Rabbit's Challenge: Aprendizagem de Conceitos de Lógica de Programação através de Aplicação Gamificada[1]

Caroline Caprini da Silveira[2], Rafael Vieira Coelho[3]

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - *Campus* Farroupilha

Farroupilha - Rio Grande do Sul, Brasil

caroline.caprini@hotmail.com; rafael.coelho@farroupilha.ifrs.edu.br

Resumo

A alta taxa de evasão em cursos da área da informática vai na contramão da demanda por desenvolvedores qualificados. É notável a necessidade de tornar o ensino de programação mais acessível e estimulante. Neste trabalho, utilizam-se conceitos de gamificação, como customização de personagens e pontos de experiência, para criar uma aplicação de ensino de lógica e conceitos de programação. A proposta combina a diversão dos jogos casuais 2D com os desafios do aprendizado técnico, em busca de facilitar a compreensão do conteúdo apresentado de forma tradicional, em sala de aula. Para o desenvolvimento, foram utilizadas as tecnologias Flutter, Flame e FlutterFlow para a criação do jogo e das telas de acesso ao sistema; Blockly, para o ensino de lógica e programação em blocos; e Firestore, para o armazenamento de dados. Como funcionalidades, a aplicação inclui explicações sobre o uso da ferramenta Blockly e sua integração com dois níveis de dificuldade, nos quais os alunos são desafiados a criar códigos que movem o personagem até o objetivo final. Além dos desafios, o sistema oferece feedbacks de pontuação, perfil de usuário e opções de customização básica do personagem jogável. Isto permite que alunos não apenas aprendam lógica de

[1] Artigo científico referente ao Trabalho de Conclusão de Curso do curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal do Rio Grande do Sul - Campus Farroupilha.

[2] Aluna do curso de Tecnologia em Análise e Desenvolvimento de Sistemas, matriculado no Trabalho de Conclusão de Curso.

[3] Professor do Instituto Federal do Rio Grande do Sul e Orientador do Trabalho de Conclusão de Curso.

programação, mas também se envolvam em uma experiência divertida e motivadora, tornando o aprendizado eficaz.

Palavras-chave: *Metodologia de ensino; Gamificação; Programação; Lógica; Blockly;*

Abstract

The high dropout rate in computer science courses contradicts the demand for qualified developers. There is a notable need to make programming teaching more accessible and stimulating. This work uses gamification concepts, such as character customization and experience points, to create an application for teaching logic and programming concepts. The proposal combines the fun of 2D casual games with the challenges of technical learning, in an attempt to facilitate understanding of the content presented in the traditional classroom way. Flutter, Flame, and FlutterFlow technologies were used to create the game and the system access screens; Blockly to teach logic and block programming; and Cloud Firestore to store data. As a feature, the application includes explanations on using the Blockly tool and its integration with two levels of difficulty, in which students are challenged to create codes that move the character to the final goal. In addition to the challenges, the system offers score feedback, a user profile, and basic customization options for the playable character. This allows students to learn programming logic and engage in a fun and motivating experience, making learning effective.

Keywords: *Teaching methodology; Gamification; Programming; Logic; Blockly;*

1. Introdução

Nos últimos anos, tem se tornado presente na mídia a alta procura por profissionais qualificados em áreas técnicas como desenvolvimento de software, seja em sistemas *web* e *desktop* como aplicativos móveis, em contraste com o baixo número de graduados nestas áreas (Jornal Hoje, 2023).

Como apresentado em revisão por Alonso (2022), há um alto nível de evasão presente no ensino técnico brasileiro, e alguns dos motivos para este número é a falta de identificação com o curso, falta de boa didática, materiais de apoio às disciplinas e dificuldade de compreensão dos assuntos abordados em aula, assim como fatores socioeconômicos, como a necessidade de conciliar trabalho com os estudos.

Como abordado por Qian e Lehman (2017), também percebe-se que alguns dos obstáculos para quem deseja inserir-se no mundo da programação são a compreensão do raciocínio lógico-aritmético, conhecimento sintático e conceitual, resolução de problemas e conforto em relação às metodologias utilizadas para ensino. Todos estes obstáculos contribuem para uma queda na motivação e aumento da insegurança do estudante, o que gera a alta evasão mencionada anteriormente.

Outro fator limitante para quem busca aprender é a baixa acessibilidade de grande parte das ferramentas de ensino disponíveis no mercado, cujas certificações apresentam apenas versões pagas, não possuem tradução para a língua portuguesa, ou não estão disponíveis na íntegra em dispositivos de fácil acesso pelo aluno, o que prejudica o aprendizado daqueles que não possuem as ferramentas adequadas fora de ambientes educacionais, não possuem proficiência linguística para compreenderem os conceitos demonstrados em outro idioma ou não dispõe do recurso financeiro necessário para arcar com os valores solicitados pelas plataformas.

Unindo conceitos como a gamificação com o ensino de programação e tecnologias atuais do mercado, a ferramenta apresentada neste trabalho foi criada, buscando como objetivo aproximar os estudantes do conhecimento de lógica e de programação e fortalecer a solução de problemas de forma gratuita e acessível, compartilhando conhecimentos de forma eficaz e divertida.

2. Fundamentação Teórica

Nesta seção, é abordado como a motivação e a gamificação se relacionam com a aprendizagem do aluno, seguido de uma análise das principais técnicas de gamificação assim como das principais aplicações de ensino de programação gamificadas já disponíveis no mercado.

2.1 A Motivação e a sua Relação com a Gamificação na Aprendizagem de Programação

Como apresentado por Alves (2015), motivação "[...] é a condição do organismo que influencia a direção do comportamento, a orientação para um objetivo, e por isso, está relacionada a um impulso que leva à ação.". Conforme apresentado anteriormente, um dos maiores desafios relacionados ao ensino de lógica e programação está intrinsecamente ligado à manter a motivação do aluno em aprender novos conceitos. Em seu estudo, Alves apresenta que mesmo que a motivação seja algo específico de cada indivíduo, o denominador comum entre todos é a diversão, ou seja, uma das formas de se manter a motivação é tornar aquilo que está sendo apresentado divertido e instigante para quem está aprendendo. É assim que a gamificação e sua interação com a motivação se tornam relevantes.

Como definido por Karl Kapp (2012), "Gamificação é a utilização de mecânica, estética e pensamento baseados em jogos para engajar pessoas, motivar a ação, promover a aprendizagem e resolver problemas.". A gamificação é uma estratégia de ensino que alia técnicas de solução de problemas e desafios educacionais à ambientes de aprendizado que possuam estrutura semelhante a jogos (ZHAN, Zehui et al, 2022), sejam eles quebra-cabeças, jogos de interpretação de papeis e narrativas, digitais ou de mesa, em busca de estimular a motivação, a aprendizagem e a compreensão do estudante, a fixação do conteúdo apresentado, e o aumento da proficiência acadêmica, de forma geral.

A gamificação na aplicação desenvolvida ao decorrer deste trabalho é utilizada na criação de desafios e cenários que incentivem o aluno a explorar todas as suas possibilidades, utilizando a construção visual de código por meio de blocos. Entende-se como necessária a instrução do professor, como percebido em meta-análise realizada

por Robert T. Hays (2005), acerca de conceitos de programação básica em portugol para que os desafios sejam resolvidos, dando assim liberdade criativa ao estudante.

É importante notar que trata-se de uma estratégia em constante evolução e análise por parte de pesquisadores, e por isso, possui pontos positivos e negativos em sua utilização na educação, sendo estes percebidos nos diferentes resultados apresentados em estudos empíricos.

Os principais pontos positivos constatados foram a melhora na motivação e desempenho acadêmico, aumento na quantidade de questionamentos, o que gerou uma maior participação em aulas. Já os pontos negativos se relacionam principalmente na utilização de estratégias que estimulam a competitividade, o que acabou diminuindo o compartilhamento de conhecimento e a cooperação entre alunos, gerando desinteresse naqueles com maior dificuldade para desenvolver os conceitos trabalhados em cada desafio (ZHAN, Zehui et al, 2022). Em estudo conduzido por Vogel, et. al., percebeu-se uma preferência dos aprendizes pela navegação em sistemas gamificados e simulações interativas, sendo que estes, comparados a métodos de ensino tradicionais, geraram um maior estímulo para os alunos.

