MathColor: Uma Aplicação para o Aprendizado de Matemática nos Anos Iniciais

Ane Luisy Grizotti[1], **Rafaela Santos da Rosa**[1],
Robson Costa[1], **Vilma Gisele Karsburg**[1]

[1]Instituto Federal de Santa Catarina (IFSC)
R. Heitor Villa Lobos, 255 - São Francisco, 88506-400- Lages - SC – Brasil

ane.lg@aluno.ifsc.edu.br, rafaela.r19@aluno.ifsc.edu.br,

robson.costa@ifsc.edu.br, vilma.karsburg@ifsc.edu.br

Abstract. *This article addresses the challenge of teaching mathematics to students in Elementary School I and presents the development of the MathColor application. The project aims to create software that engages children in mathematical activities in a playful and interactive way, combining the resolution of mathematical questions with the reward of coloring drawings. The research utilizes technology to alleviate difficulties in childhood learning, and the positive results of the application were obtained through a questionnaire administered in the classroom with students, highlighting its effectiveness in the teaching-learning process.*

Resumo. *Este artigo aborda o desafio do ensino de matemática para alunos do Ensino Fundamental I e apresenta o desenvolvimento do aplicativo MathColor. O projeto tem como objetivo o desenvolvimento de um* software *para envolver crianças em atividades matemáticas de forma lúdica e interativa, combinando a resolução de questões matemáticas com a recompensa de colorir desenhos. A pesquisa utiliza a tecnologia para amenizar as dificuldades no aprendizado infantil e os resultados positivos do aplicativo foram obtidos por meio de um questionário aplicado em sala de aula com alunos, o que destacou sua eficácia no processo de ensino-aprendizagem.*

1. Introdução

A matemática é tida, frequentemente, como um conhecimento complicado e pode ser considerada ainda mais complexa quando ensinada para crianças do Ensino Fundamental, quando é tido o primeiro contato com este componente curricular. Além disso, de acordo com Sadovsky (2007), o baixo desempenho dos alunos em matemática é algo recorrente em muitos países, inclusive no Brasil. Neste sentido, é importante desenvolver atividades diferenciadas para auxiliar no seu processo de ensino-aprendizagem desde as séries iniciais.

Ainda assim, ter dificuldades em matemática não está relacionado apenas a problemas de aprendizagem da criança, estas dificuldades estão associadas em partes, com a falta de interesse devido as formas tradicionais de ensino, como apenas o uso de quadro e giz. De acordo com Boaler (2016), o fracasso de alunos em matemática não é sem fundamento, trata-se de uma mensagem de que os métodos de ensino e aprendizagem não

estão funcionando e que é necessário mudar as abordagens para encontrar caminhos mais acessíveis e compreensíveis para o aprendizado. Desta forma, os métodos tradicionais ocasionam uma sensação de que a matemática é uma disciplina desinteressante e difícil.

Além disso, nota-se que a tecnologia revolucionou a forma de lidar com as informações e o uso desta no ensino pode proporcionar um ambiente de melhor aprendizagem, oferecendo uma alta produtividade nos estudos e diversas formas para aplicar o conhecimento. Por conseguinte, conforme Boaler (2019), as escolas não podem se dar ao luxo de ignorar as tecnologias digitais em seus métodos de ensino, pois quando usada corretamente, ela pode enriquecer a experiência educacional, permitindo que os alunos acessem informações, colaborem em projetos e criem produtos que seriam difíceis ou impossíveis sem ela.

O ato de praticar brincadeiras e utilizar jogos dentro da sala de aula, de acordo com Piaget (1975), não são apenas formas de entretenimento para as crianças, pois são meios que contribuem para o entendimento de conteúdos e enriquecem o desenvolvimento intelectual de uma forma divertida. Nesta mesma ideia, Kishimoto (2011) afirma que o jogo exerce um papel importante na educação matemática, pois através destes, é trabalhado o desenvolvimento integral do indivíduo. Assim, o uso de jogos aliado ao uso de tecnologias pode contribuir para que as aulas sejam mais atrativas aos estudantes.

Desta forma, o objetivo geral deste trabalho é realizar a criação de um aplicativo *mobile* com tema infantil para facilitar e incentivar crianças do terceiro ano do Ensino Fundamental a aprenderem desde os primeiros conceitos de matemática aos conceitos mais elaborados, de acordo com sua faixa-etária de uma forma divertida e eficaz.

Têm-se como objetivos específicos:

- Estudar artigos e *softwares* para entender as necessidades dos estudantes;
- Prototipar e estabelecer o escopo do aplicativo;
- Implementar o aplicativo *mobile* utilizando *Flutter*;
- Avaliar o aplicativo em uma escola com o público alvo.

A respeito da construção do trabalho, ele divide-se em quatro etapas. Na primeira etapa, foram realizados estudo de artigos, livros e *softwares*, a fim de desenvolver o referencial teórico do artigo. Na segunda etapa, foi realizado o levantamento dos requisitos, para que, então, serem realizadas as prototipagens para avaliar a navegabilidade e a experiência do usuário, além de permitir uma visão clara e objetiva do projeto. Em seguida, na terceira etapa, foi efetuado o desenvolvimento do aplicativo móvel, com base nos recursos recolhidos anteriormente. Por fim, na quarta etapa, foi realizada uma avaliação em uma turma de terceiro ano do Ensino Fundamental, em uma escola do município de Lages/SC, para obter os resultado e analisá-los em seguida.

Quanto a metodologia, de acordo com Zanella (2013), este trabalho classifica-se como uma pesquisa científica aplicada, visto que tem a finalidade de amenizar as dificuldades do aprendizado infantil por meio da tecnologia. A respeito dos objetivos, é exploratória, pois visa a criação de uma ferramenta que irá facilitar o entendimento de conteúdo matemático nos anos iniciais. Em relação a abordagem, adota-se uma perspectiva qualitativa, descrevendo o tema por meio de impressões, pontos de vista e opiniões. Além disso, também incorpora uma abordagem quantitativa, uma vez que foram utilizados formulários, construídos gráficos e calculadas porcentagens para analisar resultados. No que

se refere aos procedimentos, configura-se como uma pesquisa experimental, envolvendo a avaliação do aplicativo em sala de aula.

Este artigo divide-se em 5 seções. Na seção 1, encontram-se o tema principal, com a sua problemática e os pontos que serão analisados e desenvolvidos no decorrer deste trabalho. Já na seção 2, estão localizados os referenciais teóricos, bem como a seleção dos tópicos trabalhados em matemática no terceiro ano do Ensino Fundamental. Nesta seção também estão contidos os conteúdos que serão abordados no aplicativo, as referências do uso da tecnologia nos anos iniciais da escola e as linguagens de programação e *frameworks* utilizados. Na seção 3 é descrito todo o desenvolvimento do sistema elaborado, contendo modelagem e a implementação do aplicativo e do banco de dados. A respeito da seção 4, refere-se aos resultados da avaliação do aplicativo e por fim, na seção 5, abrangem-se as considerações finais e trabalhos futuros.

2. Referencial Teórico

Esta seção trata-se dos conteúdos que embasam a proposta elaborada, sendo divida em cinco subseções. Na subseção 2.1, é abordada a importância da tecnologia e dos jogos no ensino de matemática. Em sequência, na subseção 2.2, é dissertado sobre os tópicos trabalhados no terceiro ano do Ensino Fundamental. Já na subseção 2.3, são levantados os conteúdos abordados no aplicativo. Na subseção 2.4, são apresentados os sistemas similares e por fim, na subseção 2.5, são expostas as tecnologias utilizadas na composição de aplicativos móveis.

2.1. A importância da tecnologia e dos jogos digitais no ensino de Matemática

A utilização da tecnologia no ensino da matemática teve seu início na década de 1980, com a criação da linguagem de programação LOGO por Seymour Papert, matemático e educador do Instituto de Tecnologia de Massachusetts (MIT), juntamente com seus colegas. Desde então, as tecnologias digitais têm desempenhado um papel cada vez mais relevante na educação matemática, transformando a maneira como os conceitos e habilidades são ensinados e aprendidos. De acordo com Borba et al. (2014), a evolução da tecnologia digital na educação matemática pode ser dividida em quatro fases distintas que serão elencadas a seguir:

- Primeira fase: as tecnologias digitais são usadas como substitutas das ferramentas tradicionais de ensino;
- Segunda fase: as tecnologias digitais começam a ampliar as possibilidades de ensino e aprendizagem;
- Terceira fase: as tecnologias digitais têm o potencial de transformar profundamente a forma como a matemática é ensinada e aprendida;
- Quarta fase: as tecnologias digitais ainda estão em desenvolvimento e apresentam potencialidades futuras ainda não completamente exploradas.

A quarta e última fase de desenvolvimento teve seu início no ano de 2004 com o avanço tecnológico. A partir de então, especialmente no que diz respeito às tecnologias digitais e à internet rápida, tornou-se cada vez mais comum e precoce o uso dessas tecnologias, principalmente por crianças. De acordo com dados do Instituto Brasileiro de Geografia e Estatística, o IBGE (2019), 85,4% das crianças brasileiras entre 9 e 17 anos utilizaram a internet, sendo que 77,9% acessaram a rede pelo celular. Paiva (2021) afirma

que durante a pandemia da COVID-19, houve um crescimento significante de crianças na faixa-etária de 7 a 9 anos que possuem um celular próprio, bem como um aumento no tempo de uso diário deste dispositivo.

Seguindo esta ideia, desde os primeiros anos de vida, grande parte das crianças começam a se familiarizar com a tecnologia. Desta forma, conforme Borba et al. (2014), não faz sentido a ausência de tecnologias digitais em sala de aula, visto que as tecnologias fazem parte do cotidiano da maioria dos alunos. Além disso, o autor afirma que o uso inadequado das tecnologias em sala de aula, também podem afetar o aprendizado, considerando que utilizar a tecnologia em problemas que podem ser resolvidos com o uso de papel e lápis, é dispensável.

Deste modo, os jogos digitais tornam-se uma alternativa para a inclusão de tecnologias em salas de aula. De acordo com Prensky (2012), os jogos digitais estão se tornando cada vez mais relevantes no processo de aprendizagem, pois permitem que as crianças e jovens se envolvam de maneira ativa, colaborativa e social.

Ainda, segundo Prensky (2004), os jogos tornaram-se uma forma natural de aprendizagem para as crianças, as quais estão inseridas em uma cultura digital que preza pela interatividade, colaboração e experimentação. Reforçando este pensamento, Gee (2007) afirma que o uso de jogos digitais no aprendizado é uma forma poderosa e imersiva que contribui para o desenvolvimento de competências essenciais para a vida no século XXI.

