UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

EASYEDU: EDITOR WEB PARA JOGOS MULTITOQUE

FELIPE LOOSE CORSO

BLUMENAU
2017

FELIPE LOOSE CORSO

EASYEDU: EDITOR WEB PARA JOGOS MULTITOQUE

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Ciência da Computação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Prof. Dalton Solano dos Reis, M. Sc. - Orientador

BLUMENAU
2017

EASYEDU: EDITOR WEB PARA JOGOS MULTITOQUE

Por

FELIPE LOOSE CORSO

Trabalho de Conclusão de Curso aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II pela banca examinadora formada por:

Presidente:	Prof. Dalton Solano dos Reis, M. Sc. - Orientador, FURB
Membro:	Prof. Roberto Heinzle, Doutor – FURB
Membro:	Prof. Marcel Hugo, M. Sc. – FURB

Blumenau, 06 de julho de 2017

Dedico este trabalho a minha família, aos meus amigos e colegas de trabalho por me apoiar ao longo deste curso, em especial aqueles que de alguma maneira ajudaram para a realização dele.

AGRADECIMENTOS

À minha família e amigos por todo apoio prestado durante esses anos.

Ao meu pai Isomar Corso (in memoriam).

Ao meu orientador, Dalton Solano dos Reis, por acreditar na realização deste trabalho e por todo apoio prestado durante o desenvolvimento do mesmo.

A todos os professores da FURB que de alguma forma contribuíram para a minha formação pessoal e profissional.

Ao professor Mauricio Capobianco Lopes por ter emprestado o LIFE e os seus alunos para a realização dos testes do editor.

A todos aqueles desenvolvedores que não medem esforços para repassar o conhecimento, mantendo fóruns e blogs que sempre nos salvam em momentos difíceis.

A todos que direta ou indiretamente fizeram parte da minha jornada acadêmica e profissional.

A simplicidade é o último grau da sofisticação.

Leonardo da Vinci

RESUMO

Este trabalho apresenta o desenvolvimento de um editor web para jogos multitoque como complemento aos métodos tradicionais de ensino. O editor possui dois módulos: o módulo de edição que possibilita ao professor customizar os layouts disponíveis e desenvolver atividades para diferentes conteúdos e alunos. Nele é possível que sejam criadas atividades com jogos de letras e de imagens, utilizando recursos multimídia. Os conteúdos criados posteriormente podem ser disponibilizados para os alunos por meio de um QR Code, um link ou um arquivo. Já o segundo módulo de jogo possibilita aos alunos importar os conteúdos e praticar as atividades. Os exercícios podem ser praticados no modo partida única ou partida dupla. Os jogos possuem suporte multitoque, o que permite explorar o aspecto lúdico, incentivando a interação entre os alunos, contribuindo com a criatividade e o espírito de competição e cooperação. O EasyEdu está integrado com o Google, possibilitando que os conteúdos desenvolvidos sejam armazenados no Google Drive do professor. Para o desenvolvimento foram utilizados AngularJS e o suporte ao multitoque tornou-se viável por meio da utilização do Hammer.js. Ele permite mapear gestos e associar funções quando estes ocorrem. O experimento realizado com os especialistas mostrou que a utilização do EasyEdu como um método para apresentar conteúdos e desenvolver exercícios pode auxiliar na compreensão e na fixação do assunto abordado. Com isso, é possível assegurar que objetivo inicial do trabalho de criar uma ferramenta diferenciada que auxiliasse no processo de ensino e aprendizagem foi alcançado.

Palavras-chave: Editor. Jogos educacionais. Multitoque. Multiplataforma. Google Drive. Angular.

ABSTRACT

This work presents the development of a web editor for multitouch games as a complement to traditional teaching methods. The editor has two modules: the editing module which allows the teacher to customize the available layouts and develop activities for different content as for students. It is possible to create activities with games of letters and images, using multimedia features. The created contents, can posteriorly be made available to students through a QR Code, a link or a file. The second is the game module that allows the students to import the contents and practice the activities. The exercises can be practiced in single match mode or double match mode. The games have multitouch support, which allows to explore the playful aspect, encouraging the interaction between the students, contributing with the creativity and the spirit of competition and cooperation. The EasyEdu is integrated with Google, enabling the developed content to be stored in the teacher's Google Drive. For development AngularJS was used and the multitouch support became viable through the use of Hammer.js. It allows to map gestures and associate functions when they happen. The experiment performed with the experts showed that the use of EasyEdu as a method to present contents and develop exercises can help in understanding and fixing the subject matter. Therewith, it is possible to ensure that the initial objective from the work of creating a differentiated tool that would aid in the teaching and learning process have been achieved.

Key-words: Editor. Educational games. Multitouch. Multiplatform. Google Drive. Angular.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AJAX – Asynchronous Javascript and XML

API – Application Programming Interface

CSS – Cascading Style Sheets

FURB - Fundação Universidade Regional de Blumenau

GB – Gigabyte

GCG - Grupo de Pesquisa em Computação Gráfica, Processamento de Imagens e Entretenimento Digital

HTML – Hyper Text Markup Language

JSON - JavaScript Object Notation

LabVirt – Laboratório Didático Virtual

LIFE - Laboratório Interdisciplinar de Formação de Educadores

OA - Objetos de Aprendizagem

PCN - Parâmetros Curriculares Nacionais

RF – Requisito Funcional

RIVED - Rede Interativa Virtual de Educação

RNF – Requisito Não Funcional

UML - Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

O ser humano naturalmente vive em sociedade por vontade ou necessidade. A sociedade na qual este indivíduo está inserido desempenha papel fundamental na sua formação. Com base na interação social e na troca de experiências são desenvolvidas diferentes formas de linguagem, novas maneiras de expressar e interpretar a cultura, formulam-se opiniões e criam-se diferentes formas de pensar. Todas essas relações contribuem para o desenvolvimento do conhecimento (ROSS, 2006, p.274).

Desenvolver o conhecimento fez com que o ser humano evoluísse com o passar dos anos. A busca pelo conhecimento é constante e de fato vem obtendo êxito. Podemos citar como frutos dessa busca a invenção da escrita até o advento da internet (DE MELLO; TEIXEIRA, 2012, p.1). Adquirir o conhecimento é muito importante, poder transmitir esse conhecimento para os demais é tão importante quanto. Por isso é tão relevante encontrar maneiras diferenciadas de como representar e repassar o conhecimento adquirido.

Para contribuir com a disseminação do conhecimento podemos utilizar ferramentas de fácil acesso que estão presentes em nosso dia a dia, como os recursos tecnológicos por exemplo. Quando associadas ao processo lúdico essas ferramentas permitem trabalhar qualquer conteúdo de forma prazerosa e divertida, auxiliando pessoas que possuem dificuldades em adquirir conhecimentos da maneira tradicional (FALKEMBACH, 2007, p.1).

Incluir novos métodos e tecnologias como formas de ensino podem contribuir para o aumento da motivação dos alunos para a aprendizagem. Os jogos são ótimas ferramentas para serem utilizadas, pois prendem a atenção, contribuem com a integração, entusiasmam e ensinam enquanto divertem. Os jogos conseguem transmitir informações de várias maneiras, estimulando diversos sentidos ao mesmo tempo e sem se tornarem cansativos (FALKEMBACH, 2007, p.1). Utilizar dispositivos como smartphones, tablets e notebooks que possuem suporte à tecnologia multitoque, que é um método de entrada que permite que dois ou mais dedos possam ser utilizados na tela de uma só vez, podem agregar valor ao processo de ensino e aprendizagem (GALO; SERRANO; ROCHA, 2009, p.1).

Diante do exposto, foi desenvolvido um editor de jogos multitoque para a plataforma web, tendo como objetivo gerar insumos para auxiliar os alunos e educadores no ensino de seus objetos de estudo. Com o uso do editor, o professor poderá explorar o lado pedagógico, criando *templates* customizados com base no conteúdo abordado em sala. Os jogos multitoque criados no editor poderão ser utilizados de duas maneiras. A primeira delas com duas partidas

em paralelo, na qual as equipes poderão “duelar”. A outra maneira é tendo apenas uma partida ativa, em que uma equipe ou usuário buscará resolver o exercício no menor tempo possível.

1.1 OBJETIVOS

O objetivo é desenvolver um editor web para jogos multitoque.

Os objetivos específicos do trabalho são:

a) fornecer um ambiente em que o professor possa usar layouts customizáveis inserindo sons e imagens com base no conteúdo abordado em sala;
b) fornecer um ambiente no qual os alunos possam exercitar o conteúdo estudado;
c) permitir a interação entre os alunos e que estes possam resolver as atividades propostas de forma colaborativa ou competitiva;
d) validar com usuários a aplicabilidade e a aceitação do editor web para jogos multitoque.

1.2 ESTRUTURA

Esta monografia está estruturada em quatro capítulos.

O primeiro é dedicado à introdução ao tema, aos objetivos gerais e específicos e à estrutura do trabalho.

O segundo capítulo é composto pela fundamentação teórica, abordando os tópicos necessários para o entendimento do trabalho, além de exibir os trabalhos corretados a este.

No terceiro capítulo é apresentado o desenvolvimento do EasyEdu, no qual são listados os requisitos e a especificação por meio de diagramas. Na sequência é detalhada a implementação, descrevendo as técnicas e ferramentas utilizadas, mostrando e explicando os trechos relevantes do código. Neste mesmo capítulo são apresentadas as análises dos resultados, comentando sobre o experimento realizado e sobre o desenvolvimento. Ao final é feita a comparação com os trabalhos correlatos.

Por fim, o quarto capítulo apresenta as considerações finais e as conclusões do trabalho, complementando com sugestões de extensões para futuros trabalhos.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo explorar os principais assuntos necessários para realização deste trabalho. A seção 2.1 apresenta os Objetos de Aprendizagem (OA). A seção 2.2 aborda os benefícios da adoção de tecnologias e métodos diferenciados, como meios de ensino. Na seção 2.3 é abordada a tecnologia que será utilizada para o desenvolvimento do trabalho e, finalmente, na seção 2.4 são descritos alguns trabalhos correlatos.

2.1 OBJETOS DE APRENDIZAGEM

Objetos de Aprendizagem (OA) são recursos educacionais, que podem ser desenvolvidos em diversos formatos e linguagens, com o objetivo de mediar e qualificar o processo de ensino e aprendizagem (BRUZZI et al., 2017). Conforme o repositório Careo (2002, p.1), um objeto de aprendizagem:

> [...] é qualquer recurso digital com um valor pedagógico demonstrado, que pode ser usado, reusado ou referenciado para suporte de aprendizagem. Os objetos de aprendizagem podem assim ser uma applet Java, uma animação Flash, um quis online ou um filme QuickTime, mas pode também ser uma apresentação Power Point ou arquivo .pdf, uma imagem, um site ou uma web Page

Diferentes termos são utilizados para tratar os Objetos de Aprendizagem, tais como: objetos educacionais, objetos de conhecimento, componentes de software educacional, conteúdos de objetos compartilháveis, objetos de aprendizagem multimídia, entre outros. Independente da terminologia adotada para tratar os OAs, a capacidade de reutilização é uma característica compartilhada por todos. Spinelli (2007, p. 7) diz que:

> Um objeto virtual de aprendizagem é um recurso digital reutilizável que auxilie na aprendizagem de algum conceito e, ao mesmo tempo, estimule o desenvolvimento de capacidades pessoais, como, por exemplo, imaginação e criatividade. Dessa forma, um objeto virtual de aprendizagem pode tanto contemplar um único conceito quanto englobar todo o corpo de uma teoria.

Aderir ao uso da tecnologia é um processo de transformação e constitui-se uma realidade sem volta. Incluir as principais inovações tecnológicas na educação pode resultar em mudanças de todo paradigma pedagógico, modificando a maneira como as pessoas ensinam e aprendem (AUDINO, 2012). Essa transformação pode acontecer também na forma como materiais educacionais são desenvolvidos e oferecidos para aqueles que desejam aprender. A versatilidade oferecida pelos OAs pode favorecer práticas pedagógicas que alcancem significativos avanços no processo de construção do conhecimento (WILEY, 2000).

É possível encontrar na internet exemplos de práticas pedagógicas utilizando Objetos de Aprendizagem. Alguns dos exemplos estão de acordo com as áreas destacadas nos Parâmetros Curriculares Nacionais (PCNs), que são orientações gerais, propostas pelo

Ministério da Educação e visam orientar o planejamento escolar. Em 1997, o Ministério da Educação juntamente com as suas secretarias trabalhou em cooperação com os Estados Unidos, iniciando-se um processo de criação de material digital com fins pedagógicos, dando origem à plataforma que hoje é conhecida como Rede Interativa Virtual de Educação (RIVED). Em 2007, o Ministério da Educação lançou um edital para financiar o desenvolvimento de recursos educacionais multimídia para as disciplinas do currículo base do Ensino Médio. Este edital gerou diversas iniciativas de produção de conteúdos educacionais, para disponibilizá-los foi criado o Portal do Professor (BIELSCHOWSKY; PRATA, 2010). Outro exemplo de espaço que reúne diferentes Objetos de Aprendizagem é o Laboratório Didático Virtual (LabVirt) dedicado às áreas de Física e Química. No entanto, Behar et al. (2009, p. 33) destaca que:

> [...] a utilização da tecnologia pela tecnologia não é suficiente para a contemplação de uma nova concepção educacional. O diferencial está no planejamento pedagógico em que esses recursos digitais estarão inseridos. Será preciso contemplar uma pedagogia baseada na pesquisa, no acesso à informação, na complexidade, na diversidade e na imprevisibilidade.

Em outras palavras, mesmo com o avanço tecnológico acelerado e um crescente surgimento de ferramentas e recursos educacionais o professor ainda é uma peça fundamental no processo ensino-aprendizagem. A diferença é que um novo papel foi atribuído ao educador, agora além de possuir o conhecimento o professor terá a responsabilidade de conhecer as novas ferramentas e métodos disponíveis e avaliar qual destas se encaixam na sua realidade e de seus alunos. Utilizar um recurso tecnológico por si só não é suficiente para contemplação de uma metodologia de ensinar, é necessário ter os objetivos claramente definidos, o que fará do recurso escolhido o melhor é a maneira como ele será empregado na prática pedagógica (AUDINO, 2012).

2.2 A ADOÇÃO DE MÉTODOS DIFERENCIADOS COMO MEIOS DE ENSINO

A adoção de métodos diferenciados para o ensino como aulas expositivas, atividades que envolvam multimídias, laboratórios, jogos e aulas de campo podem ser muito benéficas para os alunos. Estes métodos podem instigar a busca pelo conhecimento e podem facilitar a compreensão do assunto abordado, possibilitando a criação de atividades colaborativas que visem o aprendizado mútuo (FIALHO, 2008). O educador deve procurar introduzir em suas aulas recursos diferenciados que visem enriquecer o processo de ensino e aprendizagem explorando o lado lúdico, complementando os materiais que já estão à disposição. Conforme Fialho (2008, p. 16):

> A exploração do aspecto lúdico, pode se tornar uma técnica facilitadora na

> elaboração de conceitos, no reforço de conteúdos, na sociabilidade entre os alunos, na criatividade e no espírito de competição e cooperação, tornando esse processo transparente, ao ponto que o domínio sobre os objetivos propostos na obra seja assegurado.

A formação de um indivíduo se dá por meio de experiências adquiridas em atividades simples como brincar e jogar. Durante esse processo as crianças são instigadas a desenvolver o senso de decisão, interagir com outras pessoas e permitem adquirir conhecimentos e habilidades no âmbito da linguagem. Na prática dessas atividades são moldados seus valores que possibilitam a resolução de conflitos, e de forma conjunta, buscam soluções para os problemas encontrados, criando regras de convivência social e de participação. Deste modo, entende-se que os jogos e as brincadeiras são instrumentos pedagógicos importantes e determinantes para a evolução da criança, pois desenvolvem habilidades necessárias para o seu processo de alfabetização e letramento (VIEIRA, 2010).

Os jogos representam um meio relevante para desenvolver o intelecto e o social da criança. Ao utilizar-se de atividades lúdicas o professor propicia ao aluno a oportunidade de ter acesso ao conhecimento de uma maneira que ele não está habituado, podendo instigar a sua curiosidade e prender a sua atenção mais facilmente (SOARES, 2013). Sobre os jogos Silveira e Barone (1998, p. 2) comentam que:

> [...] Um dos usos básicos e muito importantes é a possibilidade de construir-se a autoconfiança. Outro é o incremento da motivação [...] um método eficaz que possibilita uma prática significativa daquilo que está sendo aprendido. Até mesmo o mais simplório dos jogos pode ser empregado para proporcionar informações factuais e praticar habilidades, conferindo destreza e competências.

Ao optar por um método diferenciado de ensino, deve-se tomar cuidado com quais tipos de materiais e equipamentos adicionais são necessários. Alguns recursos podem não fazer parte da realidade dos alunos ou da escola comprometendo os resultados. O professor poderá optar por métodos que utilizem materiais que possam ser confeccionados pelos alunos, ou que utilizem equipamentos que são facilmente encontrados nos dias atuais como smartphones, tablets, notebooks e desktops. Uma opção que, quando disponível, deve ser considerada é a mesa digital multitoque, pois devido suportar múltiplos toques, pode cooperar na interação social entre os alunos, além de ser uma forma de ensino diferente da que os estudantes estão habituados.

2.3 TECNOLOGIA MULTITOQUE

A tecnologia *touchscreen,* que é o termo utilizado para “toque na tela”, vem evoluindo constantemente nos últimos anos. Melhorias no hardware possibilitaram o aumento da precisão da captura do toque e a quantidade de toques suportados. Dessa evolução surgiram os

displays que suportam múltiplos toques na tela denominados multitoque ou *multitouch* no termo em inglês (ASSIS, 2009). Com essas melhorias o campo de aplicação dos equipamentos que possuem essa tecnologia se expande, tornando possível empregá-los nos mais variados ambientes como nas corporações, no varejo, na indústria, na educação entre outros (AQUA, 2014). Smartphones, tablets, notebooks, *totens* de autoatendimento são alguns dos equipamentos que suportam *multitouch*. Conforme Galo, Serrano e Rocha (2009, p. 1):

> No início os detectores eram colocados em pequenos buracos na tela, o que foi logo abandonado porque eles juntavam poeira e resultavam em falhas ao "sentir" o dedo. Em seguida foi colocado um plástico transparente para protegê-los. Mas os sensores infravermelhos também não duraram muito porque surgiram novas e melhores tecnologias para detectar os movimentos dos dedos e que realmente tornaram as próprias telas sensíveis, como sensores elétricos, acústicos e óticos.

Uma superfície multitoque tem por objetivo possibilitar que as pessoas compartilhem e desenvolvam suas ideias de modo mais interativo. Porém, para conseguir ampliar a aplicação do conceito de multitoque e tornar seu uso mais colaborativo, faz-se necessário uma superfície maior, com uma boa resolução e que tenha suporte para uma quantidade maior de toques simultâneos na tela. Desta necessidade surgiu o conceito de mesas multitoque (MICROSOFT, 2012). O mercado ainda é um pouco modesto, mas é possível encontrar dispositivos com tamanhos variáveis, suportando mais de 4 toques simultâneos na tela dependendo do modelo e do fabricante.

Para este projeto é utilizada a mesa *multitouch* modelo G³ Plus Overlay da PQ Labs Multi-Touch, disponível no Laboratório Interdisciplinar de Formação de Educadores (LIFE) da Fundação Universidade Regional de Blumenau (FURB), apresentada na Figura 1.

Figura 1 – Mesa multitoque do LIFE

Fonte: elaborado pelo autor.

A mesa possui uma tela de 50 polegadas e suporta 12 toques simultâneos. O custo aproximado foi de R$ 26.000,00 incluindo tela, TV, computador e mesa já customizada (GRUPO DE PESQUISA EM COMPUTAÇÃO GRÁFICA, PROCESSAMENTO DE

IMAGENS E ENTRETENIMENTO DIGITAL, 2014). Os sensores da mesa digital multitoque são ativados por meio dos dedos, tornando a ação de jogar mais natural. Adicionalmente, contribui para o aumento da percepção do tato para os alunos das séries iniciais, pois não é necessário aprender a utilizar um controlador como, por exemplo, um *joystick* (AQUA, 2014). Na Figura 2 é demonstrada a interação com a mesa multitoque do LIFE, fazendo uso do jogo de quebra-cabeça desenvolvido nas aulas de Sistemas Multimídias pelos alunos do curso de Ciência da Computação da FURB.