No entanto, o consenso entre pesquisadores acerca desta metodologia é positivo, principalmente em áreas técnicas, como a informática. A gamificação pode ser uma forte aliada no ensino de novos programadores, visto que pode ser utilizada para estimular a motivação de quem está aprendendo, aumentando a retenção de conceitos técnicos, engajando o aluno em soluções de problemas, o que torna um aprendizado que pode ser maçante para alguns, divertido e democrático para todos.

2.2 Uma Breve Análise das Principais Técnicas de Gamificação

A utilização de técnicas de gamificação provém uma maior aderência daquele que as utiliza no aprendizado, independentemente do conteúdo que se busca aprender (FIGUEIREDO, GARCÍA-PEÑALVO, 2020). Na área da tecnologia, e com o ensino de programação, técnicas de gamificação tem se mostrado grandes aliados no auxílio da compreensão de lógica e conceitos técnicos (ZHAN, Zehui et al, 2022).

Para que uma ferramenta gamificada tenha êxito, além de elementos visuais de jogos, é necessário que se busque seguir uma arquitetura coesa, em que o jogador seja mantido em um *loop* de engajamento (ALVES, 2015), sendo que este *loop* possui três elementos: motivação, ação e *feedback*. Por exemplo, a motivação de aprender o conteúdo, ou resolver o desafio proposto, a ação de aprender e/ou resolver e o *feedback* da ferramenta acerca de seu desempenho na tarefa executada.

Pensando nesta arquitetura para o desenvolvimento de ferramentas gamificadas, foram analisadas algumas das principais técnicas de gamificação baseadas em jogos utilizados no ensino. Atualmente, estas são de forma individual: *streaks*, pontos de experiência, medalhas e conquistas, customização de personagem, e de forma comunitária: desafios e competição entre jogadores e entre ligas.

Streaks são registros diários de comparecimento ou execução de tarefas dentro da ferramenta, que são acumulados ou zerados, dependendo da assiduidade do usuário. Além de reforçar a criação de hábitos, os streaks podem trazer conquistas e pontos aos jogadores, motivando o aluno a manter a sequência de aprendizado.

Os pontos de experiência são acumulados com o cumprimento de missões, tarefas, exercícios, aulas, sendo que podem ser utilizados para subir de nível e/ou desbloquear novos desafios, incentivando o aluno a agir e resolver desafios.

As medalhas e conquistas são duas formas de premiar o jogador (ou estudante) por seu desempenho, transmitindo o *feedback* esperado ao usuário, e que podem ser adquiridas ao subir de nível, resolver desafios ou até mesmo, finalizar um módulo de conhecimento.

A customização de personagem permite inserir uma visualização gráfica do seu personagem, o que leva o usuário ao sentimento de pertencimento à ferramenta. Com a customização da aparência, também é possível adicionar itens cosméticos que podem ser comprados com moedas do jogo ou pontuações, ambos obtidos ao solucionar desafios e/ou fases, instigando o jogador a realizar as tarefas.

Em nível comunitário, a criação de ligas de nível (como madeira, bronze, prata, ouro e diamante) incentiva a competitividade entre os jogadores, que podem subir de

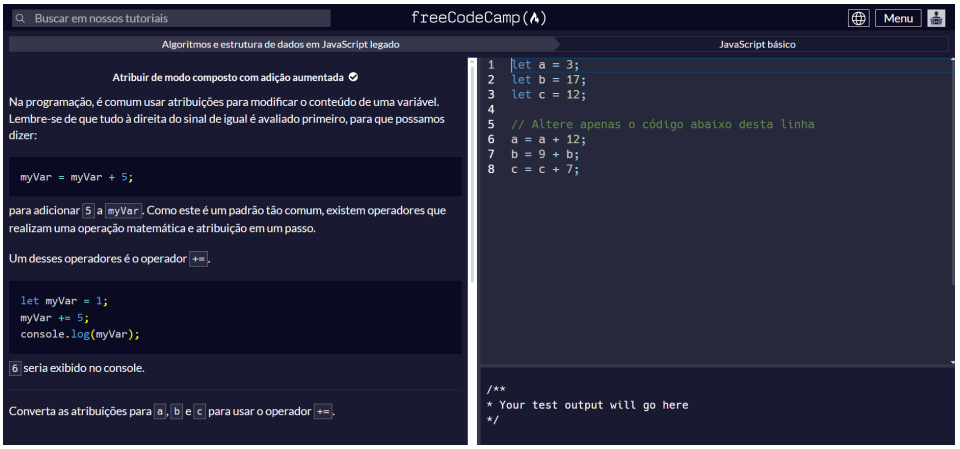
nível dentro das ligas resolvendo exercícios e/ou revisando lições. A competição entre diferentes ligas de mesmo nível pode ser um recurso utilizado, pois permite o trabalho em equipe em prol de solucionar um desafio, o que no mundo real, é uma habilidade necessária para o mercado de trabalho, e que na ferramenta, abrange o *loop* de engajamento de forma completa, garantindo que o aluno se mantenha presente e aprendendo.

2.3 Quais são as Ferramentas de Ensino Disponíveis no Mercado?

Para o desenvolvimento deste trabalho, foram analisadas quatro ferramentas de ensino de programação já disponíveis no mercado, sendo estas a FreeCodeCamp, a CodeAcademy, Mimo e a Sololearn. Para esta análise, foram estabelecidos os seguintes critérios de avaliação: formas de acesso (navegador, *desktop*, *mobile*), presença de metodologias de gamificação, suporte à língua portuguesa e presença de conteúdos gratuitos.

O FreeCodeCamp, lançado em 2021, é o mais recente na lista analisada neste trabalho. É uma aplicação *mobile* e web, criada pela organização sem fins lucrativos de mesmo nome, que disponibiliza para estudantes ao redor do mundo formações completas e gratuitas com certificação (também gratuita), em diversas linguagens de programação. Utiliza como estratégia de gamificação a contagem de *streaks* e pontuação por resolução de exercícios e acesso às aulas.

(a)



(b)

Figura 1. Página de acesso a cursos e página de conteúdo de curso na ferramenta FreeCodeCamp. Fonte: Autoria própria.

Possui interface *web* de fácil utilização (Figura 1A), com conteúdo disponibilizado junto a um editor de código (Figura 1B). Todo o conteúdo é disponibilizado em português, assim como a própria interface da ferramenta, facilitando o uso de quem não possui proficiência na língua inglesa.

Mimo é um aplicativo gamificado de ensino de programação, que busca incentivar alunos a aprenderem de forma gratuita (com alguns recursos liberados apenas com acesso pago), utilizando comunidades, ligas, e outros métodos baseados em jogos de videogame. Foi lançado em 2018, possuindo além da versão *mobile*, acesso completo *web*. A versão *mobile* possui tradução para a língua portuguesa, diferentemente da versão *web*, que está disponível apenas em inglês, limitando o acesso e a compreensão pelo navegador.

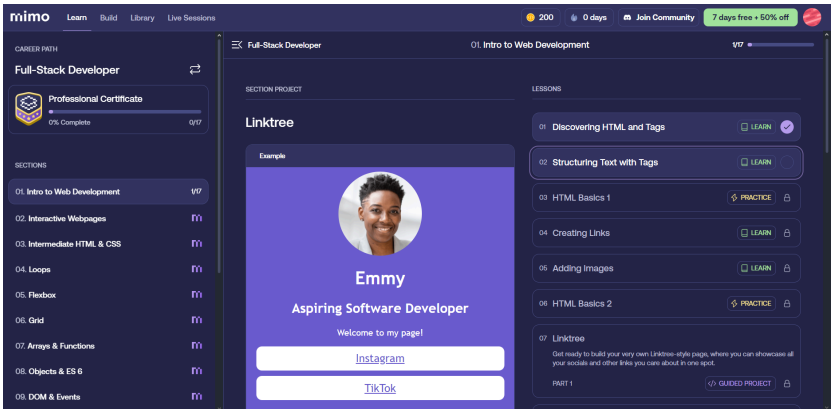
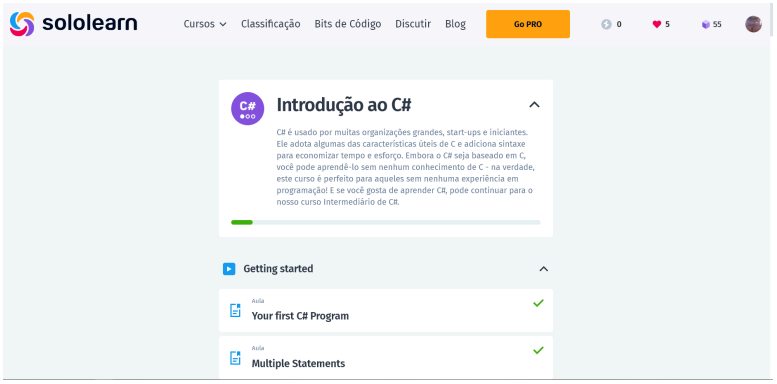