Assim, pode-se concluir a grande importância da inclusão dos jogos digitais em sala de aula, auxiliando o aprendizado das crianças e utilizando-se de ferramentas que estão presentes no cotidiano da maioria das crianças.

2.2. Conteúdos trabalhados no terceiro ano do Ensino Fundamental

Conforme descrito na Base Nacional Comum Curricular, a BNCC (2018), os conteúdos de Matemática para o terceiro ano do Ensino Fundamental I são:

1. Números:
    (a) Leitura, escrita, comparação e ordenação de números naturais de quatro ordens;
    (b) Composição e decomposição de números naturais;
    (c) Construção de fatos fundamentais da adição, subtração e multiplicação;
    (d) Reta numérica;
    (e) Procedimentos de cálculo (mental e escrito) com números naturais: adição e subtração;
    (f) Problemas envolvendo significados da adição e da subtração: juntar, acrescentar, separar, retirar, comparar e completar quantidades;
    (g) Problemas envolvendo diferentes significados da multiplicação e da divisão: adição de parcelas iguais, configuração retangular, repartição em partes iguais e medida;
    (h) Significados de metade, terça parte, quarta parte, quinta parte e décima parte.

2. Álgebra:
    (a) Identificação e descrição de regularidades em sequências numéricas recursivas;

(b) Relação de igualdade.

3. Geometria:
   (a) Localização e movimentação: representação de objetos e pontos de referência;
   (b) Figuras geométricas espaciais (cubo, bloco retangular, pirâmide, cone, cilindro e esfera): reconhecimento, análise de características e planificações;
   (c) Figuras geométricas planas (triângulo, quadrado, retângulo, trapézio e paralelogramo): reconhecimento e análise de características;
   (d) Congruência de figuras geométricas planas.

4. Grandezas e medidas:
   (a) Significado de medida e de unidade de medida;
   (b) Medidas de comprimento (unidades não convencionais e convencionais): registro, instrumentos de medida, estimativas e comparações;
   (c) Medidas de capacidade e de massa (unidades não convencionais e convencionais): registro, estimativas e comparações;
   (d) Comparação de áreas por superposição;
   (e) Medidas de tempo: leitura de horas em relógios digitais e analógicos, duração de eventos e reconhecimento de relações entre unidades de medida de tempo;
   (f) Sistema monetário brasileiro: estabelecimento de equivalências de um mesmo valor na utilização de diferentes cédulas e moedas.

5. Probabilidade e Estatística:
   (a) Análise da ideia de acaso em situações do cotidiano: espaço amostral;
   (b) Leitura, interpretação e representação de dados em tabelas de dupla entrada e gráficos de barras;
   (c) Coleta, classificação e representação de dados referentes a variáveis categóricas, por meio de tabelas e gráficos.

De acordo com Pais (2018), um professor de matemática deve envolver os alunos ativamente na aprendizagem, incentivando a participação, utilizando exemplos práticos e relacionando o conteúdo com o cotidiano. O professor deve ser um mediador e facilitador, buscando identificar as dificuldades individuais dos alunos para oferecer apoio personalizado. O autor destaca que é fundamental estabelecer uma relação de confiança e respeito mútuo com os alunos para criar um ambiente de aprendizagem seguro e acolhedor.

Portanto, é importante lembrar que esses conteúdos devem ser trabalhados de forma integrada e contextualizada, visando a formação de um raciocínio matemático consistente e a aplicação dos conhecimentos matemáticos em situações reais do dia a dia.

2.3. Conteúdos trabalhados no *MathColor*

O aplicativo *MathColor* tem como objetivo auxiliar o aprendizado das crianças de forma lúdica e dinâmica, oferecendo um ambiente seguro e amigável para que elas explorem e experimentem a matemática. Desta maneira, o *software* aborda alguns dos assuntos descritos na subseção 2.2, sendo eles:

1. Números:
    (a) Leitura, comparação e ordenação de números naturais de quatro ordens;
    (b) Construção de fatos fundamentais da adição, subtração, multiplicação e divisão;
    (c) Procedimentos de cálculo (mental e escrito) com números naturais: adição e subtração;

2. Grandezas e medidas:
    (a) Medidas de tempo: leitura de horas em relógios digitais e analógicos, duração de eventos e reconhecimento de relações entre unidades de medida de tempo;
    (b) Sistema monetário brasileiro: estabelecimento de equivalências de um mesmo valor na utilização de diferentes cédulas e moedas.

Como pode-se perceber, os tópicos trabalhados no *MathColor* são elencados pela BNCC (2018) e foram aplicados de forma lúdica, buscando estimular o interesse pelo aprendizado da matemática. O aplicativo em questão é uma ferramenta educativa para auxiliar os alunos na compreensão de conceitos fundamentais e na resolução de problemas práticos relacionados a números, grandezas e medidas. O uso do *software* pode tornar o aprendizado mais divertido e envolvente para os alunos, ajudando a superar possíveis dificuldades e tornando os conteúdos matemáticos mais atrativos aos estudantes.

2.4. Sistemas Similares

Esta subseção trata da descrição de aplicativos e sistemas com propostas similares ao trabalho realizado. Serão analisados alguns pontos como atratividade, jogabilidade e usabilidade de cada jogo.

O primeiro aplicativo refere-se ao *Math* Bingo, que é um jogo educativo oferecido pelo *site* ABCya.com (2004), que utiliza questões matemáticas em vez de números para compor as cartelas e as chamadas do jogo. Com diferentes níveis de dificuldade, que variam de acordo com o nível escolar da criança, cada cartela apresenta uma série de problemas matemáticos para serem resolvidos, incluindo adição, subtração, multiplicação e divisão.

Durante o jogo, os jogadores devem procurar a resposta da questão correspondente em sua cartela e resolvê-lo, o jogo é finalizado quando a cartela for preenchida de forma a fazer um bingo, por exemplo, quando uma diagonal for preenchida com as respostas corretas, o jogo é finalizado. Na Figura 1, pode-se observar o *Math* Bingo, que é uma forma divertida e interativa de ensinar e reforçar habilidades matemáticas em crianças de todas as idades, ajudando a manter a motivação e o interesse no aprendizado. É uma opção popular entre pais e professores para complementar o ensino em sala de aula ou em casa.

Em sequência, o aplicativo *MathKids (2023)* é uma ferramenta gratuita que tem como objetivo ensinar números e habilidades matemáticas para crianças. Ele oferece diversos minijogos fáceis e divertidos para crianças em idade pré-escolar.

O objetivo principal do jogo é aprimorar as habilidades matemáticas das crianças, como pode-se analisar na Figura 2, o *Math Kids* possui desafios envolventes, onde as

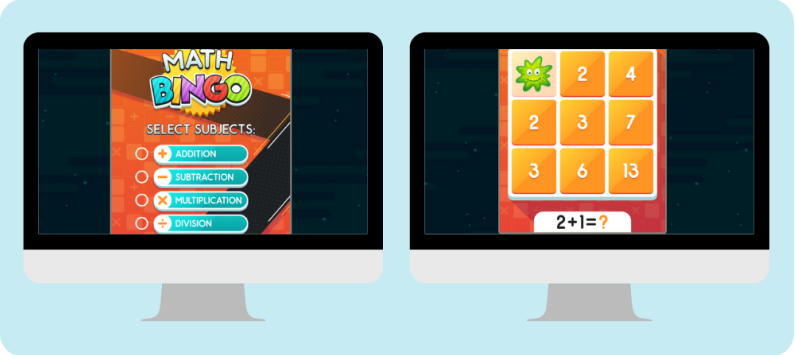

Figura 1. *Math* Bingo

crianças podem praticar contagem, adição, subtração e outras habilidades matemáticas de uma forma divertida e interativa. À medida que os jogadores jogam com maior frequência, é possível obter-se um desenvolvimento progressivo de habilidades nos jogos, o que torna a ferramenta valiosa para pais e educadores que desejam complementar a educação formal de seus filhos. Ele pode ser usado diariamente como uma forma divertida de introduzir conceitos matemáticos e incentivar a aprendizagem. O *Math Kids* pode ajudar na preparação das crianças para o sucesso acadêmico no futuro, tornando a aprendizagem da matemática uma experiência prazerosa e estimulante.

Figura 2. *Math Kids*

Já o *software Division Derby* é um jogo de matemática *online* disponível gratuitamente no *site* MathPlayground (2002), que auxilia no desenvolvimento de habilidades de raciocínio lógico e rapidez de pensamento. O jogo consiste em uma corrida de cavalo, onde o objetivo é responder corretamente às perguntas de divisão para fazer com que o

animal avance na corrida.

Ao iniciar o jogo, deve-se escolher o número de jogadores. Em seguida, o jogador é apresentado a uma pergunta de divisão e precisa responder corretamente em um curto período de tempo para fazer com que seu cavalo avance na corrida. Caso a resposta esteja correta, o cavalo avança, mas se a resposta estiver errada, o cavalo retrocede. O primeiro cavalo a cruzar a linha de chegada é o vencedor.

Division Derby é um jogo divertido e educativo que pode ajudar a aprimorar habilidades em matemática e pensamento rápido. Conforme a Figura 3, este jogo possui uma interface divertida para os alunos, contendo cores atrativas e conteúdos apresentados de forma infantil. O jogo também pode ser utilizado como uma ferramenta complementar em sala de aula para professores que buscam uma forma lúdica e interativa de ensinar matemática.

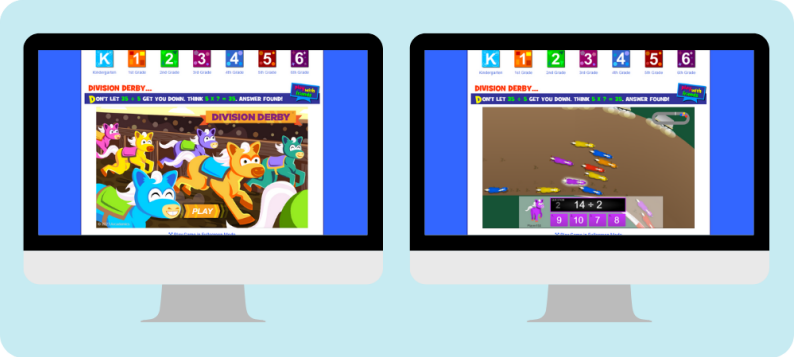

Figura 3. ***Division Derby***

Por conseguinte, o QUIFSC é uma plataforma que foi desenvolvida como uma ferramenta para auxiliar no processo de ensino e aprendizagem de alunos, permitindo que professores criem *quizzes* e utilizem em sala de aula. Esse aplicativo foi criado para oferecer ao professor a possibilidade de elaborar atividades de matemática, utilizando um banco de questões que está dividido em quatro categorias.