Figura 2 – Interação com a mesa multitoque do LIFE

Fonte: elaborado pelo autor.

2.4 TRABALHOS CORRELATOS

A seguir estão relacionados três trabalhos correlatos ao proposto. A seção 2.4.1 detalha o PAR (Peço, Ajudo, Recebo) que é um jogo que utiliza multitoque (CALPA, 2012). A seção 2.4.2 detalha um motor de jogos 2D de encaixe de imagens na plataforma Android (MACIEL, 2015). Por fim, a seção 2.4.3 detalha o jogo Dominó das Funções Inorgânicas (ASSIS; SOUZA, 2012).

2.4.1.1 PAR (Peço, Ajudo, Recebo)

PAR (Peço, Ajudo, Recebo) é um jogo colaborativo em mesa multitoque para apoiar a interação social de usuários com autismo. O objetivo do jogo é incentivar o trabalho colaborativo e a geração de interação social entre usuários com diferentes graus de autismo (CALPA, 2012). A temática do jogo consiste em conseguir as peças dos uniformes e vestir os jogadores de uma equipe de futebol.

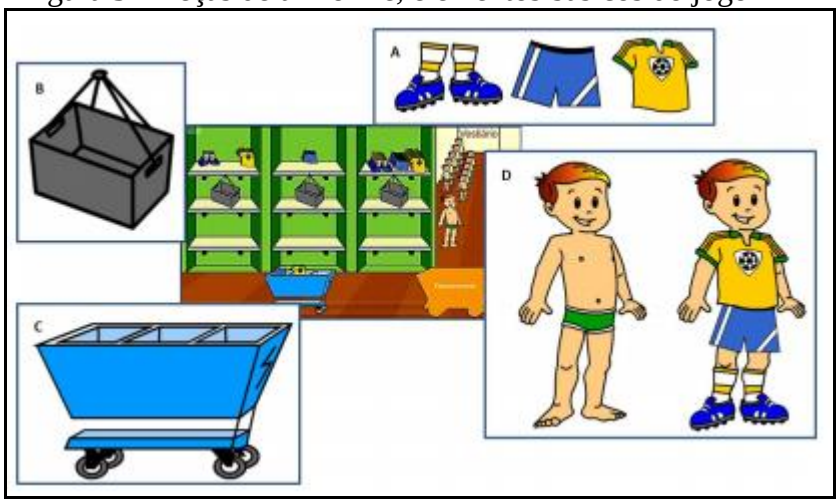
O jogo é divido em quatro fases, denominadas dimensões, e foram adaptadas para o público-alvo. Os usuários executam diferentes funções dependendo do posicionamento ao redor da mesa multitoque e da dimensão (CALPA, 2012). Conforme Calpa (2012, p. 15):

> Três dessas dimensões têm certas restrições na interação sobre os objetos do jogo,

> para motivar gradualmente a necessidade de colaboração entre os usuários. A quarta dimensão de colaboração não tem restrições, permitindo a identificação de estratégias colaborativas realizadas pelos usuários em um ambiente com livre interação com os objetos de aplicação.

A Figura 3 traz detalhes dos elementos básicos do jogo. Na Figura 3a são apresentadas as peças do uniforme: tênis, short e camisa, que estão na prateleira. A Figura 3b exibe o cesto em que será colocada cada peça do uniforme e baixada da prateleira. A Figura 3c mostra o carrinho com vagas para receber as três peças que serão baixadas nos cestos. A Figura 3d exemplifica à esquerda um jogador na fileira aguardando para ser vestido e um jogador à direita após ser vestido.

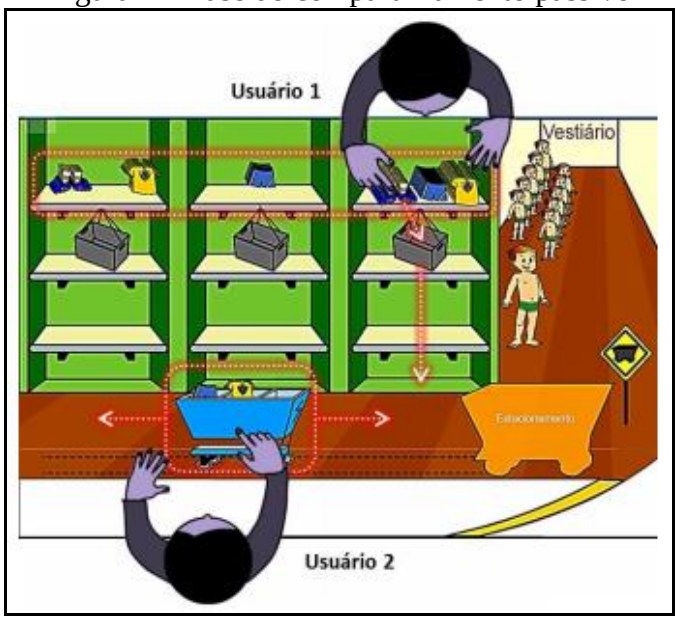
Figura 3 – Peças de uniforme, elementos básicos do jogo PAR



Fonte: Calpa (2012).

A primeira dimensão é a do compartilhamento passivo, representada na Figura 4, na qual os papéis para cada usuário são apenas de ação e resposta de um para o outro. O compartilhamento de recursos tem apenas o objetivo de concluir a tarefa sem necessidade de troca de informações (CALPA, 2012).

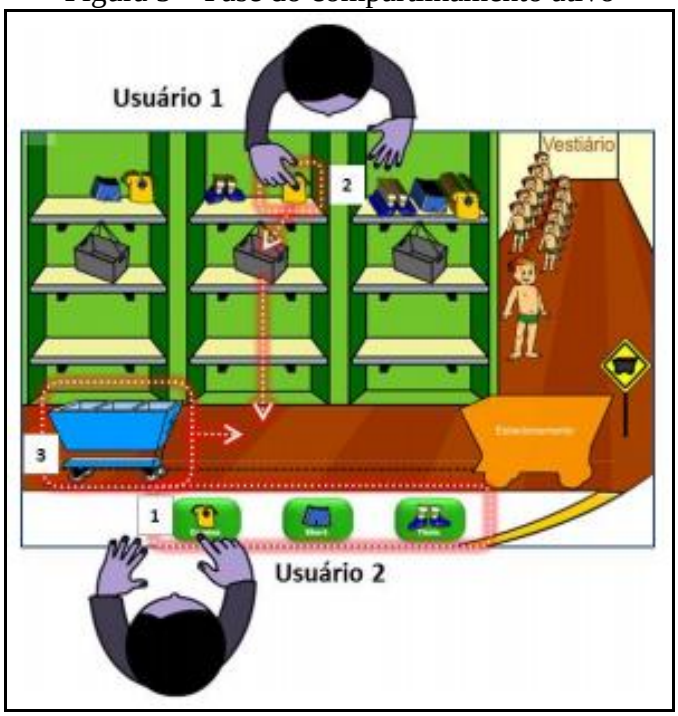
Figura 4 – Fase do compartilhamento passivo



Fonte: Calpa (2012).

A segunda dimensão representada na Figura 5 é a do compartilhamento ativo, em que além do compartilhamento dos recursos também é necessária a interação entre os usuários. Nesta fase no passo 1, o usuário 2 solicita uma peça de uniforme, no passo 2 o usuário 1 envia a peça pedida e no passo 3 o usuário 2 recebe a peça enviada.

Figura 5 – Fase do compartilhamento ativo

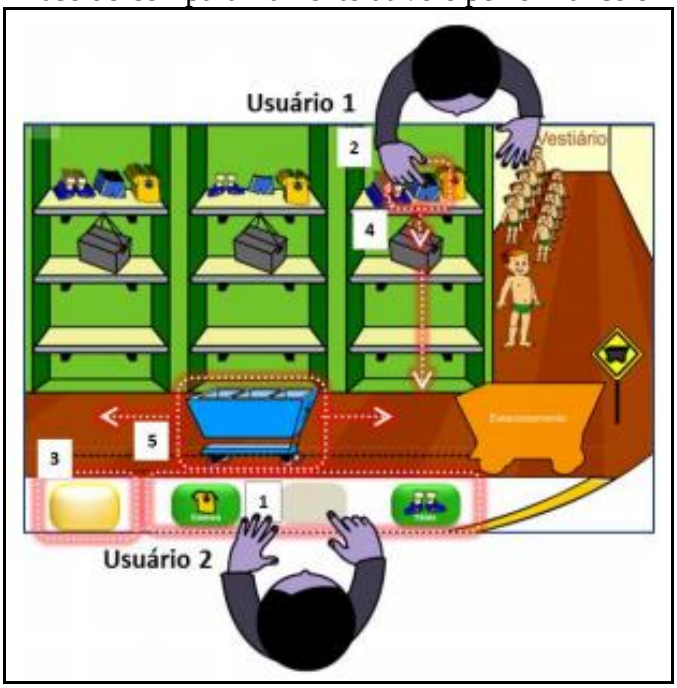


Fonte: Calpa (2012).

A terceira dimensão representada pela Figura 6 é denominada compartilhamento ativo e performance em conjunto, na qual são compartilhados os recursos e as informações, possibilitando também que um indivíduo ajude o outro em determinadas situações (CALPA,

2012). Nesta fase, no passo 1 o usuário 2 pede uma peça do uniforme, o passo seguinte 2 o usuário 1 pega a peça pedida (solicitando a ajuda do usuário 2 por meio de som do sistema). No passo 3 o usuário 2 precisa apertar o botão para abrir as caixas, no passo 4 o usuário 1 envia a peça pedida e no último passo 5 o usuário 2 recebe a peça enviada.

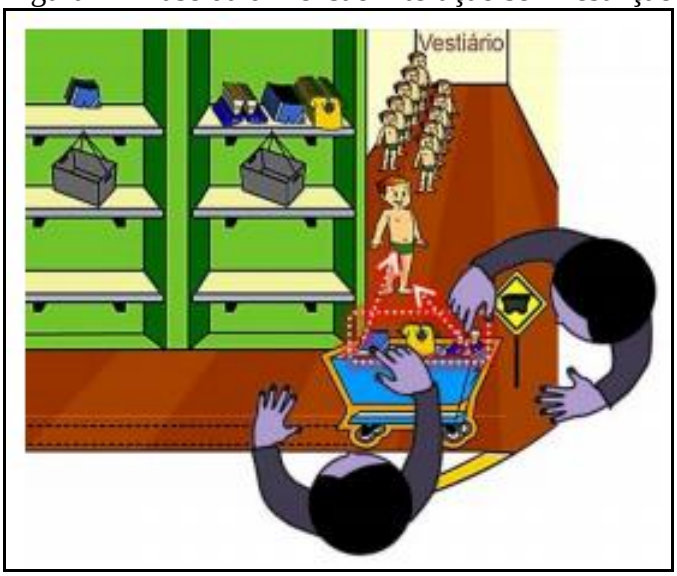
Figura 6 – Fase do compartilhamento ativo e performance em conjunto



Fonte: Calpa (2012).

A Figura 7 representa a última fase do jogo e ocorre quando o carrinho chega ao estacionamento. Neste momento, aplica-se a dimensão interação sem restrição. Segundo Calpa (2012, p. 32), “Esta dimensão não atribui nenhum papel restrito para cada usuário nem restrições sobre os objetos do jogo, permitindo assim uma interação livre sobre esses objetos para colaborar.”. Nesta etapa os usuários 1 e 2 podem uniformizar os jogadores escolhendo a peça que quiserem na ordem em que desejarem. Após uniformizar o jogador, é apresentada uma mensagem informando a quantidade de jogadores já uniformizados e se eles desejam prosseguir. Caso optem por dar continuidade no jogo, o usuário 2 deverá levar o carrinho até o armazém para continuar pedindo e recebendo as peças.

Figura 7 – Fase da dimensão interação sem restrição



Fonte: Calpa (2012).

Destaca-se a maneira como o jogo foi desenvolvido buscando utilizar todos os recursos que a mesa multitoque proporciona, a fim de promover a interação social entre os usuários. Durante o processo de avaliação, os usuários apresentaram diferentes características com relação ao jogo, tais como: necessidade de repetição de indicações sobre as ações no sistema, maior interesse em determinados recursos do sistema, não pela funcionalidade, mas talvez pela sua forma ou cor, dificuldades para coordenar um processo colaborativo. Identificaram algumas dificuldades em relação ao comportamento dos jogadores frente ao jogo, visto que um mesmo indivíduo com autismo pode mudar em diferentes aspectos de um dia para o outro, sem prévio aviso (CALPA, 2012).

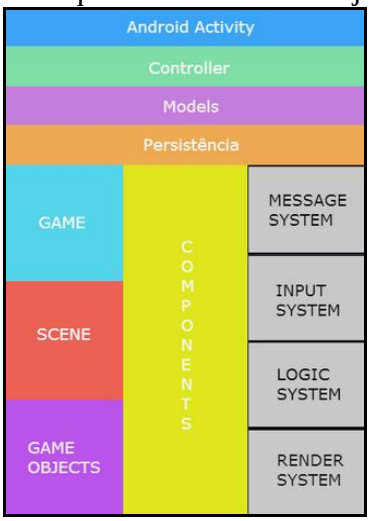
2.4.1.2 Motor de jogos 2D de encaixe de imagens na plataforma Android

Maciel (2015) desenvolveu um motor de jogos 2D de encaixe de imagens e também criou uma aplicação de jogos de encaixe. Para a criação do motor foi utilizado o conceito de arquitetura orientada a componentes, que permite que a implementação seja encapsulada em um componente e reutilizada por qualquer objeto de um jogo. O editor web para jogos multitoque será desenvolvido seguindo este mesmo conceito, no qual o professor poderá criar layouts customizados, definindo as palavras e utilizando seus próprios arquivos de mídia. Segundo Maciel (2015, p. 46), as camadas da arquitetura do motor de jogos 2D apresentadas na Figura 8 são divididas da seguinte maneira:

> A Activity no Android é responsável por todo o ciclo de vida de uma aplicação, coordenando sua execução e como responder a eventos de interação. Já os controllers são responsáveis por efetuar requisições para a camada de persistência, podendo consultar uma entidade específica ou solicitar que uma entidade seja persistida no banco de dados. A Activity também é responsável por fazer as

chamadas para a inicialização do motor, que através do Game coordena a execução do jogo. O Game possui a Scene que está em execução que contém os GameObjects do jogo e seus componentes. O motor faz a notificação aos seus sistemas sendo eles o MessageSystem, InputSystem, LogicSystem e RenderSystem para que cada sistema dispare seus respectivos métodos de atualização implementado nos componentes.

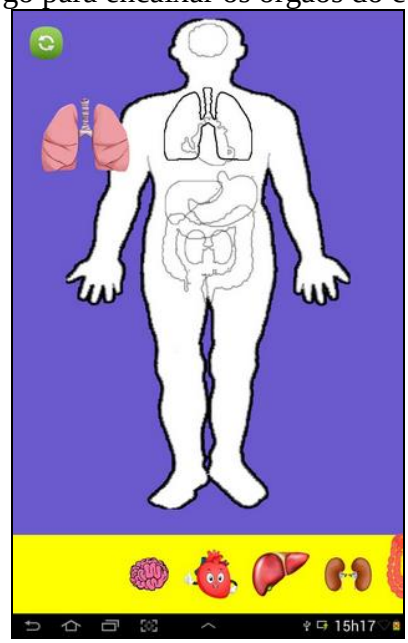
Figura 8 - Arquitetura do motor de jogos 2D



Fonte: Maciel (2015).

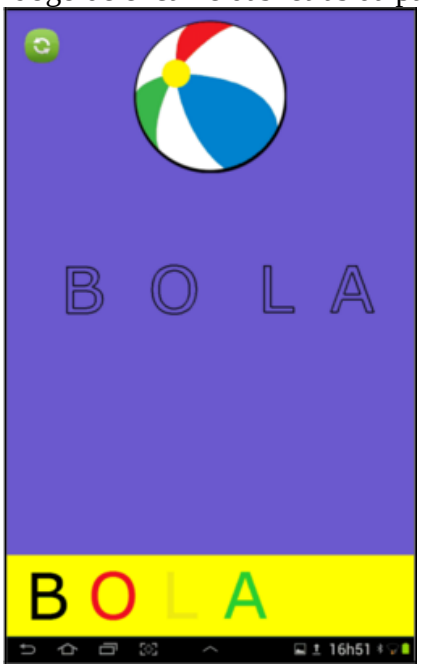
Para utilizar a camada de encaixe devem ser criadas pranchas que armazenam as peças utilizadas no jogo e suas imagens seguindo o padrão esperado. Para a demonstração do conceito utilizado no desenvolvimento da arquitetura e do motor de jogos 2D foram criados dois jogos. O primeiro jogo, para a demonstração conforme apresentado na Figura 9, no qual o usuário deve arrastar e encaixar os órgãos do corpo humano e o segundo jogo, conforme apresentado na Figura 10, no qual o usuário deve arrastar e encaixar as letras da palavra bola.

Figura 9 - Jogo para encaixar os órgãos do corpo humano



Fonte: Maciel (2015).

Figura 10 - Jogo de encaixe das letras da palavra bola



Fonte: Maciel (2015).

Após o usuário levantar o dedo do objeto é efetuada uma verificação para validar se houve colisão entre a origem e o destino. A detecção de colisão ocorre através da verificação da intersecção entre as duas *boundbox*. Quando houver colisão o objeto é desativado e encaixado na sua posição de destino (MACIEL, 2015). O motor de jogos 2D possui uma tela de configuração, na qual é possível selecionar a dificuldade de encaixe. Também é possível optar por destacar a peça de destino e se a aplicação deve arrastar para o destino peças que estejam próximas facilitando a jogabilidade (MACIEL, 2015).

2.4.1.3 Dominó das Funções Inorgânicas

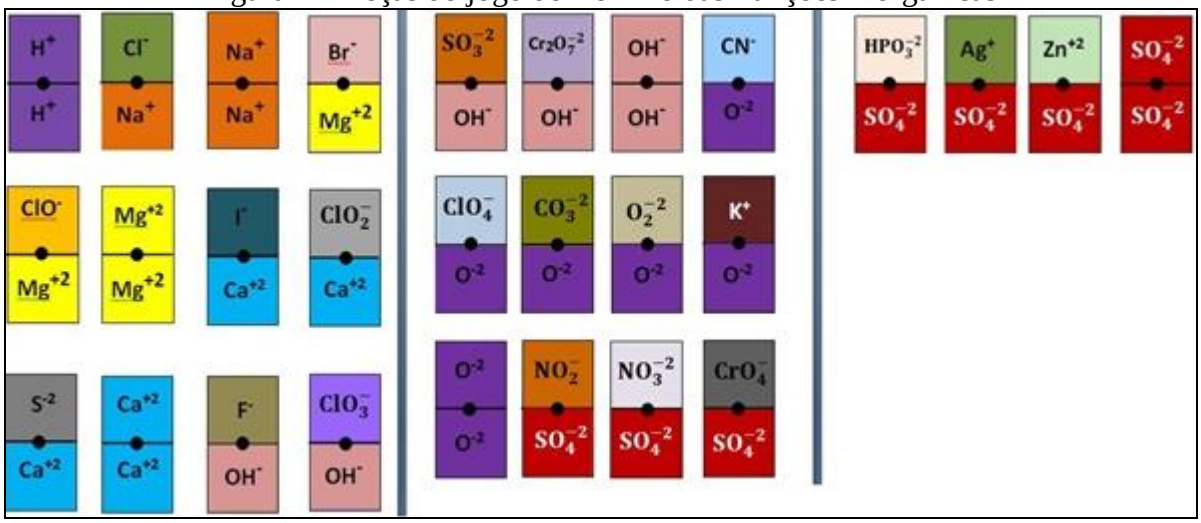
O Dominó das Funções Inorgânicas aborda a importância de introduzir jogos e atividades diferenciadas para tornar o aprendizado mais prazeroso e interessante. A ideia é apresentar o mesmo conteúdo que seria passado da forma convencional na forma de um jogo, abordando metodologias diferentes das empregadas no cotidiano (FIALHO, 2008).