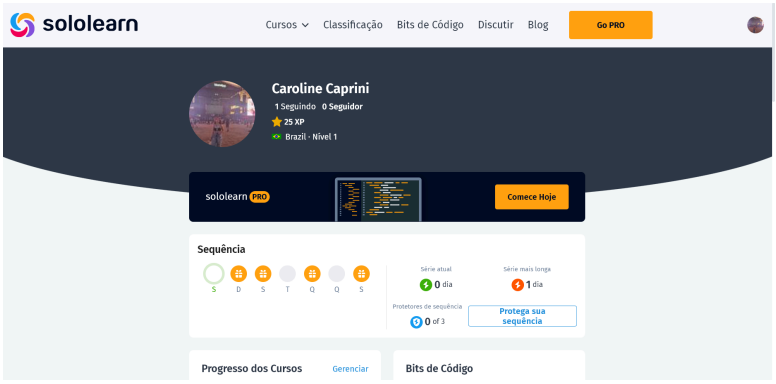
Figura 2. Tela inicial da aplicação *web* Mimo. Fonte: Autoria própria.

O acesso aos cursos é gratuito, sendo que pela versão *web* são disponibilizados apenas os cursos introdutórios, obrigando o aluno a baixar a versão *mobile* para poder continuar. Acesso a certificados e recursos ilimitados estão restritos ao plano pago da plataforma. Possui interface de fácil utilização pelo usuário no acesso *web*, sendo que os principais pontos de uso da plataforma se concentram na tela principal (Figura 2).

Sololearn é a ferramenta que está a mais tempo disponível ao público, lançada em 2016, com grande relevância no aprendizado de programação. É completo, tanto em sua versão *mobile* como web, possuindo cursos nas mais diversas linguagens de programação e níveis de dificuldade, utilizando ferramentas de gamificação em busca de facilitar a absorção do conhecimento, e apresentando o conteúdo de forma direta em sua tela inicial (Figura 3A). Com ligas de competição, desafios entre a comunidade, medalhas e fóruns, o aprendizado do aluno é incentivado pela plataforma. Também utiliza de recursos como vidas, que são perdidas se o usuário executar erros durante a lição ou selecionar respostas incorretas, *streaks* marcando a sequência diária de aprendizado (Figura 3B), e bits, que são utilizados como moeda.



(a)



(b)

Figura 3. Tela de acesso à aulas e cursos e tela de perfil do usuário na ferramenta Sololearn. Fonte: Autoria própria.

Sua versão premium não limita acesso a cursos e conteúdos, mas libera o recurso de vidas infinitas, em que o usuário não mais depende de carregamentos de vida diários para poder praticar, o recurso de prática de código com a IA da plataforma e bloqueia anúncios. Já em relação a sua disponibilização na língua portuguesa, a plataforma não possui traduções para o idioma de forma nativa, sendo estas disponibilizadas por um recurso agregado que realiza traduções automáticas diretamente nos exercícios, não possuindo exatidão ou garantia de tradução correta.

Aplicação x Método de avaliação	Formas de acesso	Gratuito (cursos, avaliações, exercícios, revisões)	Gamificação	Suporte para a língua portuguesa
FreeCodeCamp (2021)	Web e *mobile*	100% gratuito	Utiliza sistema de *streak*, marcação diária de práticas	Sim
Mimo (2018)	Web e *mobile*	Conteúdo (cursos e exercícios) limitado na versão gratuita, ilimitado na *premium*	Interface gamificada, liga de competição entre os estudantes, sistema de *streak*, pontos de experiência, sistema de vidas	Parcial
Sololearn (2016)	Web e *mobile*	Cursos são totalmente gratuitos, *premium* inclui apenas vantagens extras	Criação de desafios de programação entre a comunidade, sistema de vidas, sistema de *streaks*, liga de competição	Parcial

Figura 4. Comparativo entre todas as ferramentas analisadas. Fonte: Autoria própria.

Conforme visto nesta análise, há diferentes ferramentas disponíveis no mercado que buscam ensinar de forma gamificada conceitos, lógica e linguagens de programação, mas nem todas possuem completa acessibilidade linguística, disponibilidade independente do dispositivo de acesso ou versão completa gratuita. Para a aplicação desenvolvida neste trabalho, buscou-se utilizar os pontos positivos de cada plataforma em busca de criar algo totalmente acessível, inicialmente via navegador, disponível em português e sem recursos pagos pelo estudante.

3. Desenvolvimento

Esta seção discorre sobre as tecnologias que foram selecionadas em busca de desenvolver a ferramenta de forma acessível, as técnicas de gamificação selecionadas para utilização neste projeto, assim como a elaboração dos níveis de dificuldade e conteúdos abordados pela ferramenta.

3.1 Tecnologias de Desenvolvimento Selecionadas e sua Utilização no Projeto

Figura 5. Demonstrativo de todas as tecnologias utilizadas. Fonte: Autoria própria.

Para o desenvolvimento desta ferramenta, foi optado pelo formato de jogo de plataforma 2D (duas dimensões) em *pixel art*, em que o personagem pode se movimentar em apenas um plano de jogo, e as ilustrações são de baixa resolução, utilizando de uma técnica que permite que sejam elaboradas animações detalhistas de forma rápida.

Os recursos visuais e ilustrações utilizados neste projeto para a criação de personagens e das fases foram disponibilizados em itch.io[4] e opengameart.org[5] pela comunidade de forma gratuita, sendo que no caso dos personagens, foi feita a alteração de cor para permitir a escolha do usuário, e ajuste em quadros da animação para gerar uma animação completa, com movimentação das orelhas e focinho quando ocioso, utilizando a ferramenta Pixel Studio[6].

Cada personagem possui quatro animações, sendo elas ocioso, correndo, pulando e caindo, sendo que cada uma delas é controlada pelo método *SpriteAnimation* do motor Flame, e definidas conforme o bloco de movimento selecionado no editor de código (Figura 6).

Figura 6. Animação do personagem nos modos de ocioso, corrida, pulo e queda. Fonte: OpenGameArt com modificações de autoria própria.

Para a criação dos mapas das fases, foi utilizada a ferramenta Tiled[7] (Figura 7), um editor de níveis para jogos 2D gratuito e *open-source*, que utiliza da técnica de desenho com blocos de pixels, suporta diversas camadas, e possibilita ao desenvolvedor uma ampla gama de possibilidades, como colocação livre de imagens, adição de informações em cada componente criado, como posição e nomes de classes que podem ser integradas ao código (Tiled, 2024).

[4] Pixel Adventure by Pixel Frog. Disponível em: <https://pixelfrog-assets.itch.io/pixel-adventure-1>.
[5] REDSHRIKE. Bunny Rabbit LPC style for PixelFarm. Disponível em: <https://opengameart.org/content/bunny-rabbit-lpc-style-for-pixelfarm>.
[6] AM, H. G. Pixel Studio: pixel art editor. Disponível em: <https://play.google.com/store/apps/details?id=com.PixelStudio&hl=pt_BR>.
[7] Tiled. Disponível em: <https://www.mapeditor.org>.