Além disso, conforme Cordova e Straubel (2022), a plataforma utiliza técnicas de gamificação para engajar e motivar o usuário. Essa técnica tem como objetivo utilizar elementos presentes em jogos para aumentar o interesse e a participação do aluno nas atividades propostas. Dessa forma, a plataforma QUIFSC se torna uma ferramenta eficiente para professores que desejam incentivar seus alunos a aprender matemática de forma mais lúdica e interativa.

Com o QUIFSC, os professores podem criar *quizzes* personalizados, adaptados às necessidades e objetivos de sua aula, bem como utilizar os *quizzes* já existentes na plataforma. Com isso, é possível oferecer atividades variadas e desafiadoras para os alunos, estimulando o aprendizado e promovendo o desenvolvimento de habilidades em matemática de forma prática e divertida.

Apesar de ser uma plataforma de ensino interativa e engajada, o QUIFSC diferente dos jogos citados, foi desenvolvido para ser utilizado nos anos finais do Ensino Fundamental, o que dificulta sua utilização nas séries iniciais do Ensino Fundamental, devido à sua usabilidade e interface que abrangem conteúdos mais complexos e avançados, o que

pode dificultar o entendimento e a aprendizagem dos alunos mais jovens.

O aplicativo, segundo seus autores, foi desenvolvido para o público de alunos de nonos anos do Ensino Fundamental, como pode-se constatar na Figura 4, o QUIFSC traz questões que exigem conhecimentos prévios de matemática e maior agilidade para responder as questões que são temporizadas. Logo, é essencial que os alunos tenham uma base sólida de conhecimentos para aproveitar as atividades propostas pela plataforma.

Figura 4. QUIFSC

Por fim, o *MathColor*, aplicativo proposto neste trabalho, tem como objetivo ser lúdico e de fácil entendimento para que as crianças possam resolver questões do terceiro ano do Ensino Fundamental. O *software* proposto conta com cinco níveis de dificuldade e recompensará os usuários com uma variedade de desenhos para colorir.

No Quadro 1, são comparadas várias características do *MathColor*, *software* proposto neste trabalho, com os aplicativos e plataformas já mencionados nesta seção. Essa análise abrange os seguintes pontos:

- **Usabilidade:** Usabilidade dos aplicativos, averiguando se ele é fácil e intuitivo de usar;
- **Versão Gratuita:** Disponibilidade de uma versão gratuita, para que os usuários possam acessar algumas funcionalidades básicas sem custo;
- **Versão Paga:** Existência de uma versão paga, que possa oferecer recursos avançados;
- **Infantil:** Adequação para uso infantil, considerando sua interface e apresentação dos conteúdos matemáticos;
- **Níveis de dificuldade:** Presença de níveis de dificuldade, permitindo que os usuários avancem o nível de habilidade;
- **Recompensas:** Existência de recompensas, como pontos ou desbloqueio de conteúdos adicionais, para motivar os usuários a utilizar o *software* e melhorar suas habilidades matemáticas.

Essa comparação detalhada permite entender como o *MathColor* se destaca em relação aos demais aplicativos e plataformas, bem como seu potencial para atender às necessidades dos usuários.

Quadro 1: Relação comparativa entre os aplicativos.

Características	*Math* Bingo	*Math Kids*	*Division Derby*	QUIFSC	*MathColor*
Usabilidade	Sim	Sim	Sim	Sim	Sim
Versão Gratuita	Sim	Sim	Sim	Sim	Sim
Versão Paga	Sim	Não	Não	Não	Não
Infantil	Sim	Sim	Sim	Não	Sim
Níveis de dificuldade	Sim	Não	Sim	Sim	Sim
Recompensas	Sim	Sim	Não	Não	Sim

2.5. Tecnologias para desenvolvimento *mobile*

O desenvolvimento de aplicativos móveis é uma área específica do desenvolvimento de *software* voltada para a criação de soluções para dispositivos móveis como *smartphones* e *tablets*. De acordo com Lecheta (2016), esta área requer o uso de tecnologias especializadas, incluindo linguagens de programação, *frameworks* e plataformas, que possibilitem o desenvolvimento de aplicativos para diferentes sistemas operacionais.

Durante o processo de criação, é levado em consideração aspectos como usabilidade, performance, segurança e integração com outros sistemas e dispositivos. Por isso, a seleção das tecnologias a serem empregadas é essencial para garantir a qualidade do aplicativo desenvolvido. Nesta seção, serão apresentadas as principais ferramentas que foram utilizadas durante o desenvolvimento do projeto, bem como uma definição para cada uma delas e os motivos que justificaram sua escolha para a realização do trabalho proposto. Na Figura 5, são apresentadas as principais tecnologias que foram utilizadas no desenvolvimento do *MathColor*.

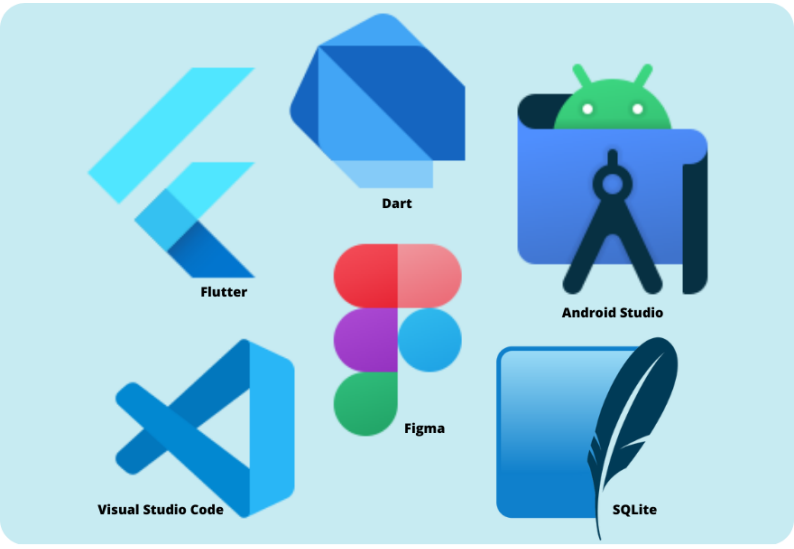

Figura 5. Tecnologias utilizadas.

Para o desenvolvimento de um aplicativo, é fundamental escolher a linguagem de programação adequada e utilizar ferramentas de programação, como ambientes de desenvolvimento integrado (IDEs) e editores de código, para criar códigos organizados e de fácil manutenção. Além disso, é importante a utilização de um *software* de prototipação, que permite criar uma visualização prévia das telas e funcionalidades do *software*. Um banco de dados também é necessário para armazenar informações e dados essenciais ao funcionamento do aplicativo. A integração harmoniosa dessas ferramentas resultará em um aplicativo funcional e de alta qualidade.

A plataforma *Figma*, lançada em 2016 pela empresa Figma, Inc. e fundada por Dylan Field e Evan Wallace, conforme descrito no próprio *site*, Figma (2016) possui objetivo de criar uma ferramenta colaborativa e acessível. Esta plataforma que permite aos *designers* e outros profissionais a construção de produtos digitais completos, desde *sites* até aplicativos para dispositivos móveis e até mesmo pequenas telas, como temporizadores de *microondas*.

Este recurso é utilizado como ferramenta de prototipagem devido à sua capacidade de colaboração entre dois ou mais profissionais, tornando possível o trabalho em conjunto na construção de *designs* de interface e fluxos completos. Dessa forma, são exploradas diversas possibilidades no *design* de interface da aplicação proposta, de forma mais eficiente e produtiva, gerando resultados mais rápidos e de qualidade.

A respeito do editor de código, o *Visual Studio Code* é um editor de código-fonte leve e poderoso, disponível para *Windows*, *macOS* e *Linux*. De acordo com sua documentação disponível no *site* Code (2015), ele pode ser integrado com *JavaScript*, *TypeScript* e *Node.js*, sendo uma ótima opção para desenvolvedores que trabalham em diferentes plataformas. Além disso, possui um ecossistema rico de extensões para outras linguagens e tempos de execução, como *C++, C#*, *Java*, *Python, PHP, Go* e *.NET*, tornando-se uma ferramenta versátil e poderosa para desenvolvimento.

O *Visual Studio Code* foi selecionado para o projeto devido à sua capacidade de integrar-se facilmente com outras tecnologias, desta forma irá se adequar sem complicações ao *framework* utilizado neste projeto. Além disso, possui uma interface intuitiva com recursos avançados, oferecendo uma experiência de programação simplificada.

Em relação ao *framework*, o *Flutter* é uma ferramenta de desenvolvimento multiplataforma para dispositivos móveis, criada pela *Google*. Segundo a própria documentação do Flutter (2017), sua principal vantagem é permitir que aplicativos sejam desenvolvidos para *Android e iOS* ao mesmo tempo, com apenas um código, o que reduz a curva de aprendizado e agiliza o processo de desenvolvimento.

A escolha desta ferramenta, deu-se devido a possibilidade de desenvolver aplicativos para diferentes plataformas ao mesmo tempo com um único código. Outro aspecto relevante é que esta ferramenta dispõe de uma documentação completa e uma comunidade ativa, o que pode facilitar a resolução de problemas e a busca por novas soluções.

Em sequência, a linguagem de programação que compõe o aplicativo é o *Dart*, uma linguagem de programação de alto desempenho, desenvolvida pela *Google* em 2011, que de acordo com a documentação disponível no *site* do Dart (2011), tem como objetivo ser a linguagem mais produtiva para o desenvolvimento multiplataforma. Seu *design* per-

mite que desenvolvedores criem aplicativos rápidos em qualquer plataforma, combinando com uma plataforma de tempo de execução flexível para estruturas de aplicativos.

Com a crescente demanda por aplicativos multiplataforma, o *Dart* foi escolhido como ferramenta para o projeto devido à sua eficiência e facilidade de uso. Além disso, é a linguagem principal usada pelo *Flutter*, o que torna a escolha da linguagem ainda mais adequada para o projeto. Ademais, possui uma sintaxe simples, facilitando a sua aprendizagem, e possui uma comunidade ativa e documentação abrangente, tornando-a uma excelente opção para desenvolvimento de aplicativos.