Para a criação do jogo de Dominó das Funções Inorgânicas foram escolhidos os principais cátions e ânions para formar as funções inorgânicas (Figura 11). As regras são as mesmas do jogo convencional de dominó; sendo que algumas alterações devem ser observadas, tais como: 1) as peças devem ser embaralhadas de modo que os participantes não vejam os íons; 2) as peças devem ser distribuídas e aquele que possuir a carroça de Mg2+ irá iniciar o jogo. Caso nenhum dos jogadores tenha a peça, a prioridade de carroças será Ca2+, H+, Na+, OH-, O2-e SO42-; 3) os jogadores seguintes deverão montar os compostos

obedecendo às valências dos íons; 4) para pontuar, o jogador deverá dizer o nome do composto formado. Cada acerto valerá 5 (cinco) pontos. Caso o jogador erre a peça ou o nome do composto, os 5 (cinco) pontos irão para o jogador anterior; 5) o jogador que terminar primeiro as peças, iniciará a próxima rodada; 6) o vencedor será aquele que somar 200 (duzentos) pontos, ou mais, ao encerrar uma rodada (ASSIS; SOUZA, 2012). Conforme Assis e Souza (2012, p. 1):

> Através dos dados coletados podemos perceber que 60% dos alunos não gostam das aulas de Química usando apenas livros e cadernos, ou seja, a maioria dos estudantes precisa de ferramentas diferenciadas no ensino desta ciência. Os jogos representam 75% da opinião dos alunos como um atrativo nas aulas de Química, 85% dos estudantes afirmaram que o Dominó das Funções Inorgânicas, [...] é um recurso prazeroso para o aprendizado da Química e 75% se posicionaram positivamente tratando como uma metodologia favorável para a memorização dos cátions, ânions e nomes dos principais compostos inorgânicos.

Figura 11- Peças do jogo de Dominó das Funções Inorgânicas



Fonte: Assis e Souza (2012).

3 DESENVOLVIMENTO

Neste capítulo é feita a descrição das particularidades técnicas e o detalhamento do trabalho, apresentando as suas características. Na seção 3.1 são enumerados os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF) do projeto desenvolvido. A especificação por meio de diagramas de atividades, pacotes, classes, arquitetura e MER é apresentada na seção 3.2.

Na seção 3.3 é feito o detalhamento da implementação com destaque para os principais pontos, apresentando e comentando os trechos relevantes do código e a operacionalidade em nível de usuário. Por fim, a seção 3.4 apresenta o experimento realizado e os resultados obtidos.

3.1 REQUISITOS

O editor proposto deve possuir dois módulos, sendo que para cada um deles há requisitos que podem ou não ser diferentes. Para facilitar a compreensão, os requisitos foram separados pelo módulo ao qual eles pertencem, e o que é comum a ambos é apresentado por último.

O editor web para jogos multitoque deverá:

a) fornecer no mínimo dois layouts, um para criar jogos de letras e outro para imagens (Requisito Funcional - RF);
b) fornecer *templates* de exemplo pré-configurados (RF);
c) fornecer a opção para a criação de *templates* customizando os layouts (RF);
d) permitir que os *templates* sejam exportados (RF);
e) permitir gerar um QR Code para acessar as atividades (RF).

Os jogos desenvolvidos no editor deverão:

a) fornecer a opção de uma ou duas partidas em paralelo (Requisito Funcional - RF);
b) tratar as partidas das duas equipes de forma independente (RF);
c) apresentar o tempo da equipe assim que a partida for finalizada (RF);
d) permitir que os objetos sejam arrastados até o espaço correspondente (RF);
e) verificar se o objeto selecionado está correto ou incorreto conforme a regra do jogo (RF);
f) permitir que sejam importados *templates* (RF);
g) permitir acessar o conteúdo utilizando o QR Code ou o link dele (RF);
h) utilizar a biblioteca Hammer.js para tratamento do multitoque (Requisito Não

Funcional - RNF);

i) suportar a funcionalidade de múltiplos toques com no máximo 12 toques (RNF).

Para o editor em geral:

a) utilizar o formato JavaScript Object Notation (JSON) para os *templates* (RNF);
b) trabalhar com mídias de imagem .png (RNF);
c) trabalhar com mídias de áudio .mp3 (RNF);
d) funcionar com plataforma web (RNF);
e) ser implementado utilizando HTML5 (RNF);
f) ser implementado utilizando CSS3 (RNF);
g) utilizar o *framework* Bootstrap para a criação da interface (RNF);
h) utilizar o *framework* AngularJS para a integração entre as *views* e os *controllers* (RNF);
i) ser implementado utilizando a linguagem de programação JavaScript (RNF).

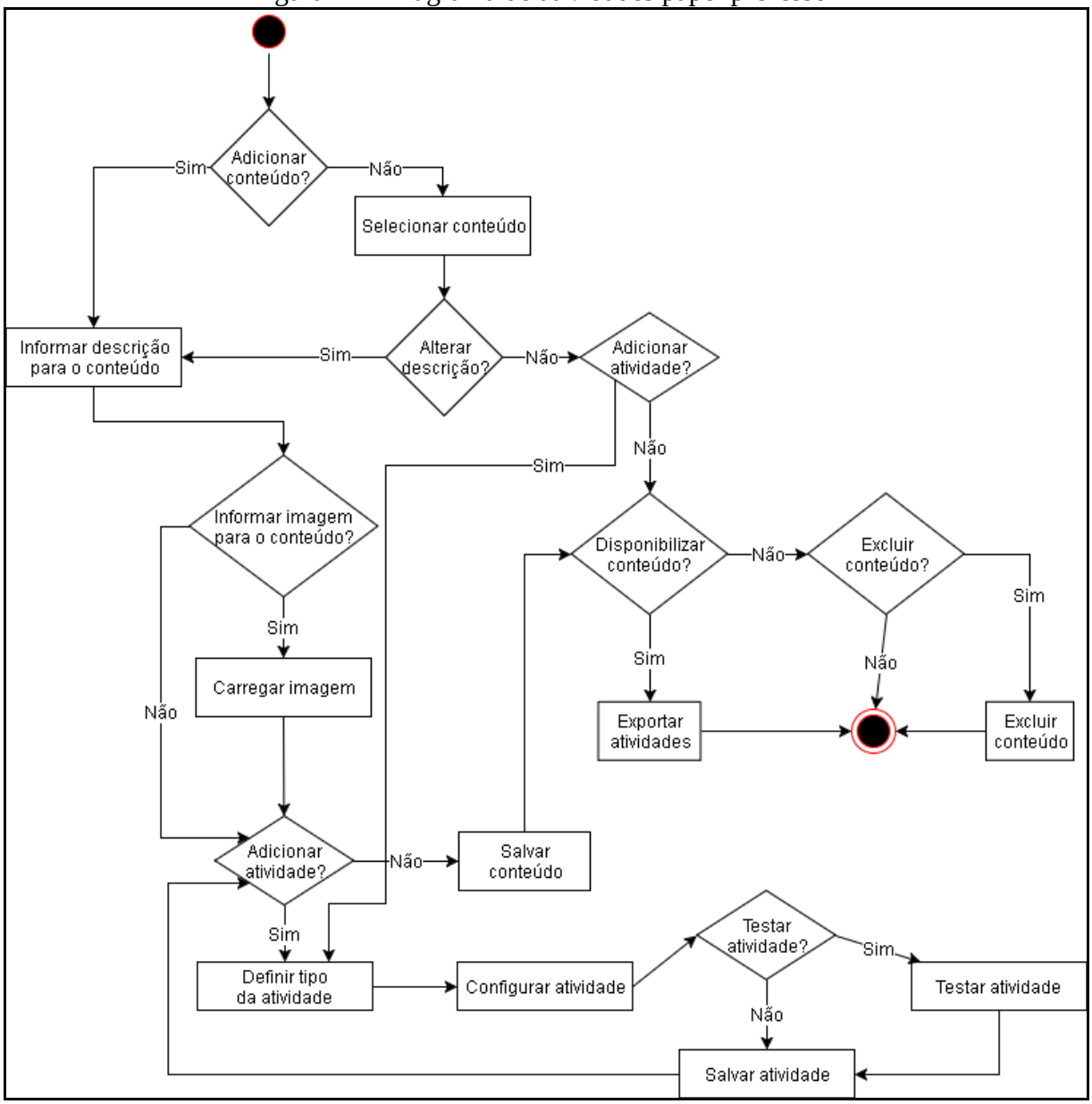
3.2 ESPECIFICAÇÃO

Para especificar esse projeto, foi feito uso de diagramas da Unified Modeling Language (UML), criados utilizando a ferramenta draw.io. Nas seções seguintes são apresentados o diagrama de atividades, de pacotes, de classes e o diagrama de arquitetura do trabalho e por fim, o Modelo de Dados.

3.2.1.1 Diagrama de atividades

Na Figura 12 é apresentado, de forma resumida, o diagrama de atividades das ações que o usuário com o papel de professor pode realizar no editor. As atividades básicas incluem criar, editar, disponibilizar e excluir um conteúdo, bem como, criar, customizar e testar as atividades.

Figura 12 - Diagrama de atividades papel professor



Fonte: elaborado pelo autor.

3.2.1.2 Diagrama de pacotes

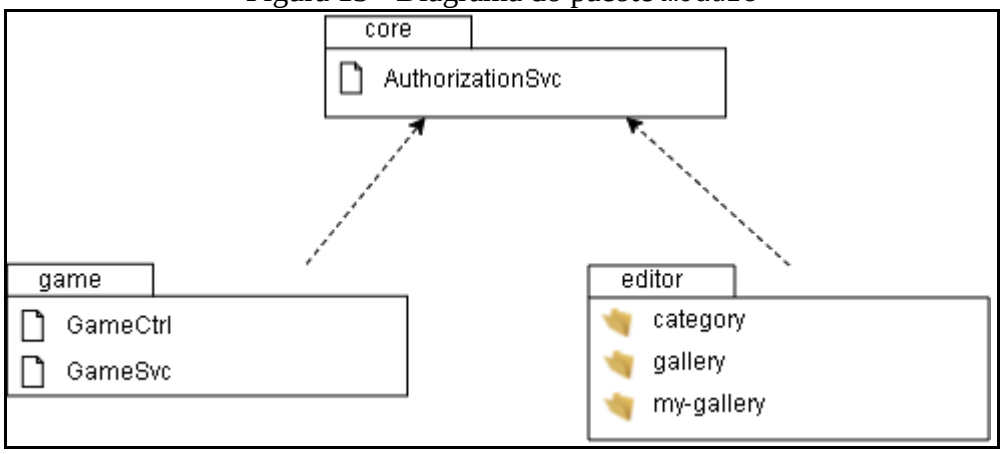
Nesta seção são apresentados os diagramas com os principais pacotes do projeto. A Figura 13 apresenta o diagrama do pacote `modules`. Este pacote está dividido em três subdiretórios `core`, `editor` e `game`. O módulo `core` armazena o serviço `AuthorizationSvc`, que possui as rotinas responsáveis por fazer a integração com o Google.

O módulo `editor` é divido em `category`, que possui as rotinas para criar, editar e manter os conteúdos e as atividades. Na pasta `gallery` são mantidas as rotinas para exibir os conteúdos padrão e o botão para adicionar um conteúdo. A pasta `my-gallery` é responsável

por manter as rotinas que exibem os conteúdos já criados pelo usuário. Ela é acessível por meio do item `Meu álbum` do menu.

No módulo `game` são mantidas as rotinas acessíveis pelo usuário com o papel de aluno. O pacote contém as funcionalidades responsáveis por exibir os conteúdos padrão, criar e fazer a gestão de uma partida, além do código para importar os dados do conteúdo de um arquivo JSON.

Figura 13 - Diagrama do pacote `module`



Fonte: elaborado pelo autor.

Na Figura 14 é ilustrado o pacote `components`. Para o AngularJS um módulo é composto por pequenos fragmentos de código que possuem comportamento genérico ou podem de alguma forma ser substituídos no futuro. Os fragmentos que possuem interface são conhecidos como componentes e os que não possuem como diretivas.

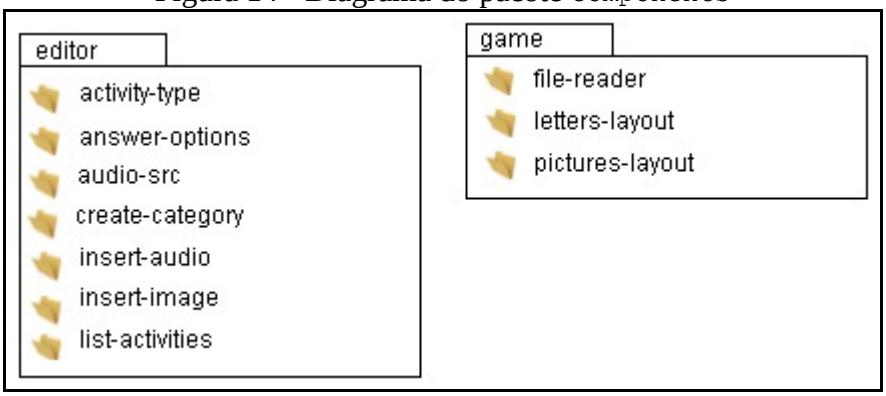
Nesse diretório são mantidos os componentes utilizados pelos módulos `editor` e `game`. Os componentes do módulo `editor` são compostos pelo `activity-type` que possibilita ao professor criar uma atividade do tipo letras ou imagens.

O `answer-options` é utilizado na criação das respostas das atividades. Ele possui os campos para informar o nome, a resposta, o nível de dificuldade, a dica e o tempo para resolução. O `audio-src` é uma diretiva utilizada para que o áudio adicionado a partir do Google Drive possa ser reproduzido utilizando o elemento `<audio>` do HTML5.

O `create-category` é o responsável pela criação do conteúdo. Nesse componente o usuário informará a descrição. O `insert-audio` é o componente utilizado pelo professor para inserir arquivos de áudio nas atividades, assim como o componente seguinte `insert-image` é utilizado para inserir imagens. O `list-activities` é responsável por listar as atividades de um conteúdo.

O pacote `game` possui a diretiva `file-reader` que é responsável por carregar os dados de um conteúdo de um arquivo JSON. O componente `letters-layout` é utilizado para criar o jogo das letras. O `pictures-layout` é encarregado de criar o jogo das imagens.

Figura 14 - Diagrama do pacote `components`



Fonte: elaborado pelo autor.

3.2.1.3 Diagrama de classes

Nesta seção são apresentados os diagramas de classes dos principais arquivos do projeto.

3.2.1.3.1 Diagrama de classes pacote `module.core`

O diagrama da classe `AuthorizationSvc` apresentado na Figura 15 possui os atributos e métodos responsáveis por possibilitar a integração com o Google Drive. As rotinas que criam o canal de comunicação e permitem a autenticação para obtenção da chave de acesso do usuário são mantidas nessa classe. Além das rotinas utilizadas para fazer a gestão dos arquivos, como: criar, atualizar, procurar e remover.

Figura 15 - Diagrama de classes pacote `module.core`

Fonte: elaborado pelo autor.

3.2.1.3.2 Diagrama de classes pacote `module.editor.category`

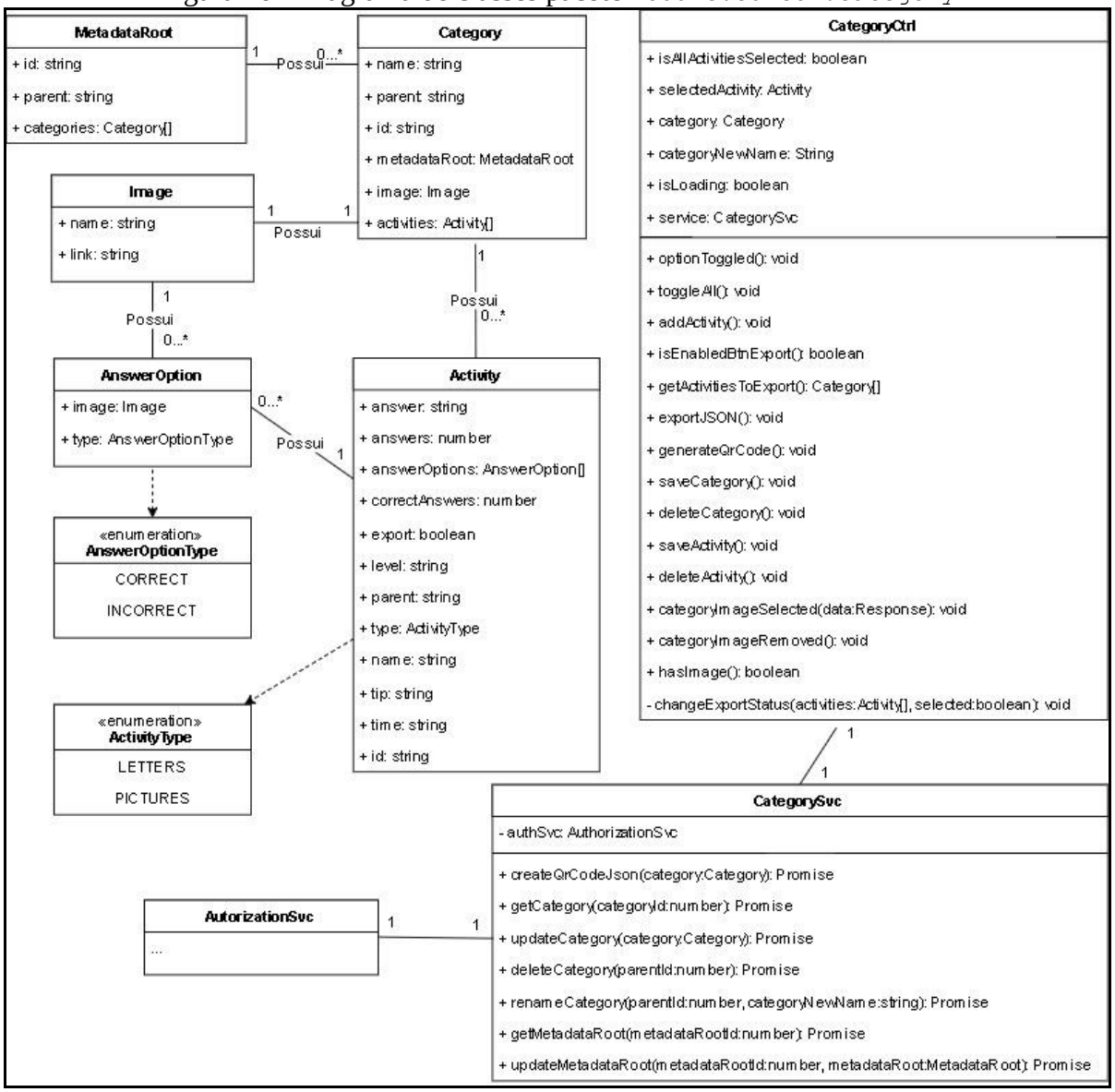
Na Figura 16 é possível observar o relacionamento entre as classes do editor. O arquivo de controle do EasyEdu, representado pela classe `MetadataRoot` possui uma lista

resumida da classe `Category`. Apenas com `name`, `id` e o objeto `image`. Essas informações são utilizadas para exibir o histórico do usuário no item do menu `Meu álbum`.

A classe `Category` possui uma lista de atividades, que são representadas pela classe `Activity`. Uma `Activity` possui um tipo presente na enumeração `ActivityType` e caso seja uma atividade do tipo imagens, terá uma lista de opções de resposta, que são representadas pela classe `AnswerOption`.

Uma opção de resposta (`AnswerOption`) é composta por um objeto da classe `Image` e a propriedade `type` representa a enumeração `AnswerOptionType`, que informa se a opção de resposta é correta ou incorreta. O *controller* desse módulo é o `CategoryCtrl`, que faz a ligação entre os dados no JavaScript e os dados apresentados na tela (`view`). O `CategorySvc` é quem faz a ligação entre o *controller* e o `AuthorizationSvc`.