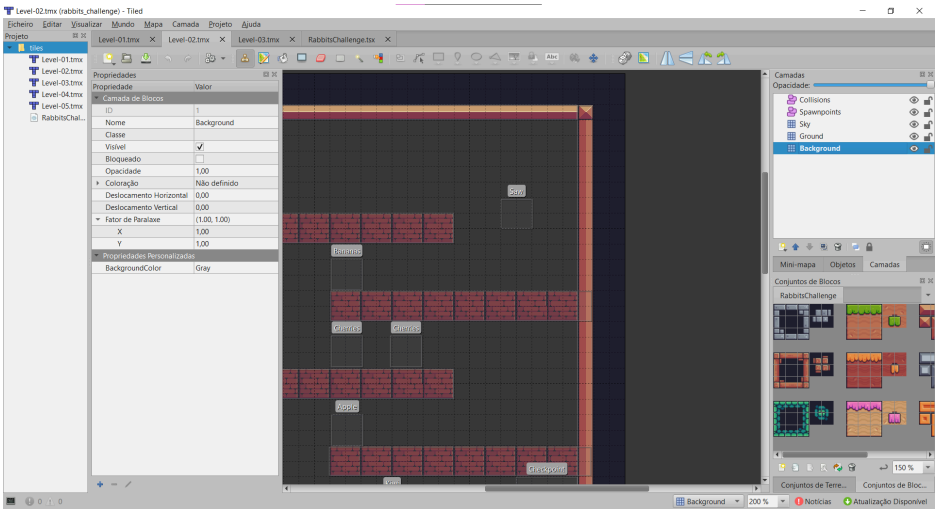


Figura 7. Visão geral do software Tiled. Fonte: Autoria própria.

Em relação ao código e seu acesso pelo usuário final, foram selecionadas tecnologias que permitem sua futura transformação em uma aplicação multiplataforma, possibilitando sua futura instalação como aplicativo mobile e *desktop*, além de seu acesso *web* já funcional. Em busca de criar uma aplicação com integração *seamless*, foi optado pela busca de ferramentas que possuem integração nativa entre si, como as tecnologias desenvolvidas pela Google.

Para isso, foi definido o uso das *framework* Flutter e Flame, pertencentes à linguagem Dart, desenvolvida pelo Google. Flutter é uma tecnologia *open-source* que permite criar aplicações multi-plataformas, compiladas nativamente, para diversos tipos de tela e dispositivos, utilizando o mesmo código, sem necessitar de adaptações. Também possui integrações nativas com diversos tipos de *hardware* e serviços, como transações financeiras, armazenamento em nuvem, autenticação e marketing (Flutter, 2024).

Flame é um motor para criação de jogos baseado em Flutter, sendo uma *framework* de uso simplificado que utiliza a arquitetura já provida pelo Flutter para ser utilizada. Possui as principais funcionalidades necessárias para a criação de jogos 2D, como entradas do usuário, manipulação de imagens, *sprites* e *sprite sheets*[8] (pequenas animações sequenciais, feitas quadro a quadro), efeitos sonoros, animações, detecção de

[8] *Sprites* são elementos gráficos 2D utilizados para construir personagens, elementos e cenas de computação gráfica em jogos. Os *spritesheets* são vários *sprites* de um mesmo objeto agrupados em sequência, formando uma animação.

colisões e gravidade, entre outros (Flame, 2024). Como é baseado em Flutter, toda a sua integração às demais telas de acesso como cadastro de conta, *home screen*, notificações, é facilitado.

Já para a criação de telas do aplicativo externas ao jogo, como cadastro de usuário, login, acesso aos níveis, perfil e edição de dados foi selecionada a tecnologia FlutterFlow (Figura 8), devido ao seu potencial de construção rápida de aplicações completas por utilizar-se da metodologia *low-code*[9] para a criação de lógica, integrações com código nativo (*high code*[10]), banco de dados, e criação de telas de acesso multiplataforma, com suporte nativo à diversos idiomas, e utilização intuitiva por parte do usuário.

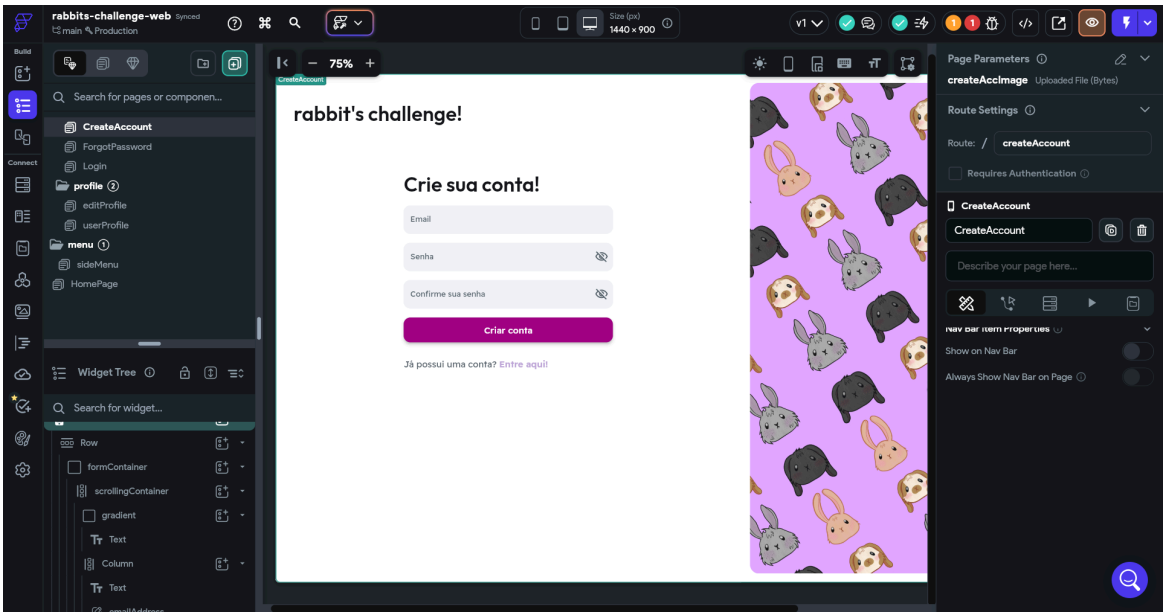


Figura 8. Editor de telas (*front-end)* do FlutterFlow. Fonte: Autoria própria.

Em vez de linhas de código, o FlutterFlow utiliza de ações para indicar o que cada fluxo deve realizar dentro do aplicativo em desenvolvimento, desde operações lógicas como cálculos, exibição de textos ou consultas e operações em banco de dados, conforme demonstrado na Figura 9, em que é utilizada a ação "Create Account" do módulo de autenticação para que a conta do usuário seja criada em banco, disparando em seguida uma ação de envio de email de verificação de contato do usuário.

---

[9] *Low-code* é um formato de desenvolvimento de software onde equipes podem montar ferramentas completas em baixo tempo e com agilidade, e com o mínimo de codificação tradicional.
[10] *High-code* é a codificação tradicional no desenvolvimento de software.

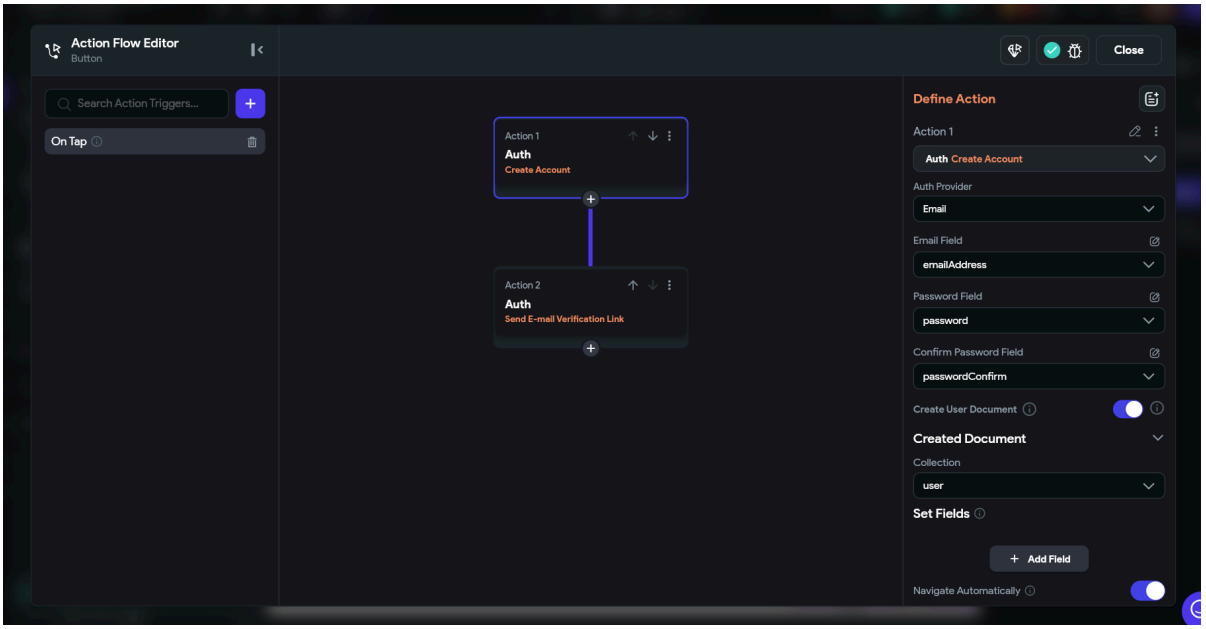


Figura 9. Editor de fluxos lógicos (*back-end)* do FlutterFlow. Fonte: Autoria própria.