Sobre o banco de dados adotado, o *SQLite* é um mecanismo de banco de dados *SQL*, que conforme documentado em SQLite (2000), surgiu no ano 2000 como um projeto de código aberto criado por D. Richard Hipp. Desde então, tornou-se o banco de dados mais implantado no mundo e é usado em uma infinidade de projetos, devido às suas inúmeras vantagens.

O principal motivo para a adição do *SQLite* no *MathColor*, foi sua portabilidade. Por ser um banco de dados que pode incorporar-se diretamente ao projeto, ele não precisa de um servidor a parte e funciona diretamente em arquivos de disco comuns. Isso significa que é um banco de dados completo, com múltiplas tabelas, índices, gatilhos e exibições, podendo ser armazenado em um único arquivo de disco. Ademais, o formato do arquivo é multiplataforma, sendo capaz de ser copiado para diferentes sistemas operacionais e arquiteturas de processador.

Por fim, o *Android Studio* também foi utilizado na construção da aplicação, sendo um ambiente de desenvolvimento integrado (IDE) oficial para a criação de aplicativos *Android*. De acordo com Studio (2013), essa ferramenta é baseada no editor de código e nas ferramentas para desenvolvedores avançados do *IntelliJ IDEA*, que oferece uma grande variedade de recursos para aumentar a produtividade dos desenvolvedores.

O *Android Studio* foi escolhido como ferramenta para o projeto pois oferece todas as ferramentas necessárias para o desenvolvimento e teste de aplicativos *Android*. No projeto atual, esta tecnologia foi utilizada apenas para emular o aplicativo e garantir que ele funcione corretamente.

3. Desenvolvimento

O desenvolvimento deste trabalho se sucedeu através de uma pesquisa detalhada, onde foram aplicados os princípios e diretrizes da etapa de desenvolvimento de um artigo científico. De acordo com Pereira et al. (2018), nesta fase é possível apresentar e explorar a metodologia adotada no desenvolvimento do trabalho, com clareza e coerência na exposição das informações sobre a evolução da pesquisa.

Sendo assim, esta seção será dividida em 2 subseções para abordar diferentes aspectos do desenvolvimento da plataforma. Na subseção 3.1, será descrito a modelagem do sistema. Já na subseção 3.2, será realizada a descrição da implementação do sistema com base nos requisitos levantados na subseção 3.1. Isso envolve a codificação do *software* e do banco de dados, ou seja, a escrita do código fonte que implementa as funcionalidades definidas anteriormente.

3.1. Modelagem

A modelagem desempenha um papel fundamental no desenvolvimento de um jogo educativo de qualidade. Nesta subseção, serão abordados os principais aspectos relacionados à modelagem do jogo, incluindo o levantamento de requisitos, a prototipação do aplicativo, o diagrama de casos de uso e o projeto de banco de dados. Esses elementos se interligam para estabelecer uma base sólida que garantirá a consistência e integração dos elementos presentes nas telas que serão identificadas no levantamento de requisitos.

Segundo Pressman (1995), os modelos auxiliam o desenvolvedor a compreender as informações, funções e comportamento do sistema, tornando a análise de requisitos mais sistemática e fácil. Também pode servir como base para o *design*, fornecendo uma representação essencial do *software*, que pode ser adaptada para a implementação.

Além de proporcionar uma visão clara da estrutura interna do jogo, a modelagem também facilitou a identificação de possíveis melhorias e otimizações ao longo do processo de desenvolvimento. No entanto, para gerenciar efetivamente todas as etapas e garantir o sucesso do projeto, foi adotada uma abordagem adaptada da metodologia *Scrum*. De acordo com o próprio site da organização, o Scrum (2011) é um *framework* ágil utilizado no desenvolvimento de projetos que promove a colaboração, transparência e entrega incremental de valor, baseando-se em *sprints*, reuniões periódicas e divisão de tarefas para garantir eficiência e adaptabilidade no processo de desenvolvimento.

A adaptação realizada levou em consideração as necessidades específicas do projeto, incluindo a ausência de reuniões diárias e de um papel formal de *Scrum Master*. Sendo assim, através dessa abordagem, foi possível planejar, executar e acompanhar de maneira eficiente todas as atividades relacionadas à modelagem do jogo. Embora não houvessem reuniões diárias formais, foram realizados encontros periódicos para garantir o alinhamento da equipe e a revisão do progresso. Conforme a Figura 6, no *backlog* as tarefas foram divididas em *sprints* de curta duração, permitindo uma abordagem iterativa e adaptativa ao desenvolvimento do jogo.

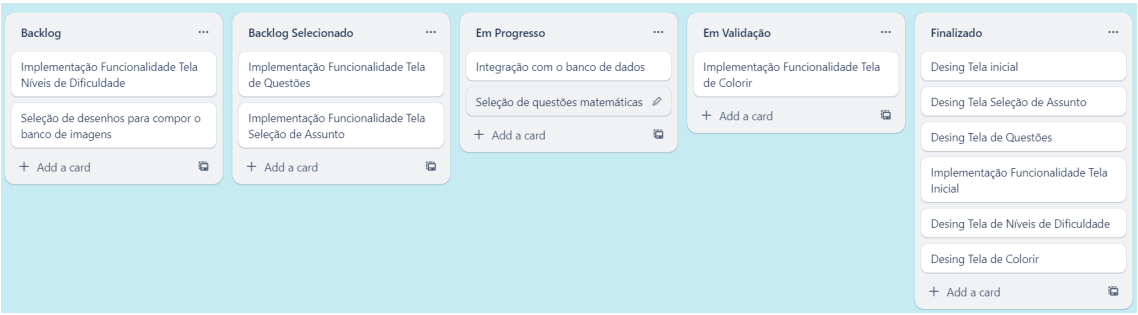

Figura 6. *Backlog* de tarefas.

Apesar de não existir um *Scrum Master* designado, foi mantido entre os integrantes a colaboração, a transparência e a entrega incremental de valor. A responsabilidade de facilitar a comunicação e resolver eventuais impedimentos foi compartilhada entre os membros da equipe e essa abordagem permitiu que fossem realizados ajustes contínuos e o alinhamento constante com as expectativas dos usuários, resultando em um produto final de alta qualidade.

Dessa forma, a combinação da modelagem abrangente com essa adaptação da metodologia *Scrum* proporcionou uma base sólida e uma abordagem ágil para o desenvolvimento do jogo educativo, garantindo que as necessidades e expectativas dos jogadores fossem atendidas de maneira eficaz.

Para realizar o levantamento de requisitos do aplicativo, considerou-se o conhecimento adquirido por meio de estudos em artigos, pesquisas sobre jogabilidade infantil e análise de sistemas similares, obtidos através do referencial teórico. Essa abordagem combinada permitiu identificar algumas práticas, tendências e funcionalidades desejáveis para a plataforma do jogo em questão. Dessa forma, foi possível estabelecer uma base sólida de requisitos que atenderão às necessidades e expectativas dos jogadores, com o objetivo de proporcionar uma experiência educativa e divertida. A seguir, serão descritas as funcionalidades identificadas:

1. **Tela Inicial:** A tela inicial será o ponto de partida para os jogadores. Ela é visualmente atraente e intuitiva, proporcionando uma primeira impressão positiva do jogo. Foi mantido um *design* infantil com um único botão para iniciar o jogo.
2. **Tela de Seleção de Assunto:** A tela de seleção de assunto permitiu que os jogadores escolham o assunto específico que desejam jogar. Foi apresentada uma lista clara e organizada dos diferentes assuntos disponíveis, possibilitando uma escolha fácil e rápida.
3. **Tela de Níveis de Dificuldade:** Após escolher o tópico de interesse, o usuário terá acesso a uma tela exibindo cinco diferentes níveis de dificuldade disponíveis, que o usuário precisará superar para alcançar seu objetivo e obter sua recompensa ao final.
4. **Tela de Questões:** Nesta tela, os jogadores terão a oportunidade de resolver questões matemáticas desafiadoras. Essas questões foram apresentadas de maneira clara e objetiva, com o intuito de estimular o raciocínio e aprimorar as habilidades de resolução de problemas. À medida que os jogadores solucionam cada questão, o nível de dificuldade progressivamente aumenta, proporcionando um desafio contínuo e gratificante.
5. **Tela de Colorir:** Depois de concluir todas as questões, como recompensa, os alunos terão a oportunidade de colorir um desenho aleatório de acordo com sua criatividade e preferência. Isso permitirá explorar artisticamente suas conquistas de maneira única e personalizada.

Essas funcionalidades foram destacadas como elementos essenciais na concepção de uma plataforma educativa e envolvente, destinada a promover a aprendizagem da matemática de maneira lúdica. Conforme ilustrado na Figura 7, a plataforma oferece uma tela inicial atrativa, uma seleção de assuntos intuitiva, uma interface para visualização dos níveis de dificuldade, desafios matemáticos apresentados de forma clara em uma tela específica de questões, e finalmente, uma seção de imagens para colorir. O objetivo é proporcionar aos estudantes uma experiência enriquecedora e gratificante, estimulando tanto o desenvolvimento educacional quanto a expressão criativa.

Para entender melhor como essas funcionalidades se encaixam na vivência dos jogadores, é importante explorar os casos de uso que impulsionam sua utilidade e impacto educacional. Os cenários criados fornecem um panorama abrangente das interações entre os jogadores e a aplicação, permitindo capturar e descrever os requisitos funcionais do sistema do ponto de vista do usuário final.

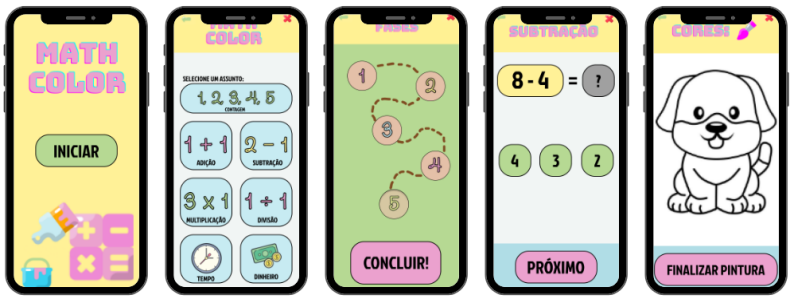

Figura 7. Prototipação do *MathColor*.

Conforme a Figura 8, essas narrativas detalhadas oferecem uma visão clara de como os estudantes utilizarão o *software*, quais funcionalidades serão mais relevantes para eles e como o aprendizado de matemática será promovido de maneira eficaz. De acordo com Booch et al. (2006), os casos de uso ajudam a direcionar o desenvolvimento do aplicativo, garantindo que a aplicação atenda às expectativas dos jogadores e cumpra seu propósito educacional de forma impactante.