Figura 16 - Diagrama de classes pacote `module.editor.category`

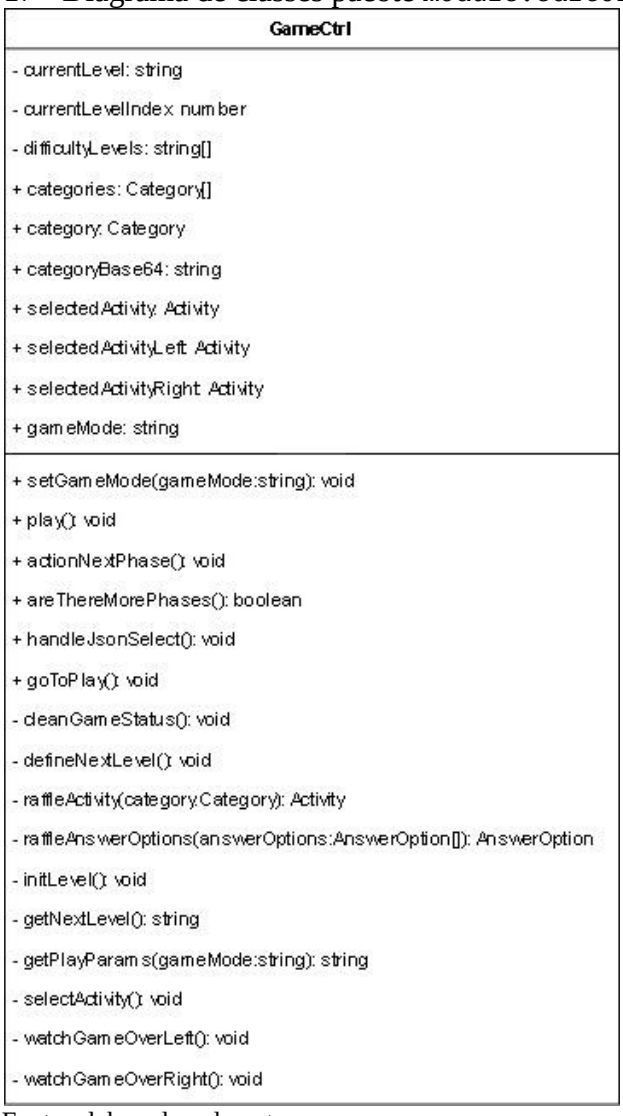


Fonte: elaborado pelo autor.

3.2.1.3.3 Diagrama de classes pacote `module.editor.game`

A Figura 17 ilustra o diagrama da principal classe do pacote `module.editor.game` `GameCtrl`. Do diagrama foram abstraídos métodos utilitários que não eram relevantes serem apresentados. Esse *controller* é responsável por carregar os dados de um conteúdo de um arquivo JSON, ou chamar o serviço do repositório público para obtê-lo. As rotinas deste arquivo permitem que o usuário defina se serão utilizadas uma ou duas partidas, sorteando as atividades e selecionando a quantidade necessária. A classe também faz o controle do nível do jogo e do status, a fim de identificar o ganhador ou o perdedor.

Figura 17 - Diagrama de classes pacote `module.editor.game`



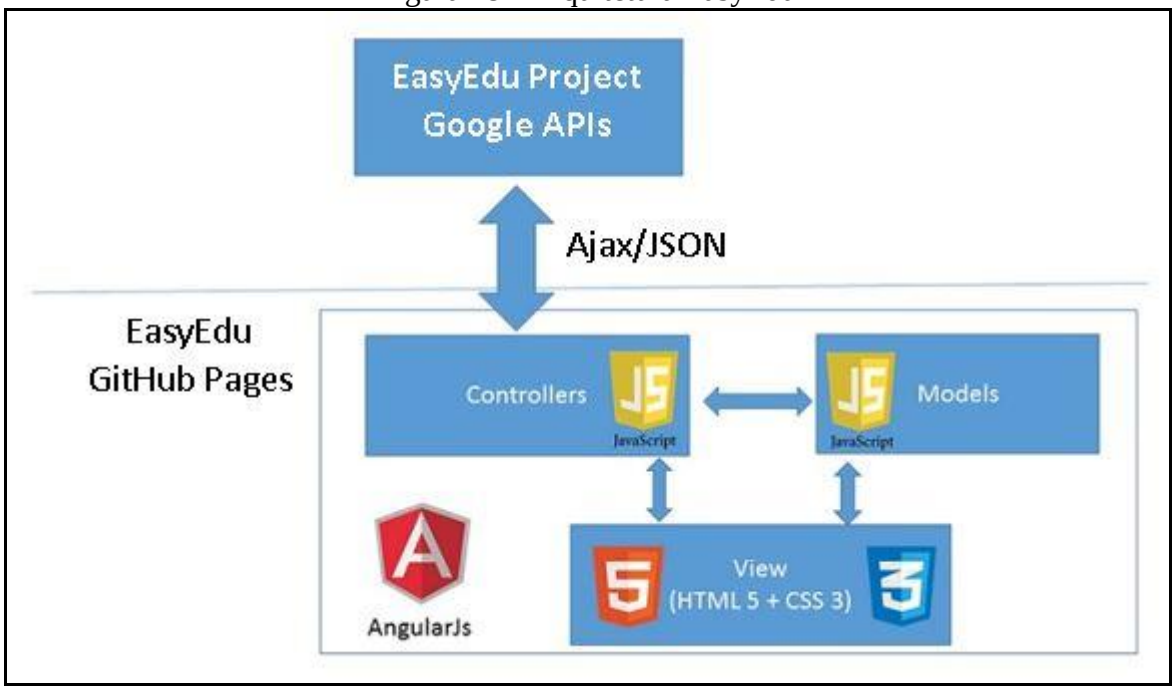
Fonte: elaborado pelo autor.

3.2.1.4 Diagrama de arquitetura

Para poder disponibilizar o EasyEdu na internet, a versão de distribuição com os arquivos estáticos HTML, CSS e JavaScript foi hospedada no GitHub Pages, permitindo criar

o site diretamente do repositório no GitHub. As rotinas em JavaScript se comunicam com as APIs do Google por meio de requisições Ajax utilizando arquivos JSON. Um esquema da arquitetura pode ser visualizado na Figura 18.

Figura 18 - Arquitetura EasyEdu

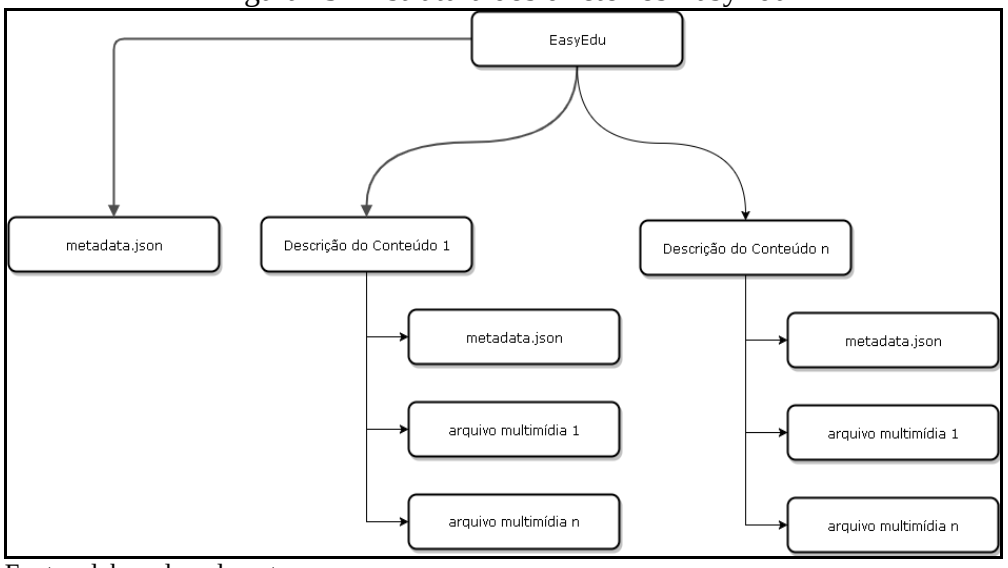


Fonte: elaborado pelo autor.

3.2.1.5 Modelo de Dados

Os dados gerados no editor web para jogos multitoque são estruturados em forma de árvore, na qual os nós sempre representam um diretório e as folhas os arquivos daquele diretório, como é possível observar na Figura 19. O arquivo de controle da pasta `EasyEdu` armazena uma lista com a referência para o arquivo de controle dos conteúdos criados. Para um conteúdo é criado um diretório onde são armazenados os recursos multimídia utilizados, já a lista de atividades é armazenada no `metadata.json`. A estrutura é formada por arquivos JSON e a persistência é realizada no Google Drive do usuário professor autenticado.

Figura 19 - Estrutura dos diretórios EasyEdu



Fonte: elaborado pelo autor.

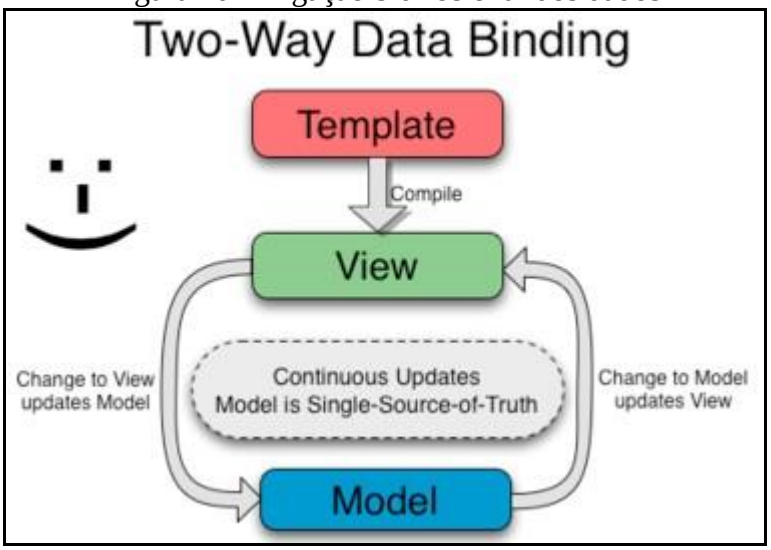
3.3 IMPLEMENTAÇÃO

A seguir são mostradas as técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

Para o desenvolvimento do editor web para jogos multitoque foi utilizado o AngularJS 1.4, que é um framework JavaScript ao qual permite a criação de *single-page applications* sob o padrão *model-view-view-model* (MVVM). O padrão também é conhecido como ligação bidirecional dos dados (*two-way data-binding*), pois possibilita que os valores carregados no *model* sejam exibidos na *view* e os valores alterados na *view* reflitam no *model*, assim como é apresentado na Figura 20.

Figura 20 – Ligação bidirecional dos dados



Fonte: AngularJS (2010).

Como base para o Cascading Style Sheets (CSS) foi utilizado o framework Bootstrap 3.3.7, que fornece estilos para formulários, tipografia, botões e componentes de interface. O jogo das letras e o das imagens foram desenvolvidos utilizando o Hammer.js (HAMMER.JS, 2017), que oferece recursos para a captura de toques na tela. O ambiente de desenvolvimento para a criação do trabalho foi o WebStorm 2017.1, com uma licença de estudante. Para auxiliar na depuração do código foi utilizado o Chrome Developer.

No versionamento dos códigos fontes foi utilizado o repositório no Bitbucket. Já os testes do editor web para jogos multitoque foram realizados utilizando o navegador Google Chrome, versão 58, com os seguintes dispositivos: tablet Samsung Galaxy Tab 2 P5110 Android 4.0 e tela com 10.1 polegadas (pol), Apple iPad Mini 2 Retina iOS 10.3 e tela com 7.9 polegadas (pol), Motorola Moto G2 Android 5.0 e tela com 5 polegadas (pol), Apple iPhone 5S iOS 10.3 e tela com 4 polegadas (pol).

3.3.1.1 Integração com o Google

Desde a concepção da ideia do editor, já se pensava na funcionalidade do usuário professor criar as atividades utilizando os arquivos multimídia de sua preferência. Com isso, identificou-se a necessidade de possuir um servidor com um serviço que recebesse os arquivos, armazenasse e retornasse um link ou um identificador permanente. Também seria necessário um serviço para obter um arquivo passando o identificador.

Para atender este cenário, foram estudadas alternativas como o Heroku, Azure da Microsoft e o S3 da Amazon, mas todos apresentaram algum empecilho. Em alguns casos o serviço não atendia alguma necessidade específica do projeto, em outros a curva de aprendizagem era muito acentuada e o tempo de implantação era significativo. A versão *free*,

disponível em algumas plataformas, possuía limitações ou o período de isenção era curto e na maioria dos casos o custo de adquirir uma licença era alto. Com a finalidade de não comprometer o desenvolvimento e o prazo do projeto, momentaneamente foi criada uma rotina simples em PHP que atendesse ao cenário.

Como os arquivos estavam sendo armazenados, surgiu a necessidade de fazer com que o usuário professor visse apenas o conteúdo criado por ele. Porém, como o objetivo do trabalho era entregar uma solução que contribuísse com o ensino, foi decidido não investir esforço em desenvolver rotinas de autenticação, entretanto, existia a necessidade de ter conhecimento de qual usuário professor estava operando o sistema naquele momento. Foi definido que seria utilizada a API disponibilizada pelo Google para acessar os seus recursos, entre eles a autenticação.

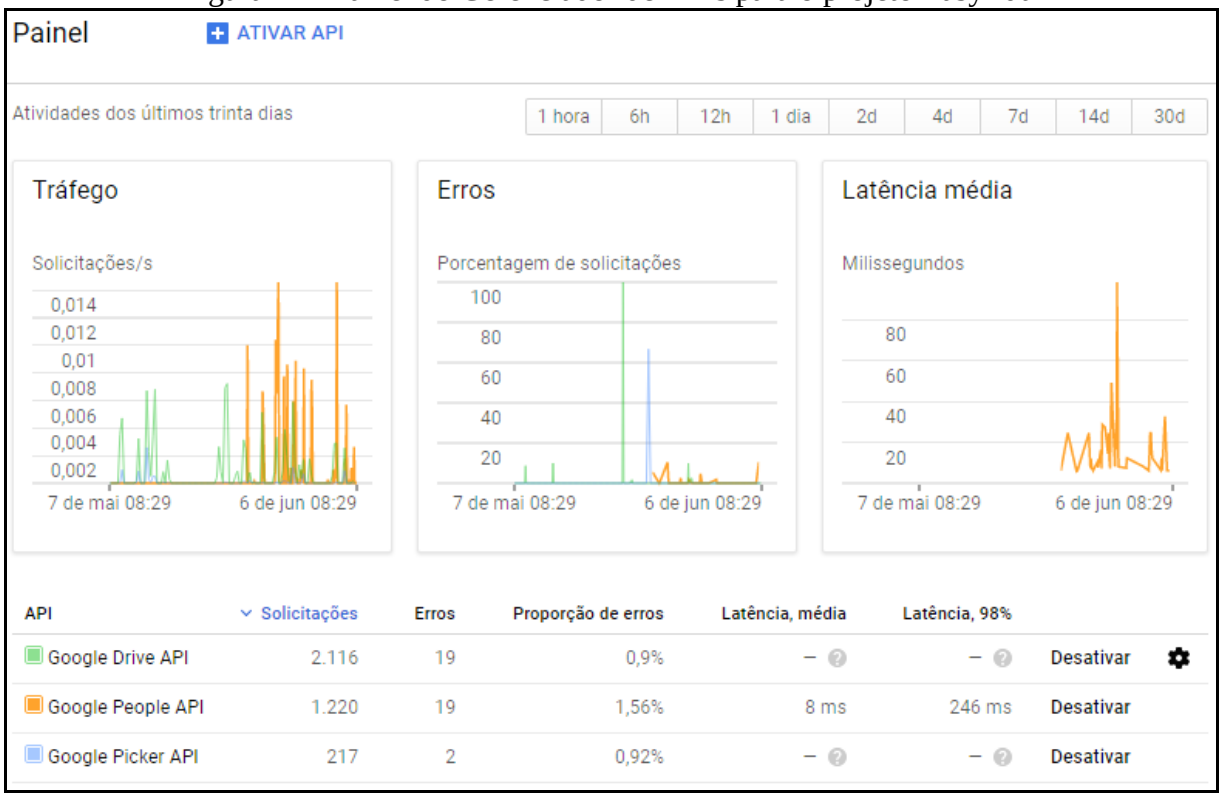
A integração com o Google possibilitou que os conteúdos, atividades e recursos de mídia fossem armazenados no serviço de nuvem Drive. Por padrão, todo o usuário que tenha uma conta no Google possui no mínimo 15GB de armazenamento on-line grátis (GOOGLE, 2017a). A utilização do Drive tornou desnecessária a criação, aquisição e manutenção de um servidor de arquivos e uma instância de banco de dados. Além disso, contribuiu para a experiência do usuário que utilizaria uma interface que já está habituado.

Apenas o arquivo criado ao gerar o QR Code não é armazenado no Drive, ele é disponibilizado em um repositório público chamado Myjson (MYJSON, 2017). O arquivo gerado precisa ser público, porque o usuário aluno não precisa estar autenticado para acessar, ou dependendo da faixa etária, ele nem possua uma conta de e-mail. Esse contorno foi necessário devido a uma restrição imposta pelo Google para hospedagem de arquivos estáticos, o que impossibilitava a obtenção do conteúdo por meio de uma requisição `GET` (G SUITE, 2015).

3.3.1.2 Arquitetura para utilizar os recursos do Google

Para utilizar os recursos do Google é preciso acessar o Gerenciador de APIs por meio do link https://console.developers.google.com, criar um projeto e ativar as APIs que se deseja consumir. Além de possibilitar que as APIs sejam ativadas ou desativadas, o gerenciador fornece ferramentas para monitorar o projeto, como é apresentado na Figura 21. No painel é possível ter acesso aos gráficos de consumo por API, como: tráfego, quantidade de erros e latência média. Também é possível extrair dados do Google Analytics, caso este esteja ativo. O período de exibição dos dados pode ser configurado.

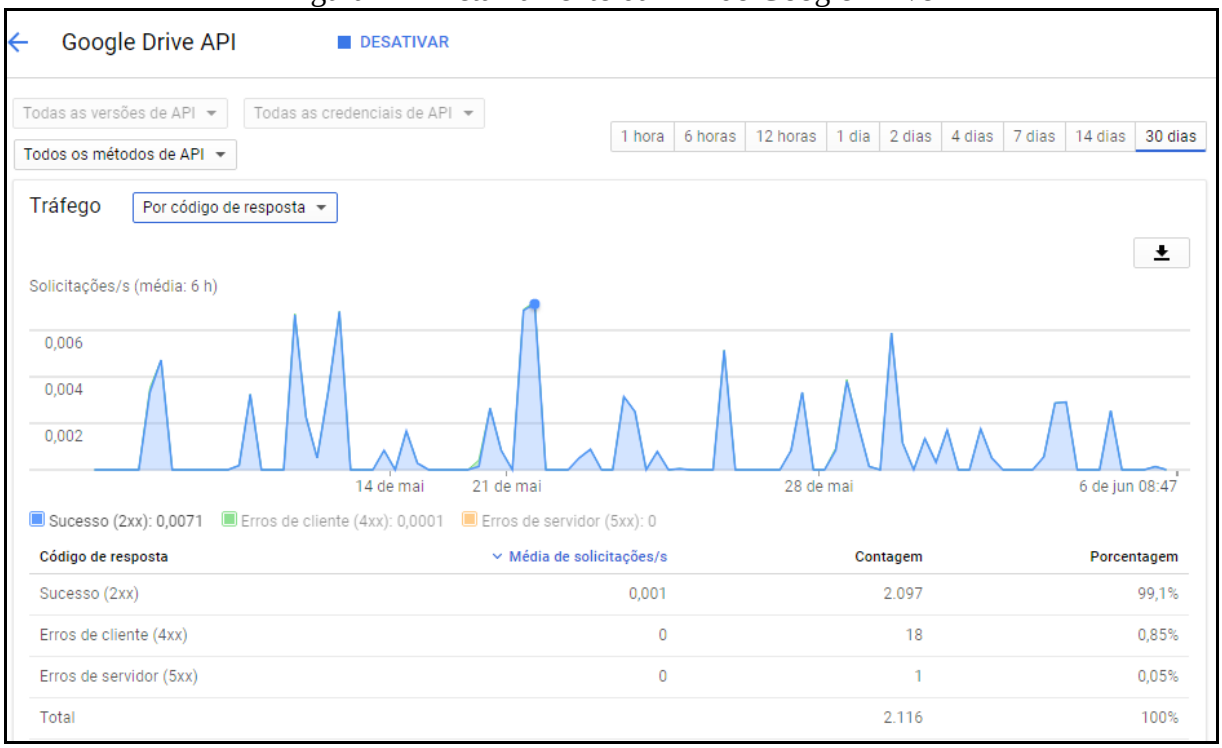
Figura 21 - Painel do Gerenciador de APIs para o projeto EasyEdu



Fonte: digitalizado pelo autor.

O gerenciador permite obter dados sobre uma API em específico, como é apresentado na Figura 22. No detalhamento é possível obter dados por versão da API, código de resposta ou sobre um método em particular. O período de exibição dos dados é configurável e também é possível fazer o download das informações apresentadas no formato `csv`.