Para o armazenamento de dados dos jogadores, foi selecionado o Cloud Firestore, um banco de dados NoSQL que possui hospedagem em nuvem na ferramenta de desenvolvimento *backend-as-a-service* Firebase. Ele pode ser utilizado por aplicações *web,* Android e Apple, utilizando seus SDKs nativos. O Cloud Firestore utiliza uma estrutura lógica que armazena os dados em documentos, estes sendo contidos em coleções, que organizam os dados de forma concisa, possibilitando consultas. Este banco também permite a criação de subcoleções dentro de cada documento, utilização de diversos tipos de dados, desde os mais comuns a tipos complexos, e criação de subcoleções dentro dos documentos, com estruturas de dados hierárquicas que podem ser escalonadas de acordo com o tamanho da base de dados (Firebase, 2024).

Como toda a lógica de cadastro, login, verificação de usuário e alteração de senha foi criada na ferramenta FlutterFlow, toda a configuração do Cloud Firestore foi realizada diretamente na plataforma durante o desenvolvimento (Figura 10), o que deixou a configuração inicial do projeto ainda mais dinâmica.

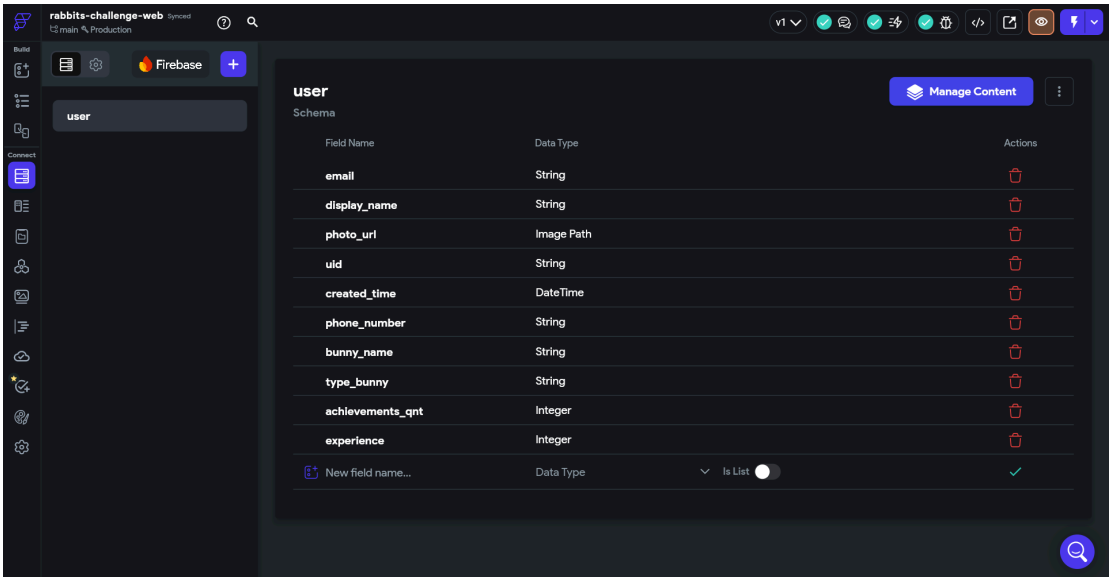


Figura 10. Módulo de integração do Cloud Firestore no FlutterFlow. Fonte: Autoria própria.

Para este projeto, foi criada apenas a coleção **user**, que armazena os dados informados pelo usuário ao criar sua conta e ao editar seu perfil, sendo que alguns campos foram adicionados automaticamente pela plataforma *low code*, como photo_url e phone_number, inicialmente não utilizados neste desenvolvimento, e created_time, gerada automaticamente ao executar a ação de autenticação na plataforma. Abaixo, segue exemplo da estrutura dos dados dentro do Cloud Firestore, que segue formatação de árvore de JSON (Figura 11).

```
"users": {
    "jZserWrsQhV6xOsdpybseNu1VZC3": { //uid
        "bunny_name": "Mingau", //(string)
        "created_time": 24 de outubro de 2024 às 23: 37: 43 UTC-3,
//(timestamp)
        "display_name": "Caroline Caprini", //(string)
        "email": "caprinicaroline1@gmail.com", //(string)
        "experience": 275, //(number)
        "type_bunny": "Branco" //(string)
            }
}
```

Figura 11. Estrutura de dados da coleção users no Cloud Firestore. Fonte: Autoria própria.

Para a funcionalidade de criação de blocos de código para solucionar cada desafio imposto pelo jogo, foi selecionada a ferramenta de códigos em bloco Blockly da Google, que não possui integração *mobile* nativa via Flutter, esta sendo integrada ao código Flutter utilizando a biblioteca flutter_blockly. Esta biblioteca disponibiliza o editor de código tanto para acesso *web* como *mobile* utilizando Javascript como linguagem intermediária, os principais blocos de código, liberdade de adição de blocos customizados, como os de movimentos, mudança de direção do personagem e pulo, tudo isso permitindo a experiência de montagem de fluxos lógicos definidos pelo usuário. Neste projeto, desenvolvido inicialmente para acesso *web*, utilizou-se integração entre Javascript e Flutter junto a um *webview* para gerar o editor de código e os respectivos blocos necessários.

Utilizando a integração nativa do Tiled com o Flame (biblioteca flame_tiled) foi possível integrar todo o código do jogo com as fases elaboradas no software. Por serem na mesma linguagem, também foi possível integrar o código-fonte gerado pelo FlutterFlow com o código-fonte do jogo, gerando uma aplicação dinâmica e funcional, com resultados satisfatórios que serão apresentados nas seções subsequentes.

3.3 Elementos de Gamificação Selecionados

Para este projeto foram selecionadas como recursos gamificados de aprendizado pontos de experiência e a customização básica de personagem. Para os pontos de experiência, foi definido que a cada fase completada, o usuário receberá cem pontos de experiência. Se a fase conter frutas, cada fruta captura corresponde a vinte e cinco pontos de experiência. Até o presente momento, os pontos de experiência de conclusão não levam em conta o desempenho ou qualidade do código apresentado pelo usuário, e sim apenas a conclusão do objetivo, sendo ele levar o personagem até a linha de chegada.

Já para customização do personagem está limitada a apenas a escolha da cor do coelho, sendo elas caramelo, cinza e branco (Figura 12). Ao escolher a cor do personagem na tela de edição de dados do usuário, o personagem padrão do jogo é alterado para o escolhido pelo usuário.

Figura 12. Diferentes cores de personagem disponíveis para escolha. Fonte: Autoria própria.

Com a customização de personagens já existente dentro do jogo, é possível futuramente adicionar novas cores assim como um sistema de recompensas em que pontuação ou uma moeda da plataforma possa ser trocada por cores especiais e temáticas ou itens cosméticos, incentivando ainda mais o aluno a solucionar desafios em troca de recompensas.

3.4 Elaboração dos Níveis de Dificuldade e Conteúdos Apresentados

Para a criação dos mapas de níveis, compreendeu-se que para que o jogador tenha sua atenção engajada, é necessário apresentar o objetivo de forma clara, com cada nível apresentando diferentes desafios, de forma planejada para que a motivação se mantenha fresca. Para isso, foi estudado o design de nível de dois jogos utilizados como inspiração, Bread & Fred (Figura 13) e Celeste (Figura 14), o primeiro lançado pela distribuidora SandCastles Studio em 2023 e o segundo, pela distribuidora Maddy Makes Games Inc., em 2018.