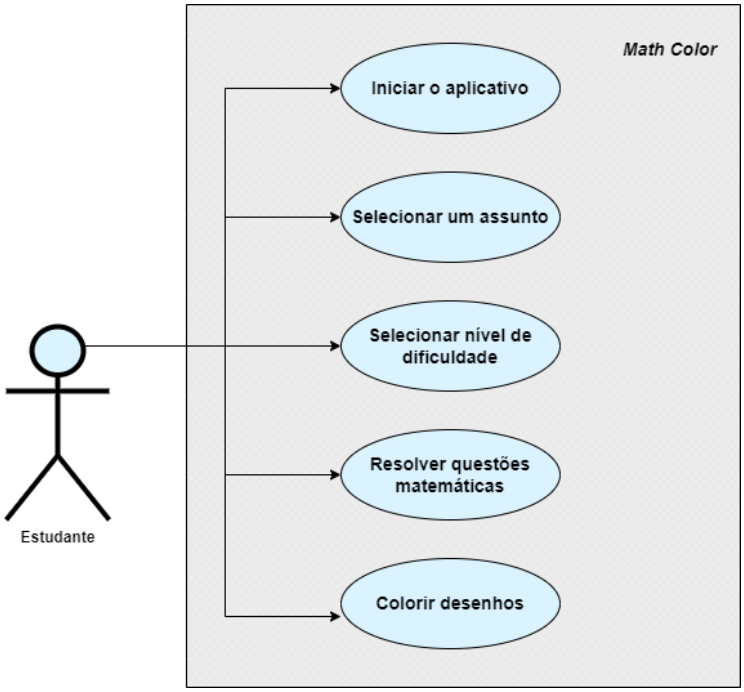

Figura 8. Diagrama de caso de uso do *MathColor*.

No contexto deste projeto, foi desenvolvido uma aplicação voltada para crianças, levando em conta suas necessidades e habilidades. Desta forma, é importante considerar que a simplicidade e a facilidade de uso devem ser priorizadas. Além de compreender os casos de uso e requisitos funcionais do sistema, é igualmente importante considerar a estrutura de armazenamento de dados necessária para suportar a aplicação.

Um banco de dados é uma tecnologia que permite armazenar, gerenciar e acessar informações de forma estruturada. Ele atua como um repositório centralizado para os dados relacionados a uma determinada aplicação ou sistema, neste caso, o *MathColor*. Os dados são organizados em tabelas, com cada tabela representando uma entidade ou

conceito específico. Cada tabela é composta por colunas que definem os tipos de dados armazenados, e as linhas contêm os registros individuais. De acordo com Oliveira (2007) em muitos sistemas computacionais, o banco de dados é o "coração" da aplicação, e o seu bom funcionamento é imprescindível para o sucesso de todo o sistema.

Sendo assim, ao utilizar o *SQLite* como sistema de armazenamento de dados, são registradas informações sobre as imagens para colorir e questões do jogo, incluindo opções de respostas e respostas corretas. Conforme apresentado na Figura 9, a modelagem do banco de dados consiste na criação de 8 tabelas para armazenar esses dados, garantindo a integridade por meio de chaves primárias. Por ser uma biblioteca leve e de fácil integração, torna-se uma opção adequada para o *MathColor*, pois não demanda uma infraestrutura de banco de dados mais robusta.

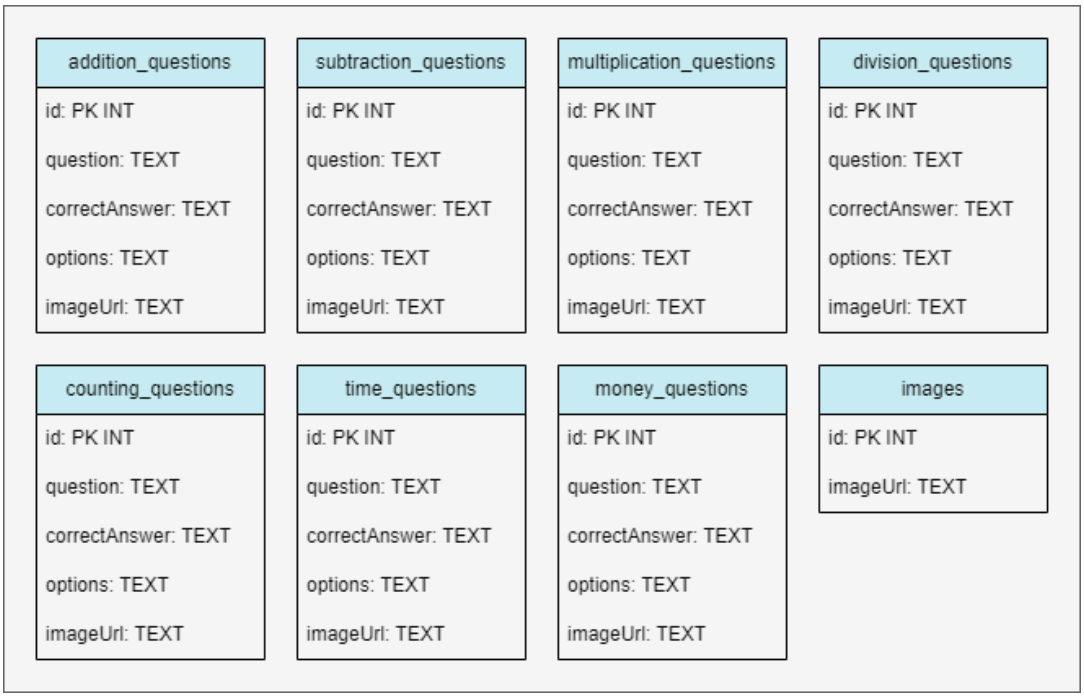

Figura 9. Modelagem do banco de dados.

É importante destacar que, para este projeto, não foi necessário estabelecer comunicação com um servidor externo. Isso se deve ao fato de que o aplicativo executa funções de busca e acesso aos dados diretamente no banco de dados local. Essa abordagem dispensa a necessidade de requisições complexas a um servidor externo, simplificando a arquitetura do aplicativo e mantendo-o eficiente em termos de recursos.

3.2. Implementação

Segundo Filho (2011), a implementação é a fase do ciclo de vida do projeto em que o *software* é desenvolvido e codificado, representando a materialização concreta das ideias propostas. Desta forma, o código é criado para transformar o projeto em realidade, e nessa fase, ocorre a construção das funcionalidades planejadas, a criação das interfaces de usuário e a integração dos componentes. Nesta subseção, são discutidos os detalhes da implementação do aplicativo, abrangendo as etapas cruciais do desenvolvimento que deram forma ao *software* final.

3.2.1. Banco de Dados

A implementação do banco de dados neste projeto foi realizada com o auxílio da biblioteca *sqflite*. Essa biblioteca permite que o aplicativo crie e gerencie um banco de dados *SQLite*, o que simplifica o trabalho com bancos de dados em aplicativos *Flutter*.

Conforme ilustrado na Figura 10, o *sqflite* oferece uma maneira fácil e eficiente de criar tabelas específicas para armazenar perguntas matemáticas, opções de resposta e *URLs* de imagens associadas. As tabelas são criadas toda vez que a base de dados é inicializada através da função *initializeDatabase*, garantindo que a estrutura esteja sempre atualizada quando o jogo inicia.

DataBaseHelper

```
class DatabaseHelper {
  static Database? _database;

  Future<Database> get database async {
    if (_database != null) return _database!;

    _database = await initializeDatabase();
    return _database!;
  }

  Future<Database> initializeDatabase() async {
    final path = await getDatabasesPath();
    final dbPath = join(path, 'questions.db');

    return await openDatabase(
      dbPath,
      version: 1,
      onCreate: (db, version) async {

        await db.execute('''
      CREATE TABLE addition_questions(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        question TEXT,
        correctAnswer TEXT,
        options TEXT,
        imageUrl TEXT
      )
    ''');

        // Repetição para as demais tabelas:
          // subtraction_questions, multiplication_questions, division_questions
          // counting_questions, time_questions, money_questions, images
      },
    );
  }
```

Figura 10. Criação das tabelas.

Para facilitar a inserção de dados nas tabelas do aplicativo, foram desenvolvidas as funções *insertQuestions* e *insertImages*, apresentadas na Figura 11. A função *insertQuestions* trabalha com uma lista chamada *questions*, que contém mapas representando a questão com a pergunta, resposta correta, opções de resposta e *URLs* de imagens associadas. Um *loop* percorre essa lista e, a partir de cada mapa, extrai as informações para inserção no banco de dados, por meio do método *insert*. Da mesma forma, o *insertImages* realiza o mesmo processo, com a lista *imageUrls*, que contém o caminho de cada imagem.

DataBaseHelper

```
Future<void> insertQuestions(
     List<Map<String, dynamic>> questions, String tableName) async {
    final db = await database;
    final batch = db.batch();

    for (var question in questions) {
      final optionsJson = jsonEncode(question['options']);
      batch.insert(tableName, {
        'question': question['question'],
        'correctAnswer': question['correctAnswer'],
        'imageUrl': question['imageUrl'],
        'options': optionsJson,
      });
    }

    await batch.commit(noResult: true);
  }

  Future<void> insertImages(List<String> imageUrls) async {
    final db = await database;
    final batch = db.batch();

    for (var imageUrl in imageUrls) {
      batch.insert('images', {'imageUrl': imageUrl});
    }

    await batch.commit(noResult: true);
  }
```

Figura 11. Inserção dos dados.

Por fim, conforme apresentado na Figura 12, para recuperar as informações armazenadas, foram criados os métodos *getImageUrls*, que realiza uma consulta através do método *query* à uma tabela específica para obter as *URLs* das imagens associadas, e *getQuestions*, que também utiliza o método *query* para recuperar as questões juntamente com todas as suas propriedades. Os resultados dessas consultas são armazenados em uma lista de mapas. Em seguida, o código gera uma lista de objetos, onde cada objeto é criado a partir dos dados recuperados, garantindo que tanto as imagens quanto as questões e suas dependências sejam carregadas de forma eficiente e precisa.