Figura 22 - Detalhamento da API do Google Drive



Fonte: digitalizado pelo autor.

Para o projeto do editor foram habilitadas três APIs:

a) Google Drive API: permite gerenciar pastas e documentos no Google Drive (por exemplo, criar, atualizar, eliminar e modificar permissões e outros metadados, como o título);

b) Google Picker API: permite utilizar a mesma interface visual do Google Drive para gerenciar os arquivos;

c) Google People API: permite acessar as informações do perfil do usuário. É utilizada no projeto para pegar o nome e a foto do usuário professor que está autenticado.

Para que seja possível utilizar as APIs é necessário importar a biblioteca `client.js`, que provê os métodos para acessar os recursos no `index.html` do projeto. Conforme demonstrado no Quadro 1.

Quadro 1 – Importação a biblioteca client.js

```
01 <script type="text/javascript"
   src="https://apis.google.com/js/client.js"></script>
```

Fonte: elaborado pelo autor.

Após importar a biblioteca, é necessário inicializá-la informando as chaves recebidas quando o projeto foi criado no Google APIs. As chaves são necessárias para registrar a aplicação que irá consumir os recursos ativados. A inicialização da biblioteca `client.js` é demonstrada no Quadro 2.

Quadro 2 – Inicialização da API

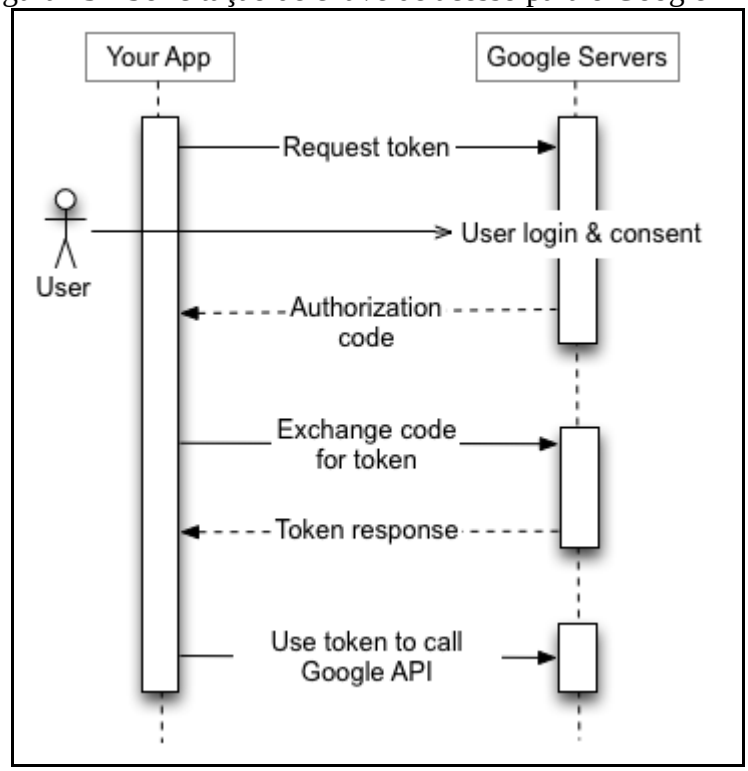
```
01 var API_KEY = 'AIzaSyBSklh1MWow4DDwhL1bna7vKA4LR1RmHQY';
02 var DISCOVERY_DOCS = [
03           "https://www.googleapis.com/discovery/v1/apis/drive/v3/rest",
04           "https://people.googleapis.com/$discovery/rest?version=v1"];
05 var CLIENT_ID = '661558756492-
06 p0agpbu1e13ac7npde96ts04mb6mv9o4.apps.googleusercontent.com';
07 var SCOPES =
08             "https://www.googleapis.com/auth/drive " +
09             "https://www.googleapis.com/auth/userinfo.profile";
10
11 function initClient() {
12       gapi.client.init({
13             apiKey: API_KEY,
14             discoveryDocs: DISCOVERY_DOCS,
15             clientId: CLIENT_ID,
16             scope: SCOPES
17       })
18       .then(success, error);
19 }
```

Fonte: elaborado pelo autor.

O valor representado pela `API_KEY` é o identificador único do projeto. O `DISCOVERY_DOCS` define quais são as APIs que estão sendo acessadas, qual a versão e outros parâmetros que o serviço possa necessitar (GOOGLE DEVELOPERS, 2017a).

Para autenticação e autorização do Google APIs, utiliza-se o protocolo OAuth 2.0, que é um framework que permite que um aplicativo de terceiro obtenha acesso limitado a serviços `HTTP`, em nome de um usuário mediante uma autorização (INTERNET ENGINEERING TASK FORCE, 2017). Para isso é utilizado o `CLIENT_ID`, que solicita uma chave de acesso para o Servidor de Autorização do Google, e a chave enviada no retorno dessa solicitação, é utilizada para acessar os recursos do Google APIs (GOOGLE DEVELOPERS, 2017b). O fluxo detalhado pode ser visualizado na Figura 23.

Figura 23 - Solicitação de chave de acesso para o Google APIs



Fonte: Google Developers (2017b).

Assim como o `CLIENT_ID`, a variável `SCOPES` também é utilizada para a autenticação e autorização com OAuth 2.0. Nessa variável são informadas quais as APIs que serão acessadas utilizando a chave de acesso fornecida. A inicialização é demonstrada no Quadro 3.

Quadro 3 – Inicialização da API de autenticação

```
01 ...
02 function initAuth() {
03       gapi.auth2.authorize(
04             {
05                   'client_id': CLIENT_ID,
06                   'scope': SCOPES,
07                   'immediate': false
08             },
09             handleAuthResult);
10 }
11
12 function handleAuthResult(authResult) {
13       if (authResult && !authResult.error) {
14             oauthToken = authResult.access_token;
15 ...
```

Fonte: elaborado pelo autor.

3.3.1.3 Persistência de dados

Ao criar o primeiro conteúdo, é verificado se já existe a estrutura do editor no Drive do usuário professor. Para essa verificação é utilizada a função apresentada no Quadro 4 da seguinte maneira: `searchFile("root", true, "EasyEdu")`. O valor `"root"` para o parâmetro `parentId` informa que a busca será feita na pasta "Meu Drive" do usuário

professor autenticado. Para o `isFolder` é definido o valor `true`, informando que o tipo do arquivo que está sendo buscado deve ser igual a `'application/vnd.google-apps.folder'` que corresponde a uma pasta do Google Drive (GOOGLE DEVELOPERS, 2017c). O último parâmetro é o nome do diretório raiz do editor, que no caso foi definido como `"EasyEdu"` e não é possível customizar.

Quadro 4 – Verificar a existência de arquivos no Drive

```
01 function searchFile(parentId, isFolder, fileName) {
02       var future = $q.defer();
03       var q = [];
04       if (parentId) {
05             q.push("'" + parentId + "' in parents");
06       }
07       q.push("mimeType " + (isFolder ? "=" : "!=") + "
08 'application/vnd.google-apps.folder'");
09       q.push("name = '" + fileName + "'");
10       gapi.client.drive.files.list({
11             q: q.join(" and "),
12             spaces: "drive",
13             pageSize: 1
14       }).then(success, error);
```

Fonte: elaborado pelo autor.

Quando o jogo é utilizado pela primeira vez, no modo editor não deve existir a pasta EasyEdu no diretório raiz do Drive. Deste modo a busca descrita anteriormente não retornará nenhum resultado. Então, o diretório raiz do projeto será criado utilizando a rotina apresentada no Quadro 5 da seguinte maneira: `createFile("EasyEdu", "", "root", "application/vnd.google-apps.folder")`. O primeiro parâmetro é o nome do arquivo que será criado, no caso o diretório raiz do editor `"EasyEdu"`. O segundo parâmetro é o conteúdo do arquivo, como se trata de uma pasta ele é vazio. O terceiro parâmetro é o identificador do diretório onde o arquivo será criado. É informado `"root"`, pois a pasta raiz do editor fica no "Meu Drive" do usuário professor autenticado. O último parâmetro é o tipo de arquivo criado.

Quadro 5 – Criar ou atualizar um arquivo no Drive

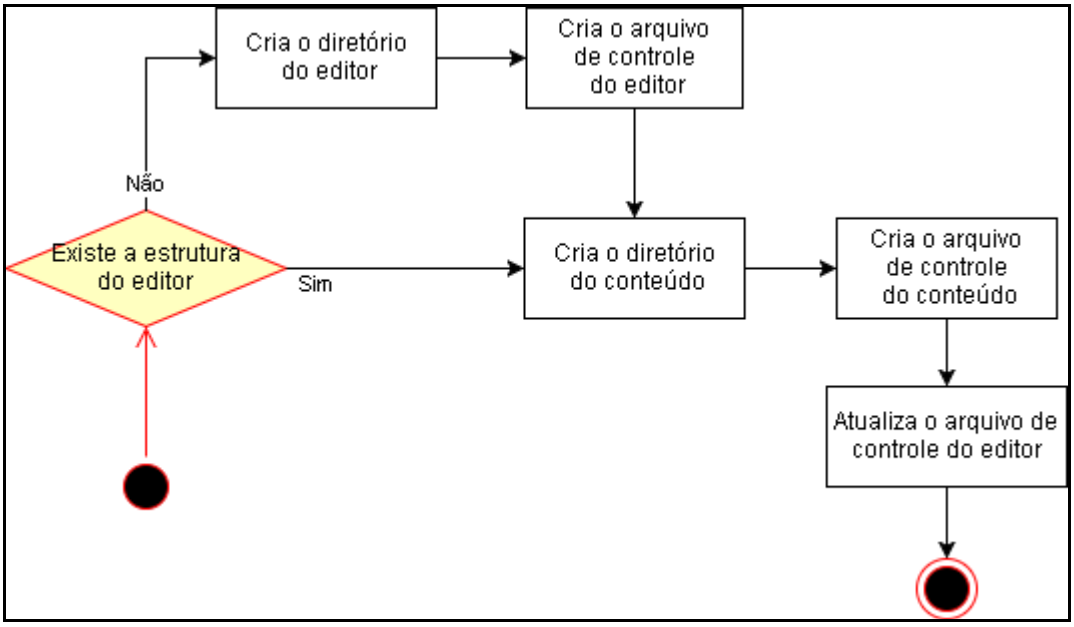
```
01 function createFile(fileName, fileData, parentId, mimeType) {
02       const boundary = '-------314159265358979323846';
03       const delimiter = "\r\n--" + boundary + "\r\n";
04       const close_delim = "\r\n--" + boundary + "--";
05       const contentType = "text/plain" || 'application/octet-stream';
06       var metadata = {};
07       metadata.name = fileName;
08       metadata.parents = [parentId]
09
10       var base64Data = btoa(fileData);
11       var multipartRequestBody =
12             delimiter +
13             'Content-Type: application/json\r\n\r\n' +
14             JSON.stringify(metadata) +
15             delimiter +
16             'Content-Type: ' + contentType + '\r\n' +
17             'Content-Transfer-Encoding: base64\r\n' +
18             '\r\n' +
19             base64Data +
20             close_delim;
21
22       gapi.client.request({
23               'path': '/upload/drive/v3/files',
24             'method': 'POST',
25             'params': {'uploadType': 'multipart'},
26             'headers': {
27             'Content-Type': 'multipart/mixed; boundary="' + boundary +
28 '"'
29             },
30             'body': multipartRequestBody
       }).then(success, error);
```

Fonte: elaborado pelo autor.

O próximo passo da inicialização é criar o arquivo de controle do editor, utilizando a função apresentada no Quadro 5 da seguinte maneira: `createFile("metadata.json", [], easyEduRoot, "application/json")`. O nome do arquivo de controle foi definido como `"metadata.json"` e o seu conteúdo inicialmente é uma lista vazia. O terceiro parâmetro é o id do diretório raiz do editor, obtido após a execução da rotina de criação. O último parâmetro é o tipo de arquivo criado, no caso um JSON.

As etapas da persistência, descritas até o momento, se repetem para a criação de um conteúdo. A diferença é que o local onde a pasta será criada não será o `"root"`, mas sim o diretório do editor. O mesmo ocorre para o arquivo de controle, que também possui o nome `"metadata.json"`, porém fica dentro da pasta do conteúdo. Após ter concluído a criação do conteúdo é necessário incluir o novo item no arquivo de controle do editor. Para facilitar o entendimento do fluxo de criação do diretório do EasyEdu, pode ser observada a ilustração da Figura 24.

Figura 24 – Diagrama de Atividades criar conteúdo



Fonte: elaborado pelo autor.

Para atualizar um arquivo é utilizada uma rotina muito semelhante à apresentada no Quadro 5. A única diferença é que na linha 24 ao invés do tipo do método ser `POST` ele será `PUT`, sendo que o parâmetro `fileData` não será uma lista vazia, mas sim uma lista com os itens.

Antes de obter o conteúdo do arquivo `metadata.json` é necessário verificar se existe a estrutura de diretórios do editor. Para isso, é utilizada a função `searchFile` apresentada no Quadro 4. A função `searchFile` retorna o id do arquivo caso ele exista, que é utilizado na função `getFile` do Quadro 6 para visualizar os dados. A mesma função é utilizada para obter os detalhes de um conteúdo.

Quadro 6 – Buscar um arquivo no Drive

```
01 function getFile(fileId) {
02       gapi.client.drive.files.get({
03             fileId: fileId,
04             alt: "media"
05       }).then(success, error);
```

Fonte: elaborado pelo autor.

Na tela de detalhamento, o usuário professor tem a opção de excluir o conteúdo, para isso é utilizada a função `deleteFile` do Quadro 7. Um exemplo do arquivo de controle do editor pode ser visualizado no Apêndice A no Quadro 26 e um exemplo do arquivo de controle de um conteúdo pode ser visualizado no Apêndice B no Quadro 27.

Quadro 7 – Deletar arquivo no Drive

```
01 function deleteFile(fileId) {
02       gapi.client.drive.files.delete({
03             'fileId': fileId
04       }).execute(success, error);
```

Fonte: elaborado pelo autor.

3.3.1.4 Principais funcionalidades do editor

No editor web para jogos multitoque, um grupo de atividades é chamado de conteúdo. É uma maneira de dividir e categorizar as atividades. Esse nome foi adotado, pois é o mesmo termo utilizado pelos professores. Por exemplo, na matéria de geografia o educador pode trabalhar o conteúdo “Estados brasileiros“. Dentro deste criar atividades para que os alunos possam exercitar e fixar o assunto.

Nas seções a seguir serão apresentadas as rotinas responsáveis por permitir que o professor possa interagir com o módulo do editor do EasyEdu. São abordadas as principais funcionalidades, como: criar um conteúdo, criar ou atualizar uma atividade, realizar o upload de arquivos para uma atividade e disponibilizar o conteúdo para os alunos.

3.3.1.4.1 Criar conteúdo

A rotina responsável por criar um conteúdo é apresentada no Quadro 8. O trecho de código já considera que a estrutura dos diretórios esteja criada. Na linha 3 é chamada a rotina para criar a pasta, o id retornado pela função é atribuído como diretório pai na linha 7. Em seguida na linha 8 é criado o arquivo de controle para o conteúdo e o id retornado é atribuído ao conteúdo na linha 12.

Na linha 13 é realizada a busca pelo arquivo de controle do editor e os dados são obtidos na linha 18. O novo conteúdo é adicionado ao `metadata.json` do editor na linha 21 e em seguida na linha 25 é feita a atualização do arquivo de controle. Um exemplo do arquivo de controle do editor pode ser visualizado no Apêndice A no Quadro 26 e um exemplo do arquivo de controle de um conteúdo pode ser visualizado no Apêndice B no Quadro 27.

Quadro 8 – Criar um conteúdo no editor

```
01 .then(function(rootFolder) {
02       vm.rootFolder = rootFolder;
03       return AuthorizationSvc.createFolder(vm.category.name,
04 rootFolder.id);
05 })
06 .then(function(categoryFolder) {
07       vm.category.parent = categoryFolder.id;
08       return AuthorizationSvc.createJson(METADATA_FILE_NAME,
09 vm.category, categoryFolder.id);
10 })
11 .then(function(categoryMetadata) {
12       vm.category.id = categoryMetadata.id;
13       return AuthorizationSvc.searchFile(vm.rootFolder.id,
14 "metadata.json");
15 })
16 .then(function(metadataRoot) {
17       vm.metadata = metadataRoot;
18       return AuthorizationSvc.getFile(vm.metadata.id);
19 })
20 .then(function(metadataContent) {
21       metadataContent.push({
22             id: vm.category.id,
23             name: vm.category.name
24       });
25       return AuthorizationSvc.updateJson(vm.metadata.id,
26 metadataContent, vm.rootFolder.id);
27 })
```

Fonte: elaborado pelo autor.

3.3.1.4.2 Criar ou atualizar atividade

A função apresentada no Quadro 9 é responsável pela criação ou atualização de uma atividade. Na linha 2 é verificado se a lista de atividades do conteúdo está definida, caso não esteja, é inicializada na linha 3. Se o usuário professor fizer uma edição em uma atividade já existente, ela consequentemente possuirá um id. Então a rotina de repetição da linha 8 realiza a busca da atividade que possua esse id e atribui o novo valor na linha 11.

Se for uma nova atividade é atribuído a ela um id na linha 16. A definição de um novo id é feita utilizando a função `valueOf()` de uma biblioteca Moment.js (MOMENT.JS, 2017) disponível no projeto. A função retorna uma estampa do tempo em milissegundos (WIKIPEDIA, 2017b) que garante que o id será único. Poderia ser utilizado um contador, mas caso o usuário voltasse a editar aquela atividade seria necessário percorrer a lista das atividades para saber qual foi o último id atribuído. Também agregaria complexidade, caso uma atividade fosse excluída.

Quadro 9 – Criar ou atualizar atividade

```
01 function saveActivity() {
02       if (!vm.category.activities) {
03             vm.category.activities = [];
04       }
05       if (vm.selectedActivity.id) {
06             var i = 0;
07             var length = vm.category.activities.length;
08             for (; i < length; i++) {
09                   var item = vm.category.activities[i];
10                   if (item.id === vm.selectedActivity.id) {
11                         vm.category.activities[i] = vm.selectedActivity;
12                         break;
13                   }
14             }
15       } else {
16             vm.selectedActivity.id = moment().valueOf();
17             vm.category.activities.push(vm.selectedActivity);
18       }
19       vm.selectedActivity = undefined;
20 }
```

Fonte: elaborado pelo autor.

3.3.1.4.3 Upload de arquivos multimídias para o Google Drive

O Google Picker é o gerenciador de informações armazenadas nos servidores do Google, permitindo que o usuário tenha acesso e possa incluir ou remover arquivos (GOOGLE DEVELOPERS, 2017d). No Quadro 10 é apresentada a função responsável por inicializar o gerenciador. Na linha 2 é verificado se a API do Google foi carregada e se o usuário está autenticado.

Para o editor foram configuradas duas visões. A primeira visão, configurada na linha 3, permite que o usuário professor visualize e selecione os arquivos. Os tipos permitidos são informados na linha 4, que podem variar dependendo do tipo de upload que está sendo realizado. Na linha 5 é definido que a visualização e a seleção dos arquivos serão relativas ao id informado, que no caso é o id da pasta do conteúdo. Na linha 7 é configurada a segunda visão que permite que o usuário possa inserir arquivos e na linha 8 é definido que a inserção também é relativa a pasta do conteúdo.

Para utilizar o Google Picker, é preciso criar um objeto do tipo `google.picker.PickerBuilder()`linha 10. Nas linhas seguintes são informadas as configurações para o gerenciador de arquivos. Para que seja possível acessar o Google API, o número do projeto que possui o Driver API ativado é informado no `setAppId`, o `setOAuthToken` recebe a chave de acesso fornecida para que o projeto possa chamar os serviços do Google API.

Na linha 13 é informada a chave de API gerada no Google APIs para o projeto. Nas linhas 14 e 15 são incluídas as visões criadas. Na linha 16 é informada a função de retorno que será invocada quando o upload dos arquivos for concluído. Para permitir que sejam selecionados mais de um arquivo é habilitada a funcionalidade `google.picker.Feature.MULTISELECT_ENABLED` na linha 20 (GOOGLE DEVELOPERS, 2017e).