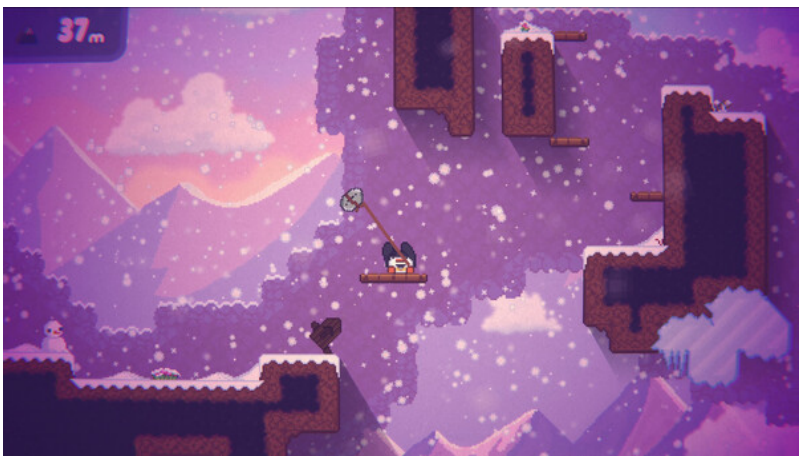


Figura 13. Nível do jogo Bread & Fred. Fonte: Steam.

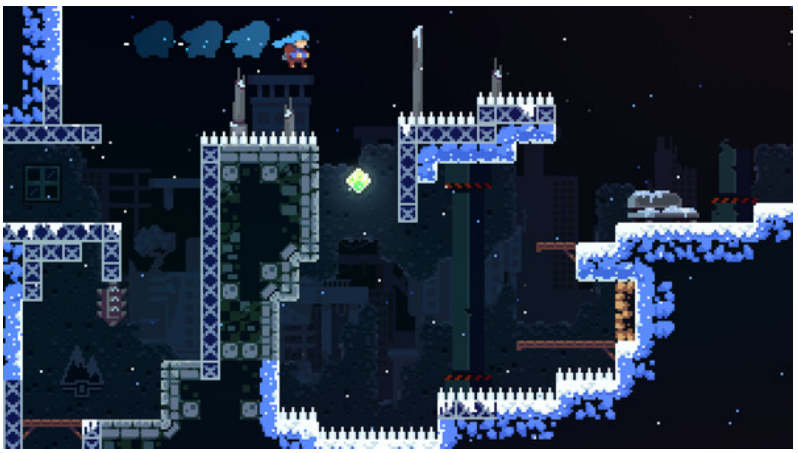


Figura 14. Nível do jogo Celeste. Fonte: Steam.

Ambos possuem um ritmo de jogo que exige reações rápidas de seus jogadores, e um bom aproveitamento do espaço em tela para a criação de plataformas e desafios, como a captura de itens ou armadilhas.

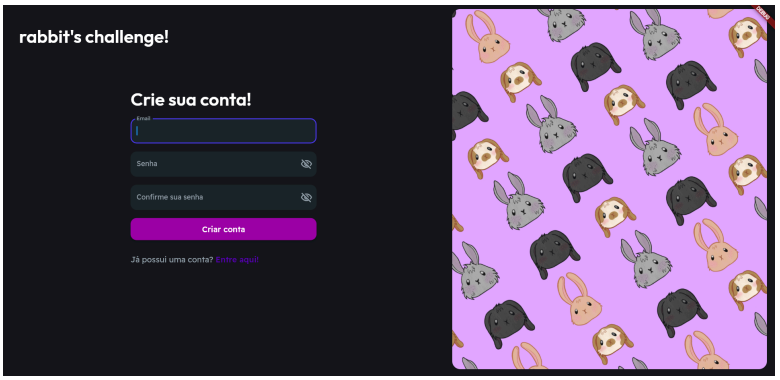
Já em relação ao conteúdo apresentado, por ser uma ferramenta apenas de apoio ao professor no ensino em sala de aula, não foram planejados módulos de aula para a plataforma, assim, explicações mais extensas relacionadas a conteúdo programático das aulas não se fazem presentes dentro do jogo.

4. Resultados

Utilizando como inspiração as ferramentas analisadas na metodologia deste trabalho, foi criada uma aplicação de acesso *web*, em que o aluno pode criar seu próprio perfil e iniciar a resolução dos desafios propostos. Até o presente momento, a aplicação desenvolvida neste trabalho se mostra promissora como instrumento de apoio em sala de aula.

Ao acessar o sistema pela primeira vez, o usuário é encaminhado à tela de *login* (Figura 15A), em que pode ser direcionado à ação de criação de conta, ou alteração de senha.

(a)



(b)

Figura 15. Telas de acesso ao sistema. Fonte: Autoria própria.

Selecionando a opção de criação de conta, o usuário indica seus dados de acesso, como email e senha (Figura 15B), e ao efetuar a ação de cadastro dos dados no Cloud Firestore, é conduzido à tela inicial da aplicação, onde o acesso aos desafios e perfil do usuário é realizado (Figura 16).

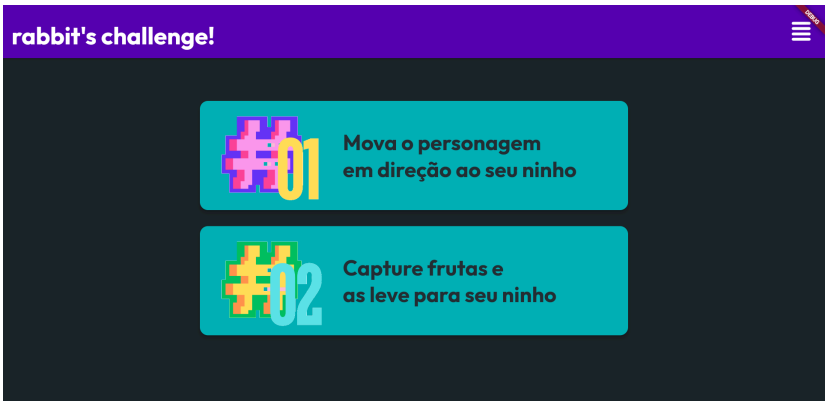


Figura 16. Telas inicial do sistema. Fonte: Autoria própria.

Ao acessar o primeiro desafio, o usuário é apresentado ao tutorial do funcionamento da ferramenta Blockly (Figura 17) e os principais comandos utilizados para solucionar cada fase. Como ambos os desafios desenvolvidos até o momento podem ser resolvidos utilizando apenas blocos de laços de repetição, condicionais simples e blocos customizados, os demais (operações matemáticas, operações entre textos, listas, variáveis e funções) não foram disponibilizados aos usuário, podendo ser acrescentados conforme o desenvolvimento de novos níveis.

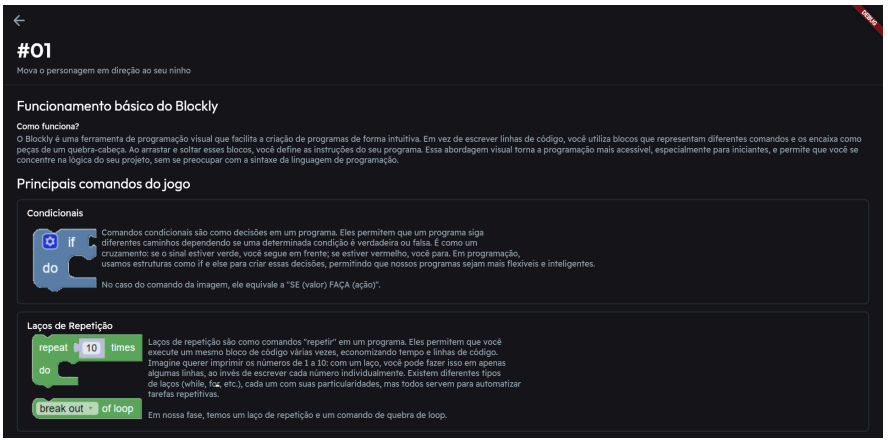


Figura 17. Página introdutória do nível. Fonte: Autoria própria.

Após a introdução o usuário tem acesso ao primeiro nível do desafio, em que o coelho deve ser guiado até o ponto de chegada, indicado pela bandeira (Figura 18), com o propósito de guiar o aluno a aprender a utilizar a ferramenta. A fase é integrada com a área de código do Blockly, em que o usuário pode criar as combinações de blocos necessárias para atingir o objetivo do nível, utilizando o botão de “Compilar e Executar Código”.