DataBaseHelper

```
Future<List<Question>> getQuestions(String tableName) async {
    final db = await database;
    final List<Map<String, dynamic>> maps = await db.query(tableName);

    return List.generate(maps.length, (i) {
      return Question(
        id: maps[i]['id'],
        question: maps[i]['question'],
        imageUrl: maps[i]['imageUrl'],
        correctAnswer: maps[i]['correctAnswer'],
        options: List<String>.from(jsonDecode(maps[i]['options'])),
      );
    });
  }

  Future<List<String>> getImageUrls() async {
    final db = await database;
    final List<Map<String, dynamic>> maps = await db.query('images');
    return List.generate(maps.length, (i) {
      return maps[i]['imageUrl'];
    });
  }
```

Figura 12. Recuperação dos dados.

Desta forma, com a estrutura criada, o *software* se torna escalável, uma vez que simplifica a adição e a recuperação de informações sem a necessidade de modificações complexas no código. Essa flexibilidade permite que o aplicativo cresça e se adapte às mudanças, garantindo sua capacidade de evoluir de acordo com as demandas e necessidades futuras, bastando utilizar os métodos já criados. Isso simplifica significativamente o processo de desenvolvimento e manutenção do aplicativo.

3.2.2. Aplicação

A tela inicial do *MathColor*[1] exerce um papel importante ao estabelecer o estilo do jogo. Segundo Braga (2010), a chave para o sucesso de um aplicativo reside na habilidade de adequar um objeto funcional ao seu público por meio da estética. Sendo assim, a intenção desta tela é ser intuitiva e simples, apresentando apenas imagens coloridas e um botão de início. Esse *design* visa transmitir claramente a proposta do aplicativo, convidando os usuários a explorar o jogo de maneira direta e envolvente desde o primeiro momento.

Para a escolha do tema do jogo, foi implementada uma tela de seleção de temas matemáticos, permitindo aos usuários escolherem o foco de interesse para o jogo. Cada tópico é representado por um ícone visual distinto, proporcionando uma forma de seleção. Conforme a Figura 13, esse processo é gerenciado pela classe *SubjectRepository*,

[1]Link para *download* da aplicação: `https://drive.google.com/file/d/1z1Fh5Xyf4Kshk2BKZRVFAyK_54QBKt5T/view?usp=drive_link`, responsivo em *tablets* com sistema Android

responsável por administrar os temas disponíveis.

SubjectRepository

```
class SubjectRepository {
  static final SubjectRepository _instance = SubjectRepository._internal();

  factory SubjectRepository() {
    return _instance;
  }

  SubjectRepository._internal();

  String _currentSubject = '';

  String get currentSubject => _currentSubject;

  void updateSubject(String subject) {
    _currentSubject = subject;
  }

  void resetSubject() {
    _currentSubject = '';
  }
}
```

Figura 13. Implementação da classe *SubjectRepository*.

Em sequência, na Figura 14, pode-se analisar o código que resulta na tela de seleção de temas e também a tela. A estrutura da tela inicia com uma barra superior que contém o nome do aplicativo, acompanhada por botões de navegação. Logo abaixo, a interface concentra-se nos ícones temáticos. Cada ícone é envelopado por um botão funcional, o qual, ao ser acionado, atualiza o tópico escolhido no repositório e encaminha o usuário para a tela seguinte.

Por meio de representações visuais, essa tela proporciona aos usuários a oportunidade de elegerem o tópico desejado. Cada ícone caracteriza um dos temas abordados: “contagem”, “adição”, “subtração”, “multiplicação”, “divisão”, “tempo”e “dinheiro”.

Essa escolha com ícones visualmente distintos e representativos ajuda a tornar a experiência do usuário mais intuitiva. Mesmo pessoas com pouca experiência tecnológica, podem facilmente identificar e selecionar o tópico de seu interesse. A abordagem acessível e amigável do *design* torna o uso do aplicativo mais inclusivo, atendendo a um público diversificado, o que é essencial para alcançar um amplo alcance e impacto educacional.

Além disso, a funcionalidade dos botões associados a esses ícones permite uma navegação suave e direta para a próxima tela, tornando o processo mais simples para os usuários.

SelectSubjectScreen

```
Padding(
        // Botões de seleção de assunto
        padding: EdgeInsets.only(bottom: screenHeight * 0.03),
        child: TextButton.icon(
          onPressed: () {
            subjectRepository.updateSubject('contagem');
            Navigator.push(
              context,
              MaterialPageRoute(builder: (context) => MathQuizScreen()),
            );
          },
          style: ButtonStyle(
            backgroundColor:
                MaterialStateProperty.all<Color>(Color(0xFFC7EBF2)),
                foregroundColor: MaterialStateProperty.all<Color>(Colors.black),
                side: MaterialStateProperty.all<BorderSide>(
                BorderSide(color: Colors.black, width: 2),
            ),
            shape: MaterialStateProperty.all<RoundedRectangleBorder>(
                        RoundedRectangleBorder( borderRadius:
                        BorderRadius.circular(screenHeight * 0.04),),
            ),
          ),
          icon: Image.asset(
            path.join('lib', 'assets', 'imagens', 'Contagem.png'),
            width: screenHeight * 0.5,
            height: screenWidth * 0.1,
          ),
          label: Text('',
            style: TextStyle(
              color: Color(0xFFFB6D993),
              fontWeight: FontWeight.bold,
              fontFamily: 'craft',
              fontSize: screenHeight * 0.01,
              decoration: TextDecoration.overline,
            ),
          ),
        ),
      ), // Repetições semelhantes para os outros tópicos
```

Figura 14. Implementação da classe *SelectSubjectScreen* e tela resultante.

Após a seleção do tema desejado, o jogador é automaticamente redirecionado para a tela que apresenta os níveis de dificuldade relacionados ao tópico escolhido. Conforme demonstrado na Figura 15, a codificação estrutura as questões em cinco fases distintas, sendo que à medida que o jogador avança de uma fase para outra, o nível de complexidade das questões aumenta gradualmente. No canto direito desta mesma Figura, também pode-se visualizar a tela resultante desta implementação.

LevelsScreen

```dart
class LevelsScreen extends StatelessWidget {
  const LevelsScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    // Declaração de variáveis utilizadas

    Widget buildLevelButton(int level) {
      return ElevatedButton(
        onPressed: currentLevel == level ? () {
          Navigator.push(context,
                  MaterialPageRoute(builder: (context) => MathQuizScreen()),);
              } : null,
        style: ElevatedButton.styleFrom( // Estilo do botão ),
      );
    }

    return Scaffold(
      backgroundColor: Color(0xFFB6D993),
      appBar: AppBar( // Implementação do desing do appBar ),
      body: Column( // Desing textos da tela ),
          SizedBox(height: 30),
          // Cria botões de nível dinamicamente
          for (int i = 1; i <= 5; i++)
            Column(
              children: [
                buildLevelButton(i), SizedBox(height: screenHeight * 0.035), ],
            ),
          ElevatedButton(
            onPressed: currentLevel == 6 ? () {
                    Navigator.push(context,
                      MaterialPageRoute( builder: (context) => ColorizeImageScreen()),);
                  } : null,
            style: ElevatedButton.styleFrom( // Estilo do botão ),
            child: Text(
              'CONCLUIR',
              style: TextStyle(fontSize: screenHeight * 0.035),
            ),
          ),
        ],
      ),
    );
  }
}
```

Figura 15. Implementação da classe *LevelsScreen* e tela resultante.

A responsabilidade de gerenciar o nível de dificuldade em que o usuário se encontra recai sobre a classe *LevelsRepository*, conforme a Figura 16. A interação do jogador com as diferentes fases do jogo é realizada através da variável *currentLevel*, que é atualizada e restaurada pelas funções *completeLevel* e *resetCurrentLevel*.

LevelsRepository

```
class LevelsRepository {
  static final LevelsRepository _instance = LevelsRepository._internal();

  factory LevelsRepository() {
    return _instance;
  }

  LevelsRepository._internal();

  int _currentLevel = 1;

  int get currentLevel => _currentLevel;

  void completeLevel() {
    if (_currentLevel < 6) {
      _currentLevel++;
    }
  }

  void resetCurrentLevel() {
    _currentLevel = 1;
  }
}
```

Figura 16. Implementação classe *LevelsRepository*.

A cada nível de dificuldade, o jogador é conduzido a uma tela de *quiz*. Nesta etapa, a interface foi planejada para avaliar o conhecimento e as habilidades do estudante em relação ao tópico escolhido. Nesse contexto, uma pergunta é exibida. A complexidade das questões se adapta de acordo com o nível de dificuldade, assegurando que o estudante experimente uma progressão gradual no grau de desafio à medida que prossegue. Essa progressão desempenha um papel fundamental, não somente para manter o interesse do usuário, mas também para incentivá-lo a aprender e superar obstáculos.

Na Figura 17, é apresentada a tela obtida a partir do código demonstrado, que não se limita a apresentar a pergunta, mas também disponibiliza três opções de resposta. Essas alternativas de resposta são elaboradas para cada pergunta e nível de dificuldade, proporcionando ao jogador escolhas que exigem atenção.

Adicionalmente, essa tela é equipada com um botão que possibilita avançar para a próxima pergunta ou desafio. Esse botão age como uma ponte entre as perguntas e o progresso do aluno, simplificando a navegação. Além disso, a interface é enriquecida com efeitos sonoros que ampliam a imersão e envolvimento da criança. Esses sons podem variar desde *feedback* positivo, quando uma resposta correta é selecionada, até alertas de erro, no caso de uma resposta incorreta.

MathQuizScreen

```dart
class MathQuizScreen extends StatefulWidget {
  @override
  _MathQuizScreenState createState() => _MathQuizScreenState();
}

class _MathQuizScreenState extends State<MathQuizScreen> {
  // Declaração de variáveis utilizadas
  @override
  Widget build(BuildContext context) {
    // Declaração de variáveis utilizadas

    return Scaffold(
      backgroundColor: Color(0xFFFFFFFFF),
      appBar: buildAppBar(),
      body: Column(
        children: [
          buildQuestionHeader(screenHeight),
          SizedBox(height: 50),
          Container(
            padding: EdgeInsets.symmetric(horizontal: 20),
            child: Column(
              children: [
                buildQuestionRow(
                    screenHeight, screenWidth, currentLevel, subject),
                buildImageIfNeeded(screenHeight, screenWidth),
              ],
            ),
          ),
          Spacer(),
          buildOptionsRow(
            screenHeight, screenWidth, currentLevel, subject
          ),
          buildNextButton(screenHeight, screenWidth),
        ],
      ),
    );
  }
}
```

Figura 17. Implementação da classe *MathQuizScreen* e tela resultante.

Para garantir a clareza na apresentação das opções de resposta, a cor que destaca a pergunta, juntamente com a resposta selecionada, auxilia na diferenciação da escolha do usuário. Essa abordagem visual pode tornar o processo de seleção de respostas mais instintivo e compreensível.