Quadro 10 – Inicializar o Google Picker

```
01 function createPicker(parentId, callback, multipleSelect , viewTypes) {
02   if (gApiLoaded && oauthToken) {
03     var view = new google.picker.View(google.picker.ViewId.DOCS);
04     view.setMimeTypes(viewTypes || "image/png,image/jpeg,image/jpg");
05     view.setParent(parentId);
06
07     var uploadView = new google.picker.DocsUploadView();
08     uploadView.setParent(parentId);
09
10     var pickerBuilder = new google.picker.PickerBuilder()
11             .setAppId(APP_ID)
12             .setOAuthToken(oauthToken)
13             .setDeveloperKey(DEVELOPER_KEY)
14             .addView(view)
15             .addView(uploadView)
16             .setCallback(callback);
17
18     if (multipleSelect) {
19       pickerBuilder
20       .enableFeature(google.picker.Feature.MULTISELECT_ENABLED);
21     }
22
23     picker = pickerBuilder.build();
24     picker.setVisible(true);
25   }
26 }
```

Fonte: elaborado pelo autor.

Quando o usuário professor adiciona um arquivo na pasta, o Google Picker se encarrega de fazer o upload para o Google Drive. Após o término do processamento é chamada a rotina do Quadro 11, que anteriormente foi passada no parâmetro `callback`. Na linha 1 é verificada se a ação executada no Google Picker foi de seleção que corresponde à chave `google.picker.Action.PICKED`. Essa verificação é necessária, pois o usuário pode abrir o gerenciador, mas fechá-lo ou clicar no botão cancelar. A rotina apresentada no Quadro 11 é utilizada quando apenas um arquivo é selecionado. A única diferença da rotina para a seleção de múltiplos arquivos, é que ao invés de pegar o objeto do índice zero, são percorridos todos os itens do retorno e atribuídos a uma lista.

Quadro 11 – Processar uma inserção ou seleção de arquivo

```
01 if (data[google.picker.Response.ACTION]==google.picker.Action.PICKED)
02   {
03     var doc = data[google.picker.Response.DOCUMENTS][0];
04     vm.model.image.id = doc[google.picker.Document.ID];
05     vm.model.image.name = doc[google.picker.Document.NAME];
06   }
```

Fonte: elaborado pelo autor.

3.3.1.4.4 Disponibilizar conteúdo para os alunos

Após o usuário professor criar os conteúdos e as atividades que o compõem, ele deve de alguma forma disponibilizar esse conteúdo para os seus alunos. O editor web para jogos multitoque permite que isso seja feito por meio do download de um arquivo JSON, de um código QR Code ou do link para o conteúdo gerado.

Independente da opção escolhida pelo usuário, o código do Quadro 12 será executado. Essa rotina é responsável por verificar quais atividades serão exportadas, já que o editor permite que sejam selecionadas quais atividades daquele conteúdo serão disponibilizadas. A lista de atividades é filtrada utilizando a função `filter`, que retorna apenas os itens que tiverem o atributo `export` definido como `true`. As atividades filtradas são atribuídas a uma cópia do conteúdo e o objeto é retornado.

Quadro 12 – Retornar o conteúdo que será exportado

```
01 function getCategoryToExport () {
02  var selectedActivities = vm.category.activities.filter(function
03 (activity) {
04             return activity.export;
05           });
06  var category = angular.copy(vm.category);
07  category.activities = selectedActivities;
08  return category;
09 }
```

Fonte: elaborado pelo autor.

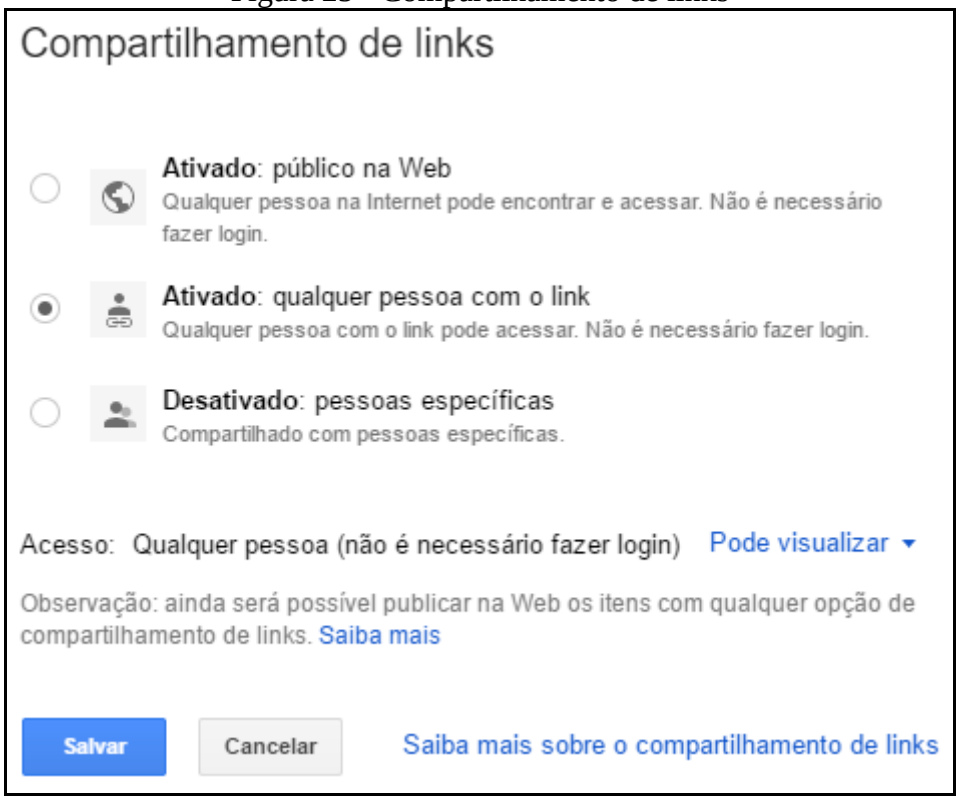
Como comentado na seção 3.3.1.1, os arquivos gerados no editor são armazenados no Google Drive. Isso faz com que quando o usuário professor optar por disponibilizar uma atividade seja necessário definir a permissão das imagens utilizadas como pública, para que elas sejam exibidas para os usuários. A alteração da permissão é feita pela rotina apresentada no Quadro 13, onde o id do arquivo cuja permissão deseja ser alterada é informado no parâmetro `fileId`. Para esse arquivo é definido que qualquer pessoa na internet que possuir o link poderá visualizar o arquivo. A alteração realizada pelo código é correspondente à configuração que pode ser realizada diretamente no arquivo no Google Drive, apresentada na Figura 25.

Quadro 13 – Definir imagem como pública

```
01  gapi.client.drive.permissions.create({
02                  'fileId': fileId,
03                  'resource': {
04                      "withLink": true,
05                      "role": "reader",
06                      "type": "anyone"
07                  }
08              }).then(success, error);
```

Fonte: elaborado pelo autor.

Figura 25 - Compartilhamento de links



Fonte: digitalizado pelo autor.

Para exportar o JSON é utilizado o código exposto no Quadro 14. Após concluir a definição da permissão das imagens como pública na linha 2, o objeto do conteúdo é codificado em Base64 na linha 4 e na linha 8 a `string` codificada é atribuída ao `href` de um elemento `<a>` que fica oculto no HTML. Feito isso é dado o nome ao arquivo, como sendo a descrição do conteúdo mais a extensão `.json` na linha 9. Na linha 10 é executado o evento do clique no elemento, que faz o download do arquivo.

Quadro 14 – Exportar JSON

```
...
 CategorySvc.setImagesPublic(category)
   .then(function () {
        var dataStr = "data:text/json;charset=utf-8," +
encodeURIComponent(JSON.stringify(category));
        var dlAnchorElem =
document.getElementById('downloadAnchorElem');
        dlAnchorElem.setAttribute("href", dataStr);
        dlAnchorElem.setAttribute("download", category.name + ".json");
        dlAnchorElem.click();
       })
...
```

Fonte: elaborado pelo autor.

Quando a opção escolhida para disponibilizar as atividades for o QR Code, são realizados os mesmos passos da exportação do JSON. É executado o filtro apresentado no Quadro 12 e a alteração da permissão das imagens do Quadro 13. Porém, conforme citado na seção 3.3.1.1, o JSON não é armazenado no Drive devido a uma restrição do Google. Como alternativa o objeto é enviado para um repositório público que permite que aplicações web ou mobile hospedem arquivos gratuitamente. O trecho de código exposto no Quadro 15 é responsável por enviar os dados para o repositório por meio de uma requisição `POST`. No retorno da requisição é recebida a `URL` para acessar o arquivo e obtido o id do arquivo utilizando a função `substr`.

Quadro 15 – Hospedar JSON no repositório público

```
...
$http.post(API_MYJSON, JSON.stringify(fileData),
          {
            contentType: "application/json; charset=utf-8",
            dataType: "json"
          })
          .then(function (response) {
              var data = response.data;
              if (data && data.uri) {
               var id = data.uri.substr(data.uri.lastIndexOf("/") + 1);
               future.resolve(id);
              }
          })
...
```

Fonte: elaborado pelo autor.

O id retornado pela rotina do Quadro 15 passa a ser conhecido como o id do conteúdo e é concatenado ao endereço do editor na linha 7 e 8 do Quadro 16. A variável `vm.qrCodeData` é utilizada no HTML para exibir o link do QR Code.

Quadro 16 – Gerar QR Code

```
01 ...
02  CategorySvc.setImagesPublic(category)
03    .then(function () {
04           return CategorySvc.createQrCodeJson(category);
05    })
06    .then(function (categoryId) {
07           vm.qrCodeData = window.location.origin +
08 window.location.pathname + "#/game/category?categoryId=" + categoryId;
09    })
10 ...
```

Fonte: elaborado pelo autor.

3.3.1.5 Principais funcionalidades do jogo

Nas seções a seguir serão apresentadas as rotinas responsáveis por permitir que o aluno possa interagir com o módulo do jogo do EasyEdu. Serão abordadas as principais funcionalidades, como: importar um conteúdo, criação e detecção de acertos de uma atividade do tipo letras e do tipo imagens. As rotinas responsáveis por permitir ao jogador avançar de fase, repetir a fase e jogar novamente.

3.3.1.5.1 Importar conteúdo

Para utilizar um conteúdo disponibilizado por meio de um QR Code, é feito apenas uma requisição `GET` para a API do Myjson (MYJSON, 2017), passando o id da seguinte forma: `$http.get(API_MYJSON + "/" + fileId)` e decodificando o retorno. Ao optar pela opção de exportar o JSON para disponibilizar as atividades, é necessário carregar esse arquivo para dentro do editor novamente.

Para isso foi criada a diretiva `game-file-reader`, apresentada no Quadro 17. A diretiva é utilizada em um elemento HTML da seguinte forma: `<input type="file" ng-model="vm.categoryBase64" game-file-reader/>`. Quando um arquivo for adicionado, a diretiva aguarda o upload ser concluído e depois mapeia o arquivo para um `FileReader`. Quando finalizar, o arquivo é retornado e atribuído ao `ng-model="vm.categoryBase64"`, utilizando o código da linha 5. Como o conteúdo do objeto foi codificado em Base64 é feito a decodificação e ele passa a ser utilizado no editor.

Quadro 17 – Importar JSON do conteúdo

```
01 ...
02  $q.all(slice.call(element.files, 0).map(readFile))
03    .then(function(values) {
04 ...
05           ngModel.$setViewValue(values.length ? values[0] : null);
06 ...
07    });
08
09  function readFile(file) {
10   var future = $q.defer();
11
12   var reader = new FileReader();
13   reader.onload = function(e) {
14       future.resolve(e.target.result);
15   };
16   reader.onerror = function(e) {
17       future.reject(e);
18   };
19   reader.readAsDataURL(file);
20   return future.promise;
21  }
22 });
```

Fonte: elaborado pelo autor.

3.3.1.5.2 Iniciar jogo

Para iniciar uma partida é utilizado o código apresentado no Quadro 18. Na linha 1 é definida a lista com os quatro níveis disponíveis para as atividades. Na linha 2 é a variável que armazena o nível atual da partida, na linha 3 o índice desse nível na lista. A função `play` linha 34, é primeira a ser executada e invoca a função `initLevel` da linha 5. Ela inicializa a variável `currentLevelIndex` com o valor 0, que representa o primeiro nível `"EASY"` atribuído à variável `currentLevel`. A próxima função a ser chamada é a `selectActivity`, que verifica se o modo de jogo é partida dupla, e caso seja, são sorteadas duas atividades, uma para a partida da esquerda e outra para a partida da direta. Caso seja o modo de jogo único, apenas uma atividade é sorteada.

A função `raffleActivity` usa todas as atividades do conteúdo e embaralha utilizando a biblioteca Lodash (LODASH, 2017). Após ter sorteado, é escolhida a atividade do índice zero, caso a atividade selecionada seja do tipo imagens, as opções de resposta também são embaralhadas.

Quadro 18 – Iniciar um jogo

```
01 var difficultyLevels = ["EASY", "MEDIUM", "HARD", "IMPOSSIBLE"];
02 var currentLevel;
03 var currentLevelIndex;
04
05 function initLevel() {
06   currentLevelIndex = 0;
07   currentLevel = difficultyLevels[currentLevelIndex];
08 }
09
10 function raffleActivity(category) {
11   var rafflesActivities = category.activities.filter(function (item) {
12     return item.level === currentLevel;
13   });
14   if (rafflesActivities && rafflesActivities.length) {
15     var activity = _.shuffle(angular.copy(rafflesActivities))[0];
16     if (activity.type === "PICTURES" && activity.answerOptions) {
17       activity.answerOptions = _.shuffle(activity.answerOptions);
18     }
19     return activity;
20   }
21
22   return undefined;
23 }
24
25 function selectActivity() {
26  if (vm.gameMode && vm.gameMode === "MULTIPLAYER") {
27      vm.selectedActivityLeft = raffleActivity(vm.category);
28      vm.selectedActivityRight = raffleActivity(vm.category);
29  } else {
30    vm.selectedActivity = raffleActivity(vm.category);
31  }
32 }
33
34 function play() {
35   initLevel();
36   selectActivity();
37 }
```

Fonte: elaborado pelo autor.

3.3.1.5.3 Criação e detecção de acertos atividade letras

Quando uma atividade do tipo letras é carregada no jogo, primeiramente a resposta é colocada em maiúscula utilizando a função `toUpperCase`. Depois a `string` da resposta é convertida em uma lista, utilizando a função `split("")`. Feito isso, para cada letra da resposta da atividade, é criado um elemento HTML `<li>` com um ponto de interrogação dentro. Esse elemento representará o local onde o aluno deverá arrastar a letra que julgará ser correta.

A letra é adicionada como uma classe de estilo, para que a rotina que verifica os acertos saiba qual letra aquele elemento `<li>` representa. Outra propriedade que também é adicionada é a `vm.customClass`, que é utilizada no modo de jogo partida dupla para informar se aquela letra é da partida da esquerda ou da direta, e assim um jogador não influenciar no

resultado do outro. Para o modo de jogo partida dupla os valores possíveis são `"left"` e `"right"`. No caso de modo de jogo partida única, o valor será `"single_player"`.

Os espaços existentes na resposta são representados por hifens. Para eles são criados elementos `<li>` que possuem as classes `answered` e `letter-answer-option`. Esse estilo é o mesmo aplicado para as opções de resposta e as letras já respondidas, dando a entender que mais nenhuma ação é necessária.

Para as opções de resposta sempre são exibidas 20 letras. Esse conjunto é composto pela quantidade de letras que a resposta da atividade possui, mais as letras escolhidas aleatoriamente. A forma e o estilo com que são apresentadas é o mesmo dos espaços e das letras já respondidas. A propriedade `vm.customClass` que representa o lado da partida também é atribuída. O diferencial das opções de resposta, é que para cada elemento é adicionado o comportamento de multitoque.

No Quadro 19 é apresentada a rotina responsável pelos passos descritos até este momento. Da linha 2 até a linha 10 são criados os elementos que recebem as respostas e os elementos que representam os espaços. Da linha 12 até a linha 18 são criados os elementos para as opções de resposta. Na linha 22 até a linha 25 é demonstrada a aplicação do Hammer.js, em que para cada opção de resposta é criado um *listener* para o tipo de movimento *pan* multidirecional. Também é dado o comando para que quando esse tipo de interação ocorra, a função `handleTouch` seja chamada. Esta função é responsável por validar a opção arrastada e será explicada mais adiante.

O movimento de *pan* que ocorre não é perceptível ao usuário, pois é apenas uma lista com coordenadas X e Y. O que faz com que o movimento seja perceptível é utilizar essas coordenadas para transladar o objeto de origem. No caso do EasyEdu, isso é feito utilizando o seguinte comando: `'transform':'translate('+event.deltaX + 'px,' + event.deltaY + 'px)'`.

Quando o usuário aluno finaliza um movimento de *pan*, a função que verifica o acerto é chamada. A rotina utiliza as coordenadas X e Y finais, para pegar sobre qual elemento HTML o movimento finalizou. O elemento é obtido utilizando o seguinte comando: `document.elementFromPoint(event.pointers[0].pageX, event.pointers[0].pageY);`. Com isso é possível saber em qual elemento originou-se o movimento e em qual ele foi concluído. Então é usado o valor do atributo `class` da origem e do destino e feita a comparação das classes de estilo que foram adicionadas anteriormente.

Quadro 19 – Criar jogo de letras

```
01 ...
02 var answerKeysLength = answerKeys.length;
03 for (var i = 0; i < answerKeysLength; i++) {
04      var answerKey = answerKeys[i];
05      var letterAnswerClass = answerKey === " " ? " answered letter-
06 answer-option" : " letter-answer";
07      var character = answerKey === " " ? "-" : "?";
08      answerKeysOut += "<li class='" + answerKey + " " + vm.customClass
09 + letterAnswerClass + "'>" + character + "</li>";
10 }
11 ...
12 var answerOptionsLength = answerOptions.length;
13 for (var i = 0; i < answerOptionsLength; i++) {
14      var answerOption = answerOptions[i];
15      answerOptionsOut +=
16                      "<li class='letter-answer-option " + answerOption +
17 " " + vm.customClass + "'>" + answerOption + "</li>";
18 }
19 ...
20 $("#" + NODE_ANSWER_OPTIONS + " > li.letter-answer-option")
21    .each(function () {
22       var mc = new Hammer.Manager(this, {});
23       mc.add(new Hammer.Pan({direction: Hammer.DIRECTION_ALL,
24 threshold: 0}));
25       mc.on("pan", handleTouch);
26    });
```

Fonte: elaborado pelo autor.

A primeira verificação a ser feita é se o destino, ou seja, onde o movimento finalizou é um local que recebe respostas, pois o usuário pode arrastar o objeto para cima de outro elemento. A segunda conferência realizada é se aquela posição já está respondida, fazendo com que, mesmo que o jogador arraste uma opção correta, ela não seja aceita. A terceira comparação é se a opção de resposta foi arrastada para uma posição do mesmo lado, para que as peças da esquerda não sejam aceitas na direita e vice-versa. Por fim, se todas as demais comparações forem verdadeiras, é verificado se a opção de resposta arrastada é equivalente à do destino, isto é, se ambos compartilham da mesma letra.

Conforme se as condições apresentadas acima sejam verdadeiras, o texto do elemento de origem é trocado pelo de destino. As classes de estilo `answered` e `letter-answer-option` são atribuídas ao elemento de destino. O elemento de origem é ocultado, utilizando a classe `hidden` e o último passo é verificar se o jogo acabou, verificando se ainda existem espaços a serem respondidos. O código correspondente aos passos descritos pode ser observado no Quadro 20.

Quadro 20 – Verificar acerto jogo de letras

```
01 function handleTouch(event) {
02  var dropEl = document.elementFromPoint(event.pointers[0].pageX,
03 event.pointers[0].pageY);
04  var elemClass = elem.getAttribute("class").split(" ");
05  var elemChecker = elemClass[1];
06  var elemSide = elemClass[2];
07
08  var dropElemClass = dropEl.getAttribute("class").split(" ");
09  var dropElemChecker = dropElemClass[0];
10  var dropElemSide = dropElemClass[1];
11  var dropElemValid = dropElemClass[2];
12  var dropElemAnswered = dropElemClass[3];
13
14  if (dropElemValid === "letter-answer" &&
15      dropElemAnswered !== "answered" &&
16      dropElemSide === elemSide &&
17      dropElemChecker === elemChecker) {
18         dropEl.innerText = elem.innerText;
19         dropEl.classList.add("answered");
20         dropEl.classList.add("letter-answer-option");
21         elem.classList.add("hidden");
22         checkWonGame();
23  }
24 }
```

Fonte: elaborado pelo autor.