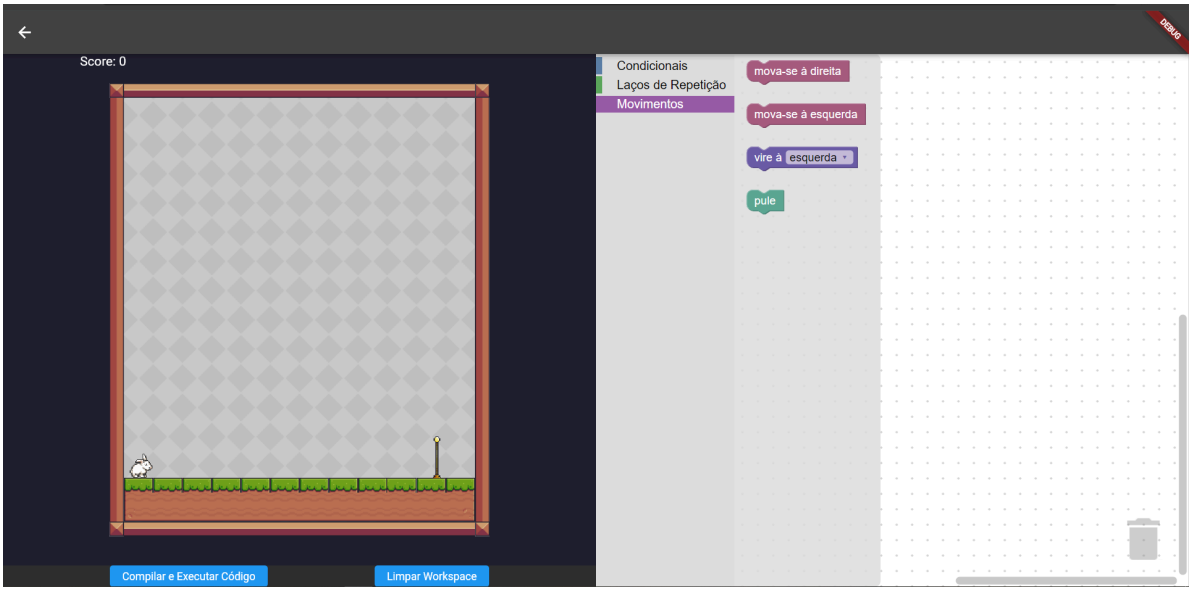


Figura 18. Fase 1. Fonte: Autoria própria

Já na fase 2, além de levar o personagem em direção a linha de chegada, o aluno deve capturar frutas pelo caminho, aumentando os pontos de experiência obtidos a cada fruta capturada (Figura 19).

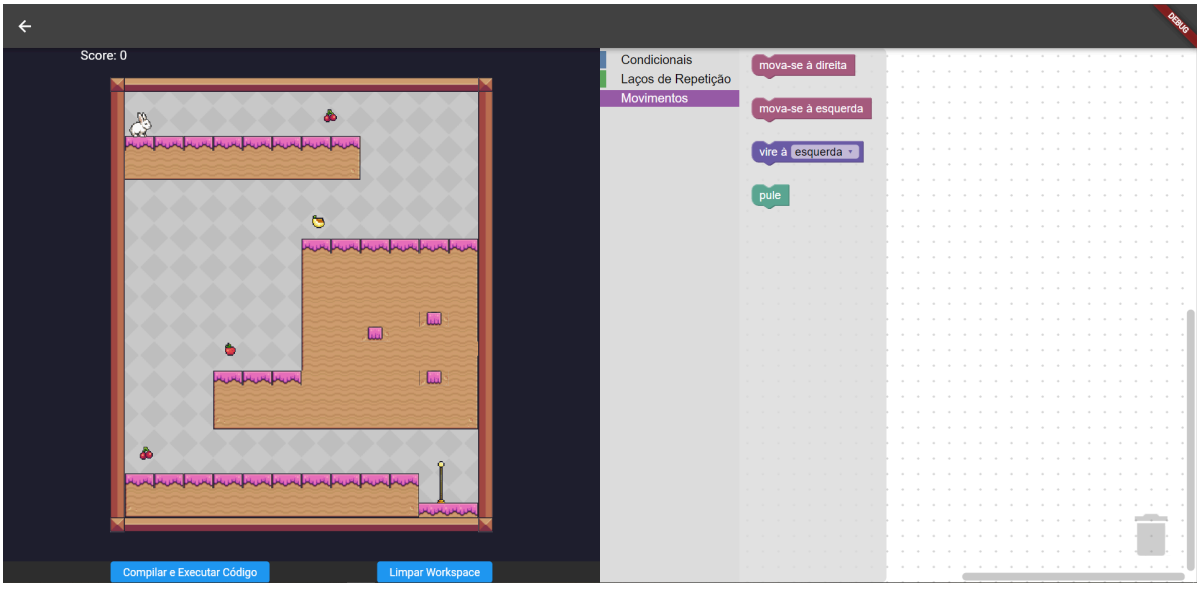


Figura 19. Fase 2. Fonte: Autoria própria

Em ambos os desafios, é apresentada a possibilidade de utilizar todos os blocos de código oferecidos pelo Blockly e configurados em código, permitindo que desde o início da utilização da plataforma, o aluno possa criar repetições de código e

condicionais, além de fazer o coelho mover-se linha a linha. Assim, o aluno possui mais liberdade e criatividade em criar sua sequência de instruções para movimentar o personagem.

Ao finalizar as fases completando o objetivo proposto, o usuário é direcionado a uma tela que apresenta o *feedback* de pontuação obtido (Figura 20).

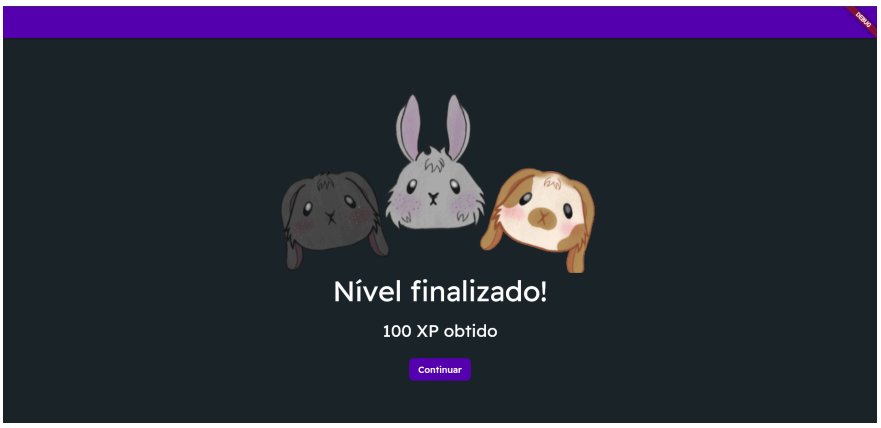


Figura 20. Tela de *feedback* de pontuação do usuário. Fonte: Autoria própria.

A pontuação obtida também é exibida no perfil do usuário, sendo que em esta tela é somada a pontuação já obtida pelo aluno, gravadas em banco de dados e recuperadas no ato do acesso (Figura 21).

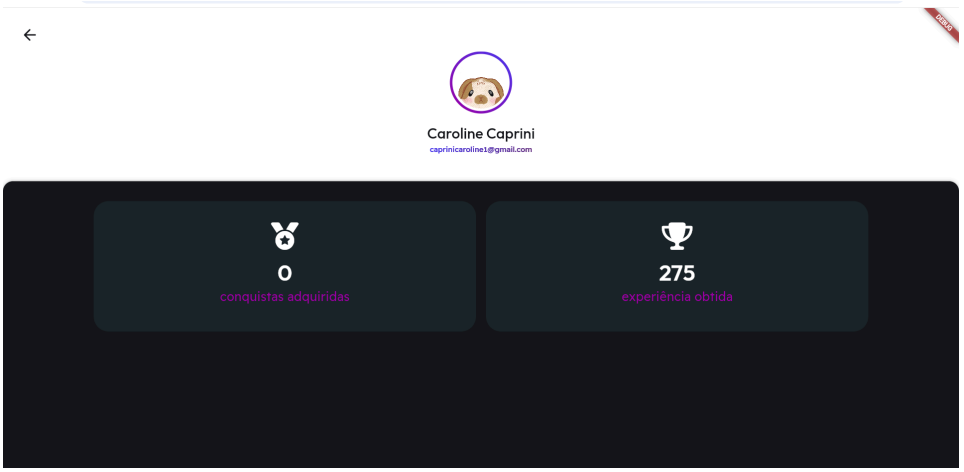


Figura 21. Perfil do usuário. Fonte: Autoria própria.