Depois de concluir os desafios matemáticos sobre o assunto escolhido, como exibido na Figura 18, o estudante recebe como recompensa uma imagem aleatória para colorir. Essa atividade de colorir não apenas proporciona uma pausa lúdica e relaxante após o exercício mental, mas também permite que os estudantes expressem sua individualidade. A escolha das cores utilizadas fica a critério do aluno, incentivando a imaginação e oferecendo uma experiência personalizada. Essa abordagem recompensa o esforço do estudante e o envolve em uma atividade artística.

ColorizeImageScreen

```
class _ColorizeImageScreenState extends State<ColorizeImageScreen> {

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Color(0xFFFFF197),
        elevation: 0,
        automaticallyImplyLeading: false,
        actions: [
          IconButton(
            icon: Icon(Icons.close),
            onPressed: resetAndNavigateToSubjectScreen,
            color: Color(0xFFFF0000),
          ),
        ],
      ),
      body: Column(
        children: [
          buildColorSelectionButton(),
          Expanded(
            child: Center(
              child: buildFloodFillImage(),
            ),
          ),
          buildFinishPaintButton(screenHeight, screenWidth),
        ],
      ),
    );
  }

  Widget buildFloodFillImage() {
    return Center(
      child: randomImageUrl.isNotEmpty
          ? FloodFillImage(
              imageProvider:
                  AssetImage(
                    'lib/assets/imagens/colorImages/$randomImageUrl'
                  ),
              fillColor: _selectedColor,
              avoidColor: [Colors.transparent, Colors.black],
              width: MediaQuery.of(context).size.width.toInt(),
              height: MediaQuery.of(context).size.height.toInt(),
            )
          : CircularProgressIndicator(),
    );
  }
```

Figura 18. Implementação da classe *ColorizeImageScreen* e tela resultante.

Para automatizar o preenchimento de uma imagem com uma cor selecionada pelo usuário, utilizou-se a biblioteca *Flood Fill*. Essa biblioteca contém um algoritmo originalmente desenvolvido por Dunlap (2006) e adaptado para o ambiente *Flutter* pelo desenvolvedor Javier (2021). Este algoritmo é uma técnica amplamente empregada em aplicações de edição de imagens e jogos que envolvem a correspondência de elementos semelhantes

para criar reações em cadeia. Ele permite preencher uma área contígua em uma matriz multidimensional com uma cor específica.

Conforme a Figura 19, o funcionamento desta biblioteca começa com um ponto de partida na imagem e examina o *pixel* nesse ponto. Se esse *pixel* tiver a mesma cor ou estiver dentro de uma tolerância configurável em relação à cor desejada, ele é preenchido com a nova cor e marcado como visitado. Em seguida, o algoritmo verifica os *pixels* vizinhos, ou seja, acima, abaixo, à esquerda e à direita do *pixel* atual, e repete o processo para cada um deles. Esse processo continua recursivamente até que todos os *pixels* conectados à região inicial tenham sido preenchidos com a nova cor ou até que a tolerância de cor seja excedida. Como resultado, o *Flood Fill* consegue preencher áreas complexas em uma imagem de forma eficiente.

QueueLinearFloodFiller

```
class QueueLinearFloodFiller {
  // Declaração das variáveis

  QueueLinearFloodFiller(img.Image imgVal, int newColor) { // Construtor da classe}

  // Função principal para preencher uma área na imagem a partir das coordenadas (x, y)
  Future<void> floodFill(int x, int y) async {
    _prepare();
    if (_startColor == null) {
      // Obtém a cor do pixel de referência inicial
      int startPixel = image.getPixelSafe(x, y);
      _startColor = [ img.getRed(startPixel), img.getGreen(startPixel),
                    img.getBlue(startPixel), img.getAlpha(startPixel),];
    }
    _linearFill(x, y);  // Preenche horizontalmente a partir de (x, y)
    _FloodFillRange range;
    while (_ranges.isNotEmpty) {
      range = _ranges.removeFirst();
      int downPxIdx = (_width * (range.y + 1)) + range.startX;
      int upPxIdx = (_width * (range.y - 1)) + range.startX;
      int upY = range.y - 1;
      int downY = range.y + 1;
      for (int i = range.startX; i <= range.endX; i++) {
        if (range.y > 0 && (!_pixelsChecked[upPxIdx]) && _checkPixel(i, upY)) {
          _linearFill(i, upY);  // Preenche acima
        }
        if (range.y < (_height - 1) &&
            (!_pixelsChecked[downPxIdx]) &&
            _checkPixel(i, downY)) { // Verifica se um pixel está dentro da tolerância
                                     // de cores em relação à cor de referência
          _linearFill(i, downY);  // Preenche abaixo
        }
        downPxIdx++;
        upPxIdx++;
      }
    }
  }
}
```

Figura 19. Código resumido da biblioteca *Flood Fill*.

Sendo assim, foi possível implementar [2] o aplicativo proposto no presente artigo, oferecendo uma abordagem inovadora de ensino da matemática para crianças, cumprindo assim, os objetivos apresentados inicialmente.

4. Resultados

Essa seção trata da análise dos resultados coletados a partir de uma avaliação que foi realizada em duas etapas, em uma escola da rede municipal de Lages. Esta avaliação foi aplicada inicialmente com cinco professores pedagogos, que avaliaram o aplicativo por meio de um formulário, podendo adicionar sugestões de melhoria do *software*. Em seguida, conforme a Figura 20, foi realizada uma avaliação final no dia 20 de outubro de 2023, com 20 alunos do terceiro ano do Ensino Fundamental I, onde foram coletados *feedbacks* para análise dos resultados.

Figura 20. Utilização do *MathColor* no momento da avaliação.

Na primeira etapa, os professores pedagogos avaliaram o aplicativo por meio de um formulário com questões que tinham como opções de resposta apenas "sim"e "não". A escolha de opções limitadas foi feita com o propósito de simplificar a avaliação e concentrar-se nos aspectos essenciais do aplicativo, permitindo uma avaliação mais direta

[2]Link do repositório da aplicação no *GitHub*: `https://github.com/anegrizotti/math_color`

e objetiva, sem ambiguidades. Esta análise proporcionou a coleta de *insights* importantes sobre a usabilidade, eficácia e adequação do aplicativo para o público infantil.

A primeira pergunta abordou a usabilidade do aplicativo, questionando se os professores consideravam o aplicativo fácil de usar. O resultado obteve 100% das respostas "sim", ficando evidente que o aplicativo obteve total índice de aprovação. Isso mostra que o *MathColor* é considerado fácil de usar pelos professores pedagogos que o avaliaram. Essa alta taxa de aprovação na usabilidade é um indicativo positivo para a eficácia do *software* no contexto educacional.

A segunda questão informou sobre a adequação do aplicativo para o público infantil, obtendo novamente, 100% de *feedback* positivo por parte dos avaliadores, exibindo sua adequação. Em seguida, a respeito da complexidade das questões apresentadas no *software*, os resultados demonstraram um retorno totalmente positivo, com 100% de aceitação.

Por fim, no que diz respeito à diversão que o jogo proporciona às crianças, também foi obtido total aprovação. Isso comprova que o aplicativo foi capaz de combinar diversão e aspectos educacionais de forma eficaz, garantindo uma experiência agradável e instrutiva para as crianças.

A segunda etapa envolveu a avaliação direta dos alunos, que são os principais beneficiários do aplicativo. Coletar *feedbacks* dos alunos foi importante para entender como o jogo está sendo percebido por seu público-alvo. Essa avaliação final ajudou a identificar aspectos específicos que foram ajustados para melhorar a experiência de aprendizado. Para isso, foram utilizadas folhas impressas com questões formuladas em uma linguagem acessível, facilitando a compreensão pelas crianças. As questões continham quatro opções de resposta: "não", "mais ou menos", "sim"e "muito".

Na primeira pergunta, os alunos foram convidados a expressar se tiveram diversão jogando o aplicativo, como refletido na Figura 21. Os resultados indicam que a maioria dos alunos acharam muito divertido. Isso demonstra que os alunos desfrutaram de uma experiência de diversão ao utilizar o aplicativo.

Já segunda questão, os alunos foram questionados sobre se acharam fácil usar o aplicativo, e os resultados correspondentes podem ser observados na Figura 22. É importante notar que, uma vez que as respostas foram fornecidas por crianças, é possível que tenha ocorrido alguma confusão na compreensão das questões relacionadas à complexidade e usabilidade do aplicativo. No entanto, os resultados ainda refletem uma recepção positiva, com 65% dos alunos considerando o aplicativo "muito"fácil e 30% respondendo "sim". Isso sugere que, apesar de possíveis desafios na interpretação da pergunta, a maioria dos alunos teve uma boa experiência em termos de usabilidade do aplicativo.

No terceiro questionamento, as crianças compartilharam sua percepção sobre a adequação do aplicativo à sua faixa etária. Conforme na Figura 23, a aceitação foi altamente positiva, com 95% das crianças considerando o aplicativo "muito"adequado para sua idade, e 5% respondendo "sim", o que indica uma clara aprovação da faixa etária apropriada para o aplicativo.

Na quarta pergunta, os alunos responderam se aprenderam algo enquanto jogavam o aplicativo, como demonstrado na Figura 24. Os resultados revelaram que 75% afirma-

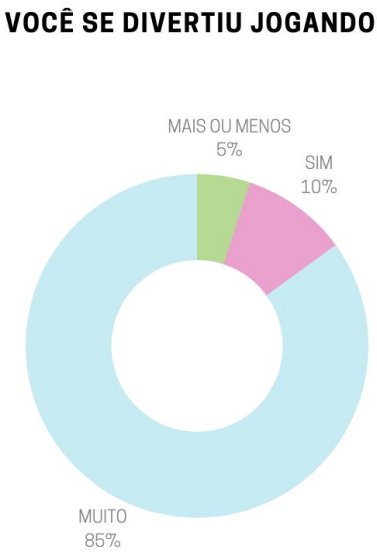

Figura 21. Você se divertiu jogando?