3.3.1.5.4 Criação e detecção de acertos atividade imagens

O funcionamento do jogo de imagens é mais simples do que o de letras, porque nesse caso não é necessário criar um espaço para cada opção correta, basta criar um espaço maior onde apenas as opções corretas serão aceitas. O jogo de imagens tem como entrada uma lista de objetos. Cada objeto representa uma opção de resposta que possui em resumo duas propriedades importantes, o tipo que representa se aquela opção é correta ou incorreta e o id da imagem.

Para cada opção de resposta é criado um elemento HTML `<img>`, possibilitando que a imagem seja exibida. No atributo `src` é concatenado o id da imagem ao endereço padrão do Google Drive. No atributo `class` é adicionado o tipo da opção de resposta. Outra classe adicionada é a `vm.customClass`, que no modo de jogo partida dupla é utilizada para saber se aquela letra é do jogo da esquerda ou da direta, para que um jogador não influencie no resultado do outro. Para o modo de jogo partida dupla os valores possíveis são `"left"` e `"right"`. No caso de modo de jogo partida única, o valor será `"single_player"`. Após concluir a criação dos elementos são criadas as rotinas do Hammer.js, que permitem a captura dos eventos multitoque. A seção 3.3.1.5.3 possui mais detalhes sobre as funções do Hammer.js que foram utilizadas. O código detalhado acima referente à criação do jogo de imagens pode ser observado no Quadro 21.

Quadro 21 – Criar jogo de imagens

```
01 for (var i = 0; i < answerOptionsLength; i++) {
02      var answerOption = answerOptions[i];
03      img_out +=
04        "<div class='col-xs-4 col-sm-2'>" +
05          "<img" +
06          " src='https://drive.google.com/uc?export=view&id=" +
07 answerOption.image.id + "'" +
08          " class='picture " + answerOption.type + " " + vm.customClass
09 + " img-responsive'" +
10          " alt='" + answerOption.image.name + "'/>" +
11        "</div>";
12 }
13 ...
14 $(".picture").each(function () {
15      ...
16      // Rotina igual do jogo das letras
17 });
```

Fonte: elaborado pelo autor.

O espaço criado para receber as respostas corretas no jogo das imagens recebeu o identificador `"answers_" + vm.customClass`. Assim é possível controlar se a opção de resposta foi arrastada para o lugar certo, sendo esta a primeira verificação realizada quando o movimento de *pan* é finalizado. A segunda comparação é se a imagem da partida da direita foi arrastada para o espaço destinado às respostas corretas do mesmo lado. A última verificação é se a opção que foi arrastada possui o tipo `"CORRECT"` evitando que imagens incorretas sejam aceitas.

Caso o usuário arraste uma opção válida, uma cópia desse elemento será adicionada ao local destinado às respostas corretas. O elemento de origem será oculto por meio da atribuição da classe `hidden`. O contador de respostas corretas será incrementado em um. Ao final, a rotina que verifica se o jogo terminou será chamada, verificando se a quantidade de respostas corretas atual é igual ao total de respostas corretas existentes da atividade. As rotinas que verificam se a opção de resposta é válida pode ser observada no Quadro 22.

Quadro 22 – Verificar acerto jogo de imagens

```
01 var dropElemId = dropEl ? dropEl.getAttribute('id') : dropEl;
02
03 if (dropElemId && dropElemId === NODE_ANSWERS &&
04       dropElemId.endsWith(elemSide) &&
05       elemChecker === "CORRECT") {
06             dropEl.appendChild(angular.copy(elem.parentElement));
07             elem.classList.add("hidden");
08             vm.activity.answers += 1;
09             checkWonGame();
10 }
```

Fonte: elaborado pelo autor.

3.3.1.5.5 Próxima fase

Para avançar de fase é utilizado o código apresentado no Quadro 23. Na linha 15 é definido o nível da próxima fase, incrementando a variável de controle `currentLevelIndex`. Se existir um nível é chamada a função `selectActivity`, que buscará uma atividade que possua aquele nível. Após a chamada da função é verificado se as atividades foram definidas. Caso não tenham sido, a função `actionNextPhase` será chamada novamente até que uma fase seja encontrada, ou até que não haja mais níveis. Isso é feito porque o usuário pode ter cadastrado atividades sem seguir a sequência fácil, médio, difícil e impossível. Ao término da execução o status do jogo é zerado.

Quadro 23 – Definir próxima fase

```
01 function defineNextLevel() {
02   currentLevelIndex += 1;
03   currentLevel = difficultyLevels[currentLevelIndex];
04 }
05 function cleanGameStatus() {
06   vm.isWonMatch = false;
07   vm.isWonGame = false;
08   vm.isGameOver = false;
09   vm.isLeftWonMatch = false;
10   vm.isLeftWonGame = false;
11   vm.isRightWonMatch = false;
12   vm.isRightWonGame = false;
13 }
14 function actionNextPhase() {
15   defineNextLevel();
16   if (currentLevel) {
17       selectActivity();
18       if (vm.gameMode && vm.gameMode === "MULTIPLAYER") {
19           if (!vm.selectedActivityLeft && !vm.selectedActivityRight) {
20               actionNextPhase();
21           }
22       } else {
23           if (!vm.selectedActivity) {
24               actionNextPhase();
25           }
26       }
27       cleanGameStatus();
28   }
29 }
```

Fonte: elaborado pelo autor.

3.3.1.5.6 Recomeçar o jogo e repetir a partida

Ao término das fases, tanto no modo de jogo partida dupla quanto em partida única, é possível optar por recomeçar o jogo. Quando o modo de jogo é partida única, o usuário pode optar por repetir a fase que acabou de jogar. Para isso é utilizado o código exibido no Quadro 24. Caso o jogo tenha terminado, é chamada à função `play` que zera os níveis das atividades.

Caso o aluno queira jogar novamente a fase atual, o contador que armazena a quantidade de imagens corretas é zerado na linha 5. Esse comando apenas é aplicado ao jogo de imagens. Na linha 6 e 7 são limpos os espaços para inserir as respostas e as opções de repostas. O código da linha 8 embaralha novamente as opções de respostas (aplicado somente ao jogo de imagens). Ao final, o status da partida é zerado.

Quadro 24 – Recomeçar o jogo e repetir a partida

```
01 function actionPlayAgain() {
02   if (vm.isWonGame) {
03       vm.play();
04   } else {
05       vm.activity.answers = 0;
06       cleanChildNodes(NODE_ANSWERS);
07       cleanChildNodes(NODE_ANSWER_OPTIONS);
08       vm.activity.answerOptions = _.shuffle(vm.activity.answerOptions);
09       init();
10   }
11   vm.isWonMatch = false;
12   vm.isWonGame = false;
13   vm.isGameOver = false;
14 };
```

Fonte: elaborado pelo autor.

3.3.2 Operacionalidade da implementação

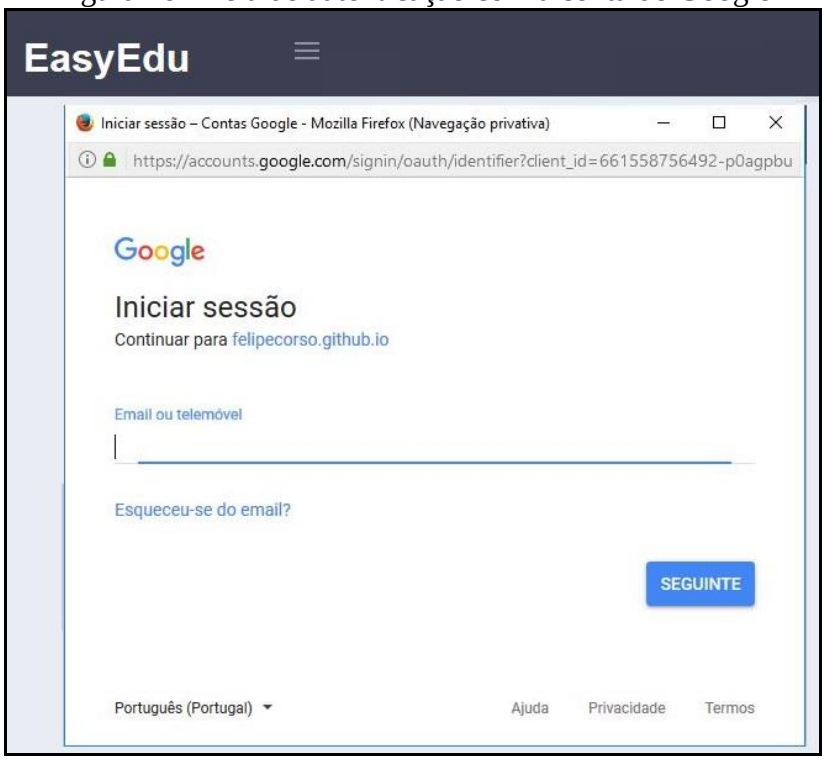
Esta subseção demonstra o funcionamento da implementação, por meio da utilização do editor web para jogos multitoque em nível de usuário. As subseções a seguir foram separadas pelas funcionalidades que os usuários com o papel de professor e aluno podem executar.

Para o professor são apresentados três menus. No menu `Galeria` é apresentado a opção para criar um novo conteúdo. Também são apresentados os conteúdos padrão que permitem serem usados como base para a customização. No menu `Meu álbum` são apresentados os conteúdos que o usuário autenticado já criou. O terceiro menu é um link que direciona para o módulo de jogo. No módulo de jogo são apresentados dois menus, o primeiro item é o próprio jogo, no qual o aluno poderá importar as atividades e no segundo item é fornecido o link para acessar o editor.

3.3.2.1 Autenticar com o Google

Ao acessar o EasyEdu no modo de edição é apresentada a tela ilustrada na Figura 26. Nessa tela o editor solicita que o professor faça a autenticação, utilizando a conta do Google. Esse procedimento é necessário para que os conteúdos e as atividades possam ser gravados no Google Drive. O usuário estando autenticado, o nome e a foto dele são apresentados no canto superior direito.

Figura 26 - Tela de autenticação com a conta do Google



Fonte: digitalizado pelo autor.

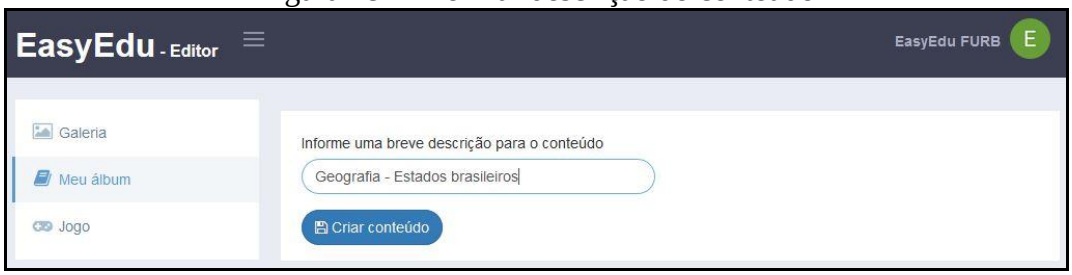
3.3.2.2 Criar conteúdo

Caso o usuário esteja autenticado, a opção que permite adicionar um conteúdo é habilitada. Para criar um novo conteúdo basta clicar na opção como ilustrado na Figura 27. Feito isso, o professor é direcionado para uma tela apresentada na Figura 28, na qual é possível informar uma breve descrição. O conteúdo criado ficará disponível no menu `Meu álbum` e pode ser editado clicando no item.

Figura 27 - Adicionar conteúdo

Fonte: digitalizado pelo autor.

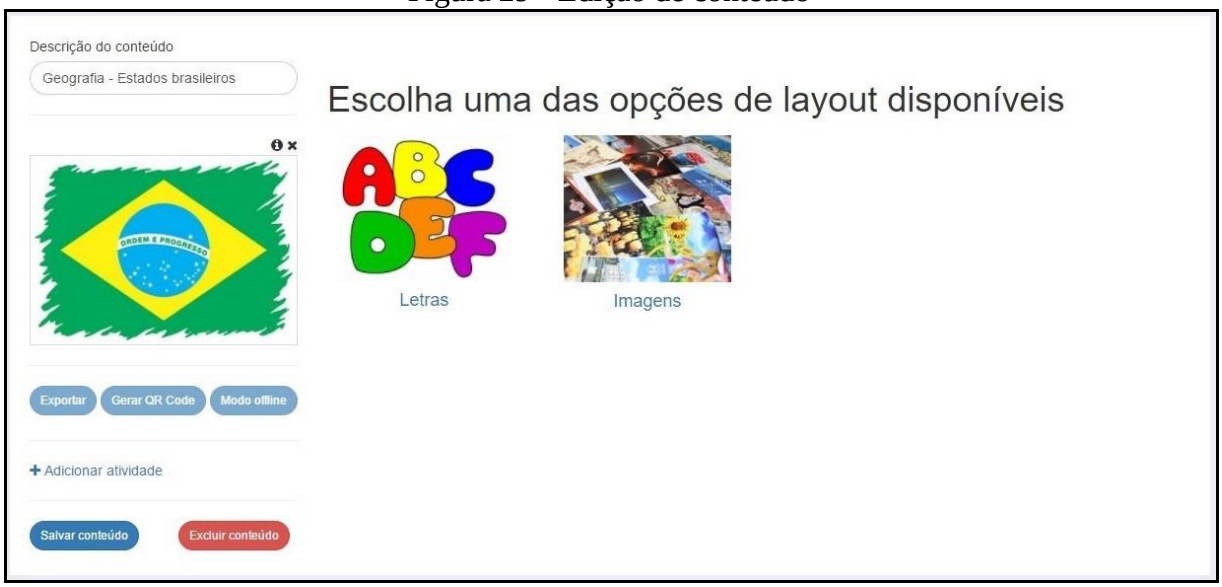
Figura 28 - Informar descrição do conteúdo



Fonte: digitalizado pelo autor.

Após ter criado o conteúdo, o usuário é direcionado para a tela exposta na Figura 29. Nessa tela é possível editar a descrição e adicionar uma imagem. Também é fornecida a opção de exclusão do conteúdo, que não pode ser desfeita. Ao clicar no botão `Adicionar atividade` são exibidas as opções de layout ilustradas na Figura 29.

Figura 29 - Edição de conteúdo



Fonte: digitalizado pelo autor.

3.3.2.3 Criar atividade tipo letras

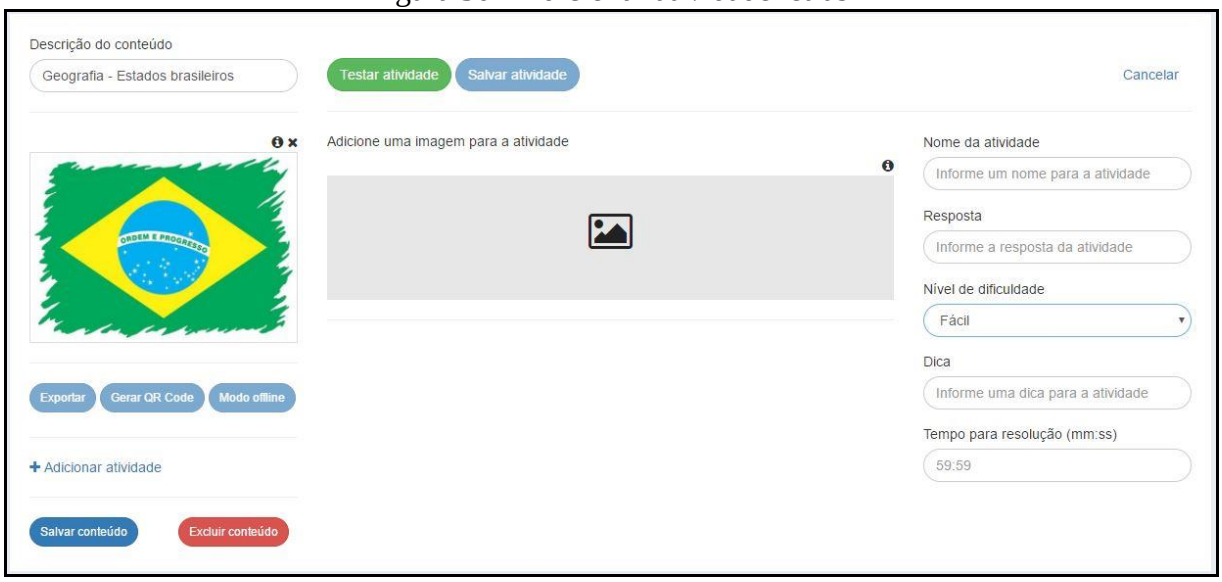
A atividade do tipo letras funciona de maneira semelhante ao jogo da forca. O objetivo consiste em o professor formular uma questão que esteja relacionada com o assunto estudado e passar para os alunos responderem. A tela utilizada para criar uma atividade do tipo letras é mostrada na Figura 30.

O editor permite adicionar uma imagem a questão, clicando na área em cinza. Para a atividade é necessário informar um nome que será utilizado para apresentar as atividades que compõem o conteúdo no canto inferior esquerdo. No campo `Resposta` deve ser informado o que se deseja que os alunos encontrem. A questão que se deseja ser respondida deve ser informada no campo `Dica` de forma sucinta. O professor pode optar por dificultar as atividades, conforme os alunos avançam nas fases. Para isso é utilizado o campo `Nível de`

`dificuldade` que possibilita definir qual o nível da atividade que está sendo criada ou editada. É possível definir um tempo máximo para a resolução da atividade. Para isso o valor deve ser preenchido no campo `Tempo para resolução`, no formato de minutos e segundos, ambos com dois dígitos. O tempo padrão para resolução da atividade é de 59:59.

Na parte superior é apresentado o botão `Testar atividade`, utilizado para que o professor possa visualizar como está ficando a atividade antes de disponibilizar para os alunos. O botão `Salvar atividade` adiciona a atividade ao conteúdo. O botão `Cancelar` descarta as alterações feitas. O botão `Excluir` apenas é apresentado se uma atividade for editada. Após concluir a criação ou edição de uma atividade é necessário clicar no botão `Salvar assunto` para que as alterações sejam enviadas para o Google Drive.

Figura 30 - Adicionar atividade letras



Fonte: digitalizado pelo autor.

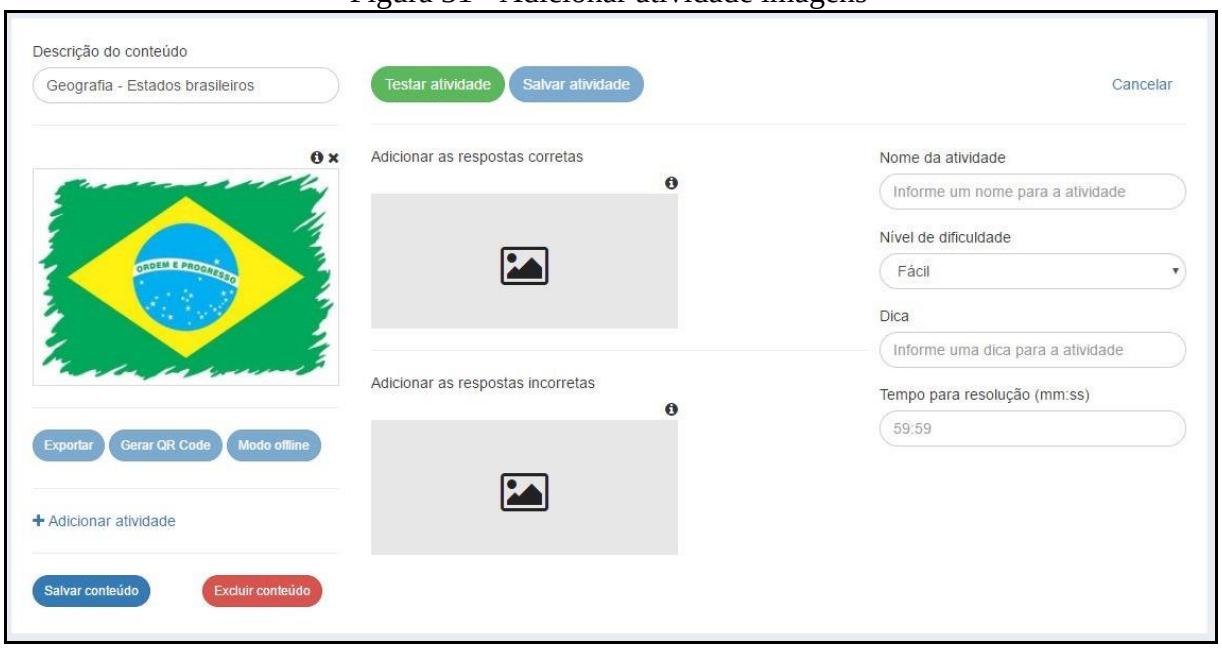
3.3.2.4 Criar atividade tipo imagens

Um outro layout disponibilizado é o de imagens. O objetivo desta atividade é que o professor crie uma sentença, ou informe características que possam definir uma imagem como correta. Feito isso o professor deve fornecer um conjunto de imagens contendo opções corretas e incorretas. Os alunos, por sua vez, terão que identificar naquele grupo quais opções atendem ao que foi solicitado.