A alteração de dados de cadastro do usuário assim como a escolha da cor do personagem jogável é realizada na tela de edição de perfil, acessível pelo menu lateral

(Figura 22). Nela também é possível acessar o perfil do usuário e deslogar da plataforma.

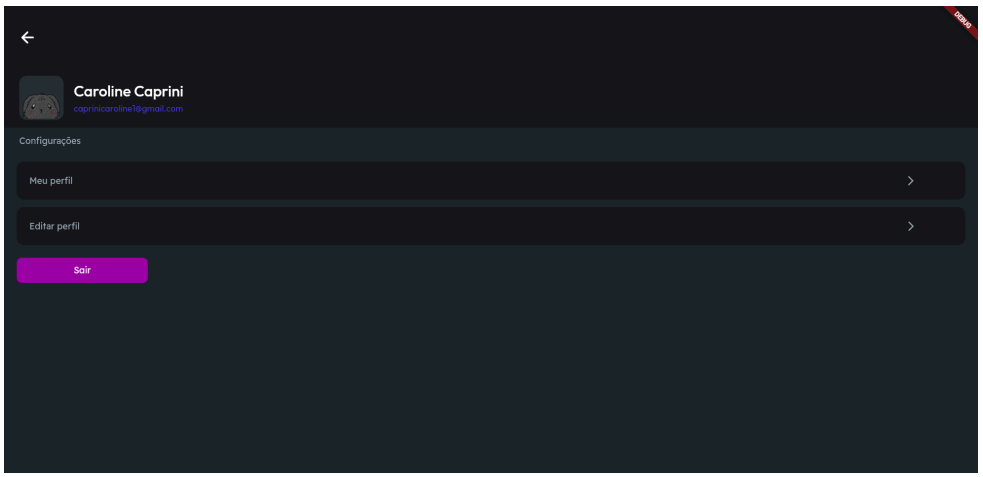


Figura 22. Página introdutória do nível. Fonte: Autoria própria

Na tela de edição do perfil (Figura 23), é possível criar um nome para o personagem do usuário, funcionalidade que pode ser integrada à futuras customizações de personagem que o aluno possa fazer, utilizando novas implementações da plataforma.

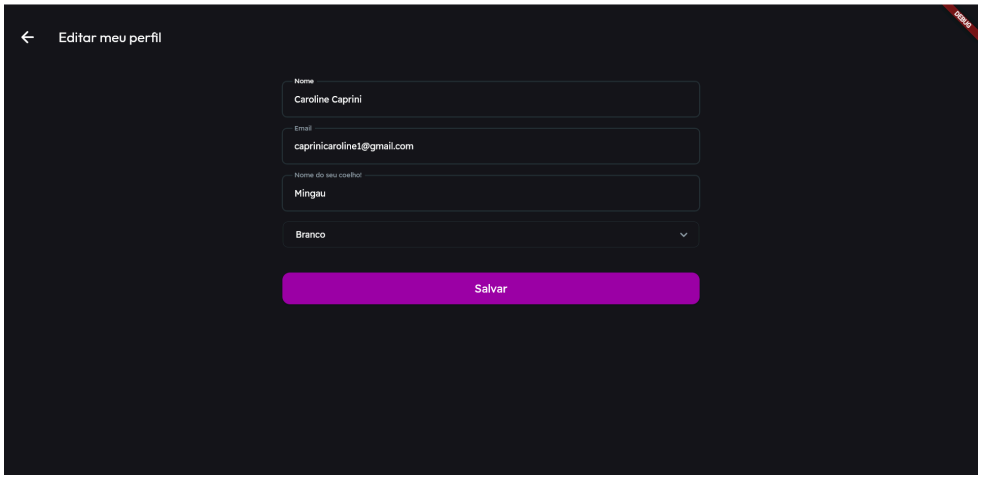


Figura 23. Página introdutória do nível. Fonte: Autoria própria

O presente resultado apresentado nesta seção comprova a possibilidade de criação da ferramenta proposta no trabalho, expandindo a possibilidade de adição de melhorias, como a implementação de novas funcionalidades que serão mencionadas a

seguir, possibilitanto seu uso proveitoso em sala de aula como ferramenta de apoio por professores e alunos.

5. Considerações Finais

É nítido o aumento da procura de profissionais capacitados na área do desenvolvimento de software, e assim, com o desenvolvimento desta ferramenta de apoio, busca-se auxiliar o professor no ambiente de instrução tradicional, a sala de aula, oferecendo a oportunidade de tornar o ensino de conceitos técnicos divertido e demoncrático. Mesmo sendo uma metodologia ainda em crescimento e discussão, a gamificação tem se mostrado uma forte aliada no ensino de novas habilidades e assuntos, principalmente em áreas técnicas como a informática, impulsionando o aluno a se desenvolver de forma prática.

Por ser uma ferramenta com diversas possibilidades de expansão, nota-se as oportunidades de desenvolver como trabalhos futuros a adaptação do código-fonte para que a utilização do editor Blockly seja possível em dispositivos móveis, a expansão dos conteúdos apresentados e dos desafios propostos para que incluam variáveis, listas, funções e demais conceitos de programação, assim aprofundando a prática do aluno, e relacionado a gamificação, a implementação de *streaks* e conquistas, criação de ligas e grupos, para que os estudantes possam competir entre si em busca da maior pontuação, assim como trocar experiências e conhecimentos sobre a resolução de cada desafio, e a implementação da customização completa do personagem, incluindo a criação de uma moeda da aplicação, que possa ser obtida pela resolução de exercícios e contribuição em discussões das ligas e grupos, permitindo sua troca por itens cosméticos.

5. Referências

ALONSO, Élida Froes; FIGUEIREDO, Helenara Regina Sampaio. Evasão em cursos técnicos na área de informática: revisão de literatura de 2015 a 2019. Educitec-Revista de Estudos e Pesquisas sobre Ensino Tecnológico, v. 8, p. e196022-e196022, 2022.

ALVES, Flora. Gamification: como criar experiências de aprendizagem engajadoras. DVS editora, 2015.

BREAD & Fred. 1 ed. ed. Steam: SandCastles Studio, 2023. Disponível em: <https://store.steampowered.com/app/1607680/Bread__Fred/>.

CELESTE. 1 ed. ed. Steam: Maddy Makes Games Inc, 2018. Disponível em: <https://store.steampowered.com/app/504230/Celeste/>.

FIGUEIREDO, José; GARCÍA-PEÑALVO, Francisco José. Increasing student motivation in computer programming with gamification. In: 2020 IEEE Global Engineering Education Conference (EDUCON). IEEE, 2020. p. 997-1000.

FLUTTER. Flutter - Beautiful native apps in record time. Disponível em: <https://flutter.dev/>.

G1 Globo. Área de TI deve gerar quase 420 mil vagas até 2025, mas faltam profissionais. G1 Globo, Jornal Hoje, 2023. Disponível em: https://g1.globo.com/jornal-hoje/noticia/2023/03/13/area-de-ti-deve-gerar-quase-420-mil-vagas-ate-2025-mas-faltam-profissionais.ghtml.

Getting Started — Flame. Disponível em: <https://docs.flame-engine.org/latest/>.

GOOGLE. Cloud Firestore | Firebase. Disponível em: <https://firebase.google.com/docs/firestore>.

HAYS, Robert T. The effectiveness of instructional games: A literature review and discussion. (No Title), 2005.

KAPP, K. M. The gamification of learning and instruction: Game-based methods and strategies for training and education. Pfeiffer, 2012.

QIAN, Yizhou; LEHMAN, James. Students' misconceptions and other difficulties in introductory programming: A literature review. ACM Transactions on Computing Education (TOCE), v. 18, n. 1, p. 1-24, 2017.

VOGEL, Jennifer J. et al. Computer gaming and interactive simulations for learning: A meta-analysis. Journal of educational computing research, v. 34, n. 3, p. 229-243, 2006.

ZHAN, Zehui, et al. The effectiveness of gamification in programming education: Evidence from a meta-analysis. Computers and Education: Artificial Intelligence, p. 100096, 2022.