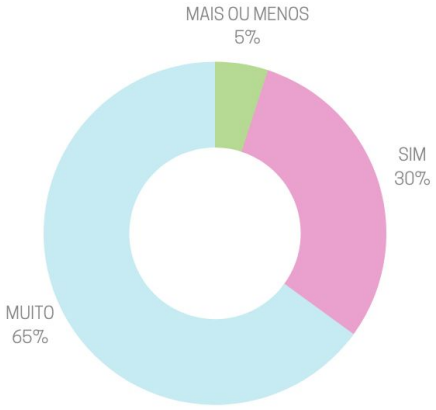

Figura 22. Você achou fácil usar este aplicativo?

ram que aprenderam “muito”, enquanto 25% responderam “sim”. Isso demonstra que é possível conciliar a diversão com a aprendizagem de matemática por meio do aplicativo e reforça a eficácia do aplicativo em proporcionar uma experiência educativa e lúdica simultaneamente, contribuindo para o enriquecimento do processo de ensino-aprendizagem.

Por fim, quando questionados sobre o que mais gostaram no aplicativo, muitos alunos destacaram sua apreciação pelas questões matemáticas que puderam resolver. Além disso, grande ênfase foi dada aos desenhos para colorir, revelando que essa atividade foi muito valorizada pelos estudantes.

Os resultados dessas avaliações desempenharam a análise da eficácia do aplica-

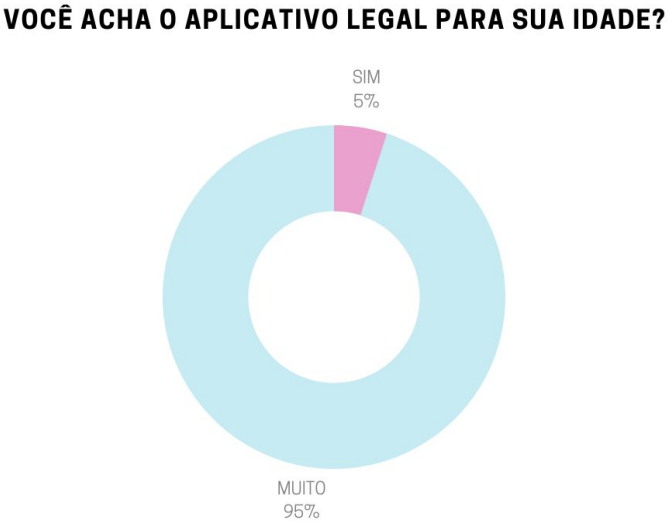

Figura 23. Você acha o aplicativo legal para sua idade?

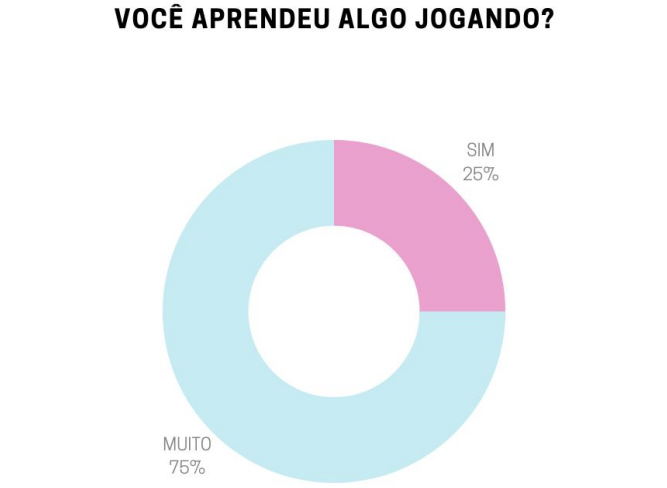

Figura 24. Você aprendeu algo jogando?

tivo e identificaram áreas que necessitaram aprimoramento. Isso garantiu que o *software* pudesse ser refinado de acordo com as necessidades reais dos professores e alunos.

5. Considerações Finais

A dificuldade persistente no Ensino Fundamental em relação ao aprendizado da matemática tem sido um desafio ao longo dos anos. O aplicativo *MathColor* emprega uma combinação de diversão, tecnologia e ensino, buscando minimizar os impactos desse desafio com um aplicativo temático infantil composto por desafios matemáticos e desenhos para colorir.

Os objetivos mencionados no início deste trabalho foram alcançados. No entanto,

durante o desenvolvimento do aplicativo, houveram desafios consideráveis ao tentar criar uma experiência que fosse funcional, educativa e divertida ao mesmo tempo. Além disso, a implementação da funcionalidade de colorir os desenhos foi um obstáculo devido à escassez de bibliotecas disponíveis que pudessem executar essa tarefa. Felizmente, ao longo do processo, através de muita pesquisa, foi encontrada uma biblioteca adequada que permitiu superar esse desafio, possibilitando que a ideia do projeto fosse concretizada.

Além disso, tanto as questões matemáticas quanto a criatividade na coloração dos desenhos foram avaliadas de forma positiva, conforme evidenciado pelos resultados da avaliação. Portanto, é possível concluir que o uso da tecnologia aplicada de forma divertida e atrativa para as crianças, pode facilitar o aprendizado da matemática.

Quanto aos trabalhos futuros, considera-se a possibilidade de expandir o aplicativo, adicionando mais questões, abrangendo uma variedade maior de tópicos matemáticos e também ajustá-lo para mais turmas do Ensino Fundamental. Além disso, a adaptação do aplicativo para *smartphones* é um objetivo importante, uma vez que, atualmente, o *software* está disponível exclusivamente para *tablets*. Adicionalmente, pode-se analisar a inclusão de um *timer* e de uma barra de progresso na tela de seleção de assuntos, a fim de motivar ainda mais as crianças a concluir todas as questões propostas. Também pode-se considerar a criação de uma versão para *desktop* e disponibilizar o jogo para *download* em lojas de aplicativos.

Referências

ABCya.com (2004). Abcya.com. Disponível em: <https://www.abcya.com/games/math-bingo>. Acesso em: 14 de abr de 2023.

BNCC (2018). Base nacional comum curricular. Disponível em: <http://basenacionalcomum.mec.gov.br/images/BNCC_EI_EF_110518_versaofinal_site.pdf>. Acesso em: 12 de abr de 2023.

Boaler, J. (2016). *Desenvolvendo uma Mente para Matemática*. WMF Martins Fontes.

Boaler, J. (2019). *Limitless Mind: Learn, Lead, and Live Without Barriers*. Harper Collins.

Booch, G., Rumbaugh, J., e Jacobson, I. (2006). *UML - Guia do Usuário*. GEN LTC; Tradução da segunda edição.

Borba, M., da Silva, R. S. R., e Gadanidis, G. (2014). *Fases das tecnologias digitais em Educação Matemática*. Autêntica Editora.

Braga, A. S. (2010). *Design de Iterações nos Games*. PUC, São Paulo.

Code, V. S. (2015). Visual studio code. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 16 de abril de 2023.

Cordova, F. e Straubel, J. (2022). Desenvolvimento de uma plataforma para geração de simulados. Disponível em: <https://repositorio.ifsc.edu.br/handle/123456789/2699>. Acesso em: 14 de abr de 2023.

Dart (2011). Dart. Disponível em: <https://dart.dev/>. Acesso em: 25 de abr de 2023.

Dunlap, J. (2006). Queue-linear flood fill: A fast flood fill algorithm. Disponível em: <https://www.codeproject.com/Articles/16405/Queue-Linear-Flood-Fill-A-Fast-Flood-Fill-Algorith>. Acesso em: 23 de set de 2023.

Figma (2016). Figma. Disponível em: <https://www.figma.com/about/>. Acesso em: 16 de abr de 2023.

Filho, A. M. (2011). *Desenvolvimento de Software requer Processo e Gestão*. Revista Espaço Acadêmico, N° 123.

Flutter (2017). Flutter. Disponível em: <https://flutter.dev/>. Acesso em: 25 de abr de 2023.

Gee, J. (2007). *What video games have to teach us about learning and literacy*. Palgrave Macmillan.

IBGE (2019). Pesquisa nacional por amostra de domicílios contínua: Tic 2019. Disponível em: <https://biblioteca.ibge.gov.br/index.php/biblioteca-catalogo?view=detalhes&id=2101794>. Acesso em: 05 de abr de 2023.

Javier, G. (2021). Flood fill in flutter. Disponível em: <https://www.meekcode.com/blog/flood-fill-in-flutter>. Acesso em: 23 de set de 2023.

Kishimoto, T. M. (2011). *Jogo, brinquedo, brincadeira e a educação*. Ed. 14. Cortez.

Lecheta, R. R. (2016). *Android Essencial: edição resumida do livro Google Android*. Novatec.

MathKids (2023). Math kids. Disponível em: <https://play.google.com/store/apps/details?id=com.rvappstudios.math.kids.counting&hl=en&gl=US>. Acesso em: 14 de abr de 2023.

MathPlayground (2002). Division derby. Disponível em: <https://www.mathplayground.com/ASB_Division_Derby.html>. Acesso em: 14 de abr de 2023.

Oliveira, M. S. (2007). Desenvolvimento de aplicações de banco de dados. Disponível em: https://www.ic.unicamp.br/ geovane/mo410-091/Ch06-DBApp-art.pdf. Acesso em: 16 de jun de 2023.

Pais, L. C. (2018). *Didática da matemática: uma análise da influência francesa*. Autêntica, Coleção Tendências em Educação Matemática.

Paiva, F. (2021). Panorama mobile time/opinion box - crianças e smartphones no brasil. Disponível em: ¡https://www.mobiletime.com.br/noticias/29/10/2021/aumenta-o-uso-de-smartphone-por-criancas-brasileiras-de-7-a-9-anos/¿. Acesso em: 19 de abr de 2023.

Pereira, A. S., Shitsuka, D. M., Parreira, F. J., e Shitsuka, R. (2018). *Metodologia da Pesquisa Científica*. 1ª Edição. UAB/NTE/UFSM.

Piaget, J. (1975). *A formação do símbolo: imitação, jogo e sonho, imagem e representação*. 3. ed. Rio de Janeiro: Zahar.

Prensky, M. (2004). *Digital game-based learning*. McGraw-Hill.

Prensky, M. (2012). *Aprendizagem Baseada em Jogos Digitais*. Senac.

Pressman, R. S. (1995). *Engenharia de Software*. Makron Books.

Sadovsky, P. (2007). *Falta Fundamentação Didática no Ensino da Matemática*. Ed. Abril. Nova Escola.

Scrum (2011). Scrum. Disponível em: <https://www.scrum.org/> Acesso em: 30 de maio de 2023.

SQLite (2000). Sqlite. Disponível em: <https://sqlite.org/index.html>. Acesso em: 25 de abril de 2023.

Studio, A. (2013). Android studio. Disponível em: <https://developer.android.com/>. Acesso em: 25 de abr de 2023.

Zanella, L. C. H. (2013). *Metodologia de Pesquisa*. 2. ed. reimpressa. UFSC.