a melhoria no quesito de segurança do aplicativo, adicionando módulos de autenticação e

criptografia nos *endpoints*; criação de um sistema de ranqueamento, mostrando os usuário com maior pontuação durante um período de tempo; e implementação das sugestões apontadas pelos usuários na avaliação.

Referências

Berbel, N. (2011). As metodologias ativas e a promoção da autonomia de estudantes. *Semina: Ciências Sociais e Humanas*, 32.

Bermudes, W. L., Santana, B. T., Braga, J. H. O., e Souza, P. H. (2016). Tipos de escalas utilizadas em pesquisas e suas aplicações. *Revista Vértices*, 18(2):7–20.

da Costa Chagas, L. B. e De Oliveira, M. G. (2011). Metodologia anea para avaliação online de lógica de programação. In *Anais do XVII Workshop de Informática na Escola*, pages 1394–1397. SBC.

da Silva, C. F. G. e de Oliveira, S. J. (2022). Desenvolvimento de uma plataforma para geração de simulados.

de Abreu Cybis, W., Betiol, A. H., e Faust, R. (2015). *Ergonomia e Usabilidade 3ª edição: Conhecimentos, Métodos e Aplicações*. Novatec Editora.

Farias, S. C. (2013). Os benefícios das tecnologias de informação e comunicação (tic) no processo de educação a distância (ead). *RDBCI: Revista Digital de Biblioteconomia e Ciência da Informação*, 11(3):15–29.

Fernando Andrade Matos, L., Rodrigues Costa, A., de Oliveira Siqueira, G., Alves dos Santos Menezes, J., Ferreira Neves, D., e Cerqueira Santos, L. (2019). Labmorfoquiz: um aplicativo gamificado como recurso para aprendizagem em cursos superiores de saúde. *Revista Novas Tecnologias na Educação*, 17(3).

Gomes, A., Henriques, J., e Ántónio. José Mendes (2008). Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Educação, Formação e Tecnologias*, 01(1).

Gomes, D. C. F. e de Castro, V. B. (2013). Ensino e aprendizagem de vocabuláio da língua inglesa de forma dinâmica: desenvolvimento de aplicativo para dispositivos móveis. In *Anais do I Encontro Nacional de Computação dos Institutos Federais*. SBC.

Guzzo, D. A. (2020). A utilização de jogos educacionais digitais como proposta de metodologia ativa de ensino para uma aprendizagem significativa na educação básica.

IFSC (2022). *RESOLUÇÃO CEPE/IFSC No 028 DE 12 DE MAIO DE 2022.*

Lopes, L. A., Pinheiro, E., da Silva, T., e Zaina, L. (2019). Requisitos de usabilidade para softwares aplicados ao e-learning: uma proposta para elaboração de user stories. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 1121.

Machado, L., Berkenbrock, C., Bianeck, G., e Siple, I. (2018). Uma ferramenta colaborativa para apoiar a aprendizagem de programação de computadores. *Revista Brasileira de Computação Aplicada*, 10(1).

Mocbel, M. Â. R. e Farias, F. d. S. (2018). Aplicativo exatas: Ferramenta de apoio ao aprendizado móvel.

Pontes, T. B. e Arthaud, D. D. B. (2018). Metodologias ágeis para o desenvolvimento de softwares. *Ciência e Sustentabilidade*, 4(2):173–213.

Pressman, R. e Maxim, B. (2021). *Engenharia de software - 9.ed.* McGraw Hill Brasil.

Rafaela, d. S. M. e Breno, G. B. N. (2014). Aplicativos educacionais livres para mobile learning. *Tecnologias na Educação*.

Soares, B. C. e Resende, R. S. (2017). Requisitos para utilização de prototipagem evolutiva nos processos de desenvolvimento de software para web. *Universidade Federal de Minas Gerais (UFMG)*.

StackOverflow (2022). Survey. `https://survey.stackoverflow.co/2022`. Acesso em: 03 de maio de 2023.

Witt, D. T. e Kemczinski, A. (2020). Metodologias de aprendizagem ativa aplicadas à computação: uma revisão da literatura. *Informática na educação: teoria & prática*, 23(1 Jan/Abr).

Zhan, Z., He, L., Tong, Y., Liang, X., Guo, S., e Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence*, page 100096.

6. Apêndice de dados da avaliação

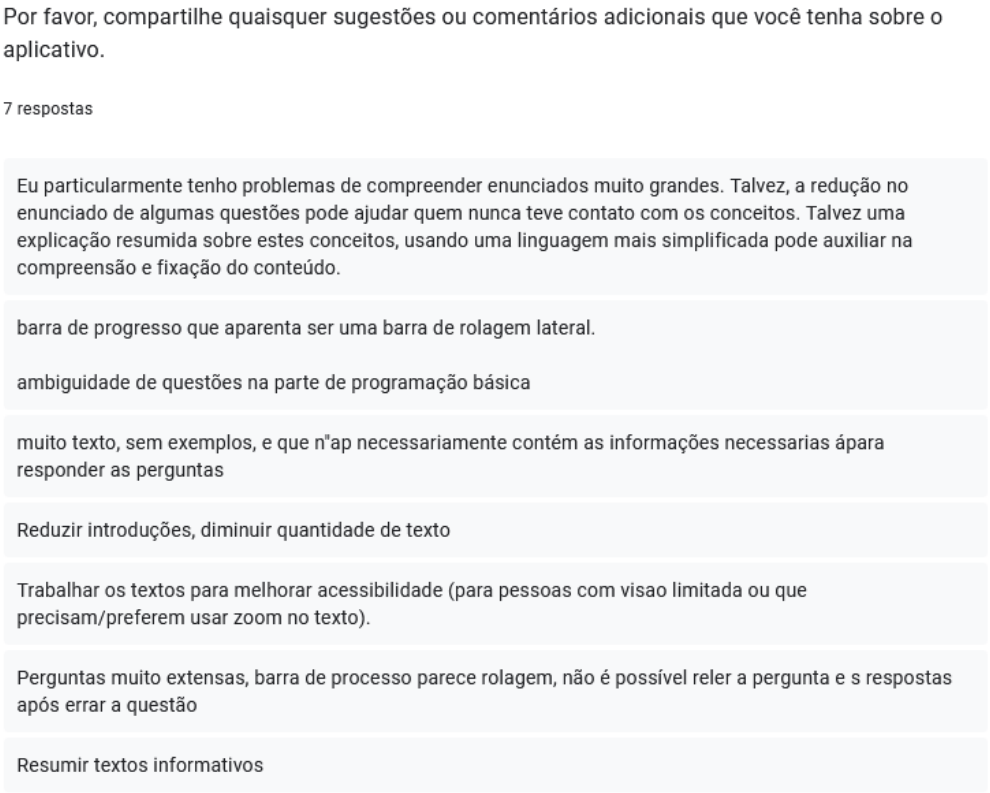

Figura 36. Dados.