Na Figura 31 é exibida a tela na qual é feita a configuração da atividade. A primeira área em cinza é onde as imagens corretas devem ser adicionadas. A área cinza apresentada abaixo é onde as imagens incorretas devem ser adicionas. Para listar as atividades do conteúdo é necessário que um valor seja informado no campo `Nome da atividade`. Em seguida é apresentado o campo para que seja informado o nível de dificuldade da atividade.

Para a atividade do tipo imagens o professor informará a pergunta que ele deseja que seja respondida no campo `Dica`. No próximo campo `Tempo para resolução` pode ser informado um tempo máximo para a resolução da atividade. O valor deverá ter o formato de minutos e segundos, ambos com dois dígitos. O tempo padrão para resolução da atividade é de 59:59. Após concluir a criação ou edição de uma atividade é necessário clicar no botão `Salvar assunto` para que as alterações sejam enviadas para o Google Drive.

Figura 31 - Adicionar atividade imagens

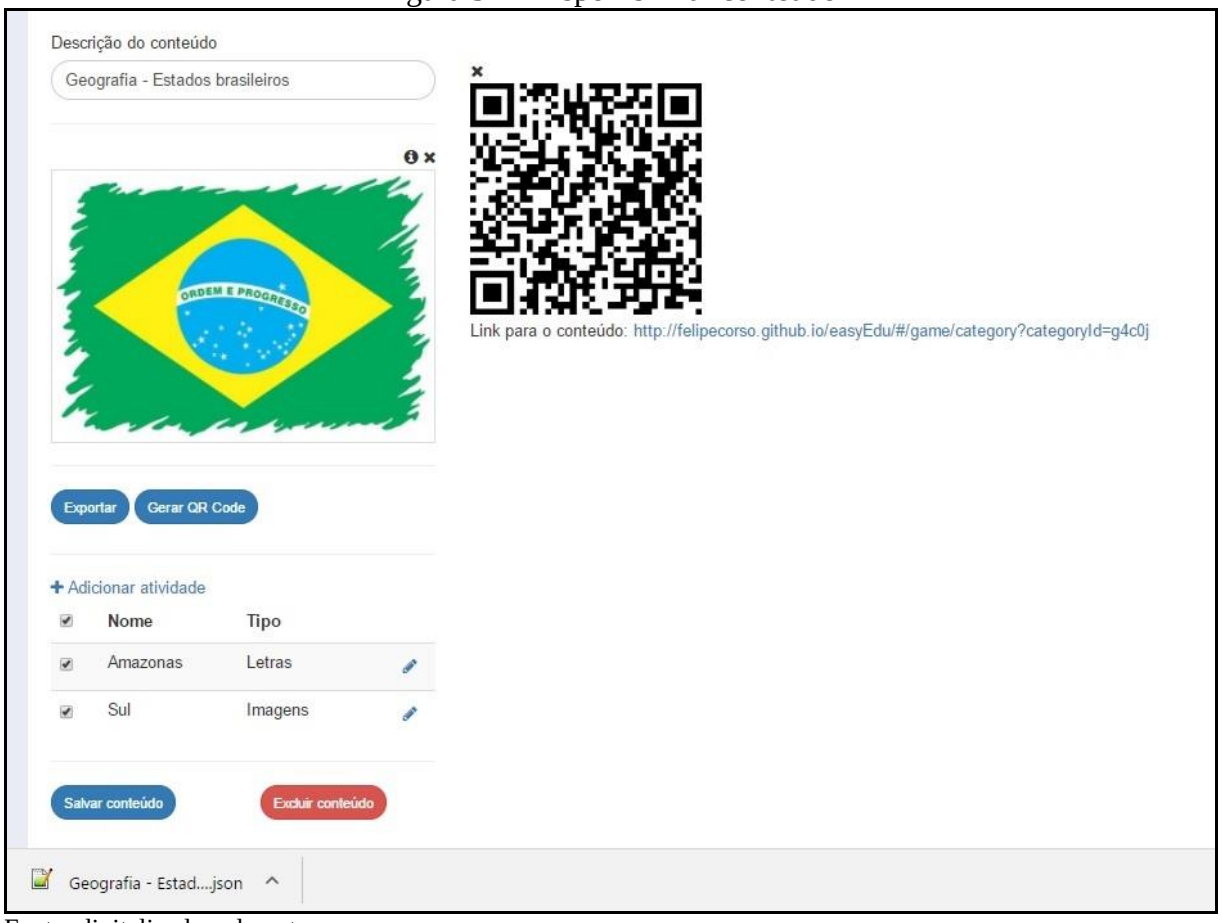


Fonte: digitalizado pelo autor.

3.3.2.5 Disponibilizar conteúdo

Para disponibilizar as atividades o professor deverá acessar a página `Meu álbum` e clicar no conteúdo desejado. Com isso ele será direcionado para a tela de edição do conteúdo apresentada na Figura 32. No canto inferior esquerdo são listadas as atividades existentes que podem ser selecionadas para serem disponibilizadas. A seleção é necessária, pois o editor permite que o professor crie inúmeras atividades para o conteúdo, mas as disponibilize gradativamente. Uma das maneiras de disponibilizar o conteúdo é por meio do botão `Exportar` que faz o download de um arquivo JSON, que posteriormente será importado no jogo pelos alunos. Outra forma é gerando um código de barras bidimensional, popularmente conhecido como QR Code. A imagem gerada pode ser salva por meio do clique da direita do mouse e clicando na opção “Salvar imagem como”. A imagem pode ser enviada para os alunos, que teriam acesso ao conteúdo utilizando um leitor de QR Code. O link contido no QR Code também é apresentado na tela e pode ser copiado e compartilhado.

Figura 32 - Disponibilizar conteúdo



Fonte: digitalizado pelo autor.

3.3.2.6 Jogo

Ao acessar o jogo com o papel de usuário aluno são apresentados os conteúdos padrão do EasyEdu. No canto superior direito é exibido o botão `Importar conteúdo`, que ao clicar nesse botão é aberto o explorador de arquivos do sistema operacional. Assim o usuário poderá navegar até o diretório do arquivo disponibilizado e abri-lo (Figura 33).

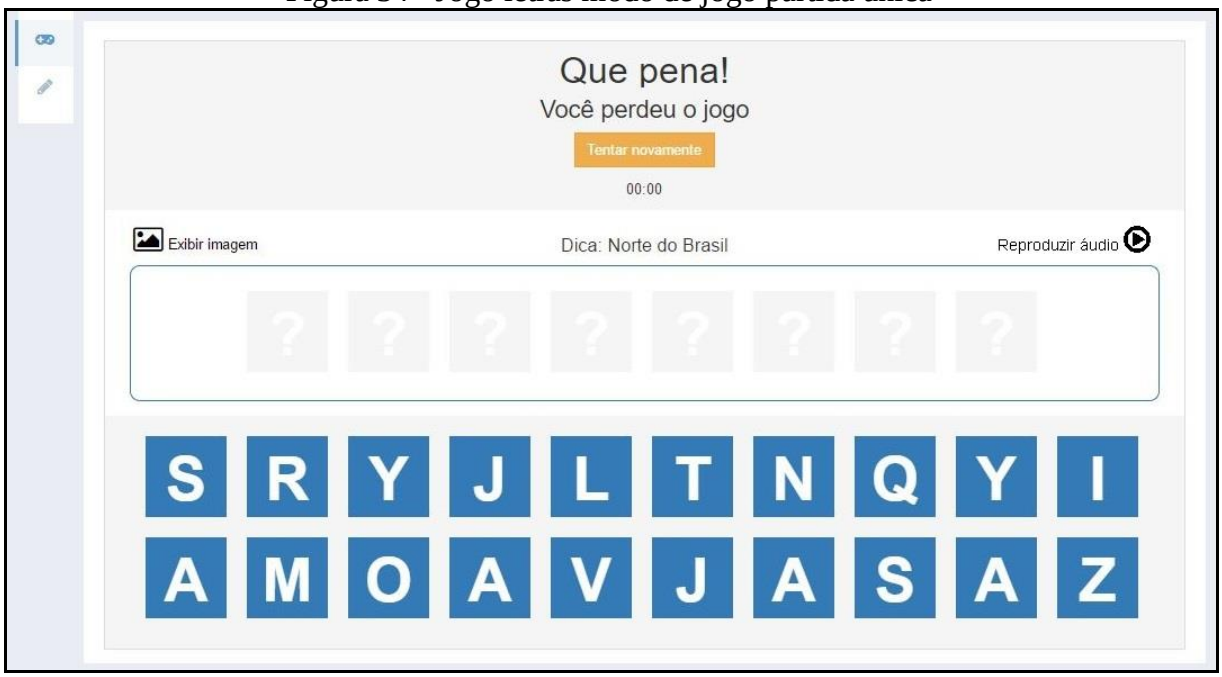
Figura 33 - Tela inicial jogo

Fonte: digitalizado pelo autor.

Caso o aluno tenha acessado o jogo por meio do QR Code ou o link para a atividade, o EasyEdu carregará o conteúdo chamando o serviço do Myjson (MYJSON, 2017), passando o id contido na URN. Nesta situação o aluno não precisa selecionar o conteúdo que ele deseja visualizar, pois será utilizado o conteúdo carregado. Desta forma o aluno é direcionado para a tela do modo de jogo para escolher se deseja criar uma ou duas partidas.

No modo de jogo partida única (Figura 34) ao finalizar uma atividade o usuário pode optar por repetir ou ir para a próxima fase, entretanto caso a atividade não seja resolvida no tempo estipulado, não é possível avançar de fase, apenas tentar novamente.

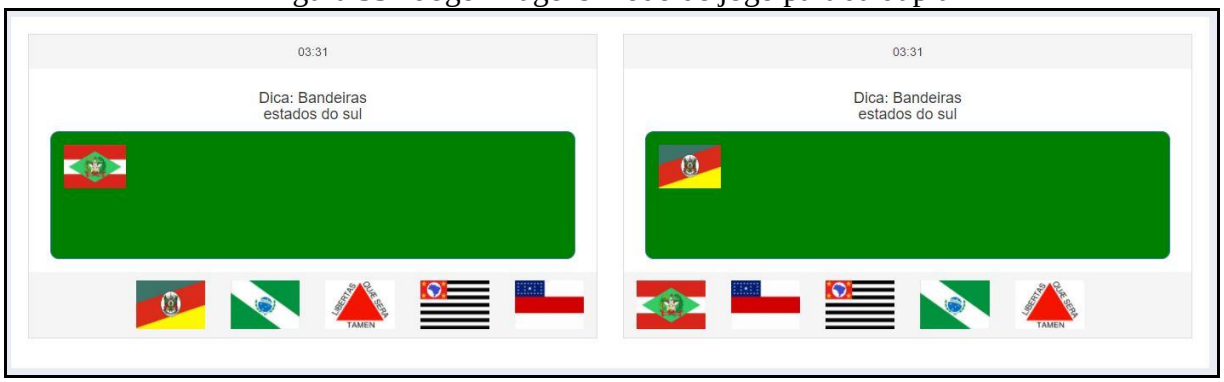
Figura 34 - Jogo letras modo de jogo partida única



Fonte: digitalizado pelo autor.

No modo de jogo partida dupla ilustrada pela Figura 35 não é possível repetir a fase atual, apenas ir para a próxima. Se a atividade não foi solucionada no tempo, será necessário começar novamente o jogo. Em ambos os tipos de partida, quando não existe uma próxima fase, é dada a opção de jogar novamente. Isso faz com que o jogo volte para o primeiro nível de dificuldade.

Figura 35 - Jogo imagens modo de jogo partida dupla



Fonte: digitalizado pelo autor.

3.4 ANÁLISE DOS RESULTADOS

Nesta seção é apresentada o experimento realizado com o editor web para jogos multitoque, EasyEdu, bem como o relato sobre as situações encontradas durante o desenvolvimento. Na seção 3.4.1 é descrita a metodologia do experimento realizado. Na seção 3.4.2 é feita a análise dos resultados obtidos com o experimento. A seção 3.4.3 apresenta algumas experiências obtidas no decorrer do desenvolvimento deste trabalho. Por fim na seção 3.4.4 é apresentada uma comparação entre os trabalhos correlatos e o editor desenvolvido.

3.4.1 Metodologia

O experimento aconteceu no dia doze do mês de junho de 2017. Por meio de testes realizados com dez usuários no laboratório LIFE da FURB, acompanhados pelo autor desde trabalho, junto com o seu orientador. Os usuários são acadêmicos da 7ª fase do curso de Pedagogia, da disciplina de Tecnologia Educacional e Aprendizagem do professor Mauricio Capobianco Lopes e se encaixam no perfil de especialistas. O experimento teve como objetivo avaliar a usabilidade e a aplicabilidade do editor como um complemento aos métodos tradicionais de ensino. O registro fotográfico do experimento pode ser consultado no Apêndice C na Figura 37 e na Figura 38.

Para os testes foram utilizados os equipamentos disponíveis no laboratório. Os notebooks possuem tela multitoque de 15 polegadas, com suporte de 10 toques simultâneos. Também foram realizados testes na mesa multitoque, que é detalhada na seção 2.3. Alguns usuários preferiram utilizar seus próprios dispositivos para realizar os testes.

Para o experimento foi aplicado o questionário disponível no Apêndice D, que está dividido em quatro partes. A primeira parte continha questões relacionadas ao perfil do usuário. A segunda seção guiava o usuário no editor com o papel de professor. As questões

apresentavam o ambiente, enquanto os conteúdos eram criados e as atividades incluídas. A última questão dessa etapa orientava o professor a disponibilizar o material criado para os alunos.

Na terceira parte do questionário o acesso ao EasyEdu era feito com o papel de aluno. A primeira questão mostrava como importar as atividades desenvolvidas anteriormente. Após isso, foi dado um tempo para que as atividades fossem praticadas. As questões seguintes tratavam sobre a experiência do jogo. Na última seção do questionário, era solicitado que o usuário opinasse sobre a percepção obtida ao utilizar o EasyEdu. As questões tratavam da usabilidade, aplicabilidade no dia a dia e sobre as dificuldades encontradas na utilização.

Junto ao questionário foram disponibilizados os arquivos para a criação do conteúdo e das atividades, a fim de que todos executassem a mesma tarefa, tornando o teste homogêneo.

3.4.2 Experimento com especialistas

O início do experimento deu-se com uma introdução sucinta do problema, o que motivou a realização do trabalho e o objetivo que se tentava alcançar. Depois foi realizada uma breve apresentação do editor web para jogos multitoque, com o papel de usuário professor e usuário aluno. Foram demonstrados superficialmente os passos que o questionário solicitaria e quais eram os resultados esperados. A próxima seção trata de analisar os resultados obtidos com o experimento, apresentando e interpretando os dados das quatro etapas do questionário.

3.4.2.1 Análise e interpretação do perfil dos usuários

Após a demonstração, iniciou-se a primeira parte das questões, que podem ser visualizadas no Apêndice D Figura 39 e Figura 40. Os resultados obtidos com as respostas são apresentados na Tabela 1. Com base nos dados é possível concluir que em sua maioria os usuários são formados pelo público feminino, com idade entre 21 a 25 anos e com o ensino superior incompleto.

Um dado relevante é que apenas 30% dos entrevistados sempre utilizam métodos diferenciados para o ensino. Não foi questionado o motivo da não utilização de outras técnicas. Já 90% deles possuem algum dispositivo com suporte à tecnologia multitoque. Os alunos de 60% dos entrevistados possuem acesso a algum dispositivo com tecnologia multitoque, 30% não possuem e para 10% a questão não era aplicável. Após a aplicação do teste identificou-se a necessidade de uma questão que esclarecesse quantos dos entrevistados lecionam diariamente e qual a faixa etária dos alunos.

Tabela 1 – Perfil dos usuários

Sexo	10% masculino 90% feminino
Idade	0% entre 11 a 15 anos 10% entre 16 a 20 anos 90% entre 21 a 25 anos 0% entre 26 a 30 anos 0% mais de 30 anos
Nível de escolaridade	0% ensino fundamental completo – 1º grau 0% ensino médio incompleto 90% ensino superior incompleto 10% ensino superior completo
Utilizam algum método, técnica ou abordagem diferenciada de ensino, como aulas expositivas ou recursos tecnológicos.	0% nunca utilizaram 70% as vezes 30% sempre utilizam
Possuem algum dispositivo que tenha suporte à tecnologia multitoque, como celular, tablet ou computador.	90% sim 10% não
Os alunos para que lecionam possuem acesso a algum dispositivo que tenha suporte à tecnologia multitoque, como celular, tablet ou computador.	60% sim 30% não 10% não se aplica

Fonte: elaborado pelo autor.

3.4.2.2 Análise dos resultados das instruções ao professor

A segunda parte do questionário continha instruções para o usuário com o papel de professor. As questões podem ser consultadas no Apêndice D Figura 41, Figura 42 e Figura 43.

Para realizar o teste e responder o questionário, era necessário que o usuário acessasse o editor utilizando a sua conta do Google. Caso algum usuário não possuísse uma conta, ou não se sentisse seguro ou confortável em informar os seus dados, ele poderia utilizar uma conta compartilhada. A conta criada teve os dados de acesso disponibilizados nas instruções.

Com base nos resultados apresentados na Tabela 2 é possível observar que todos os entrevistados conseguiram acessar o site do EasyEdu e se autenticar utilizando a conta do Google, pessoal ou compartilhada. Para as demais questões, apenas um usuário não conseguiu realizá-las. Este usuário estava usando o notebook pessoal e relatou que não estava conseguindo criar o conteúdo. O usuário tentou realizar o procedimento utilizando a conta particular e a conta compartilhada, mas o resultado foi o mesmo.

Ao término do teste, observou-se no Google Drive da conta compartilhada que o arquivo de controle do editor estava corrompido. O diretório e o arquivo de controle do conteúdo eram criados, porém, ocorria um erro no último passo, que é atualizar o arquivo de controle do editor. Não se sabe ao certo o que causou o problema, se foi uma particularidade da máquina pessoal ou o compartilhamento da conta.

As instruções fornecidas guiavam o usuário para que ao término dessa etapa ele tivesse criado um conteúdo com uma atividade de cada tipo. O conteúdo deveria se chamar “Geografia – Estados brasileiros”. A atividade de imagens seria do nível fácil e solicitaria que fossem selecionadas apenas as bandeiras dos estados do sul do Brasil. A atividade do nível médio seria de letras e o objetivo era selecionar as letras correspondentes ao nome do estado do Amazonas.

Tabela 2 - Respostas do questionário para as instruções ao professor

Acessar a página do editor por meio do link disponibilizado e informar o usuário e senha da conta do Google.	100% sim 0% não
Criar um assunto e informar a descrição solicitada.	90% sim 10% não
Adicionar uma imagem para o assunto criado.	90% sim 10% não
Criar uma atividade do tipo imagens, informando os campos obrigatórios.	90% sim 10% não
Criar uma atividade do tipo letras, informando os campos obrigatórios.	90% sim 10% não
Verificar na galeria do usuário se o assunto foi criado corretamente, exportar o arquivo com as atividades e gerar o QR Code.	90% sim 10% não

Fonte: elaborado pelo autor.

3.4.2.3 Análise dos resultados das instruções ao aluno

Na terceira parte do questionário foram apresentadas as questões direcionadas ao papel do usuário aluno. As questões podem ser observadas no Apêndice D Figura 44 e Figura 45. Com os resultados dessa etapa apresentados na Tabela 3, é possível observar que 90% dos usuários conseguiram importar o conteúdo utilizando o arquivo JSON ou acessando por meio do link do QR Code. Destes, todos conseguiram acessar as atividades e avançar nas fases. O único usuário que não conseguiu completar o questionário foi o que teve problemas de acesso, conforme descrito anteriormente.

Tabela 3 – Respostas do questionário para as instruções ao aluno

Acessar a página do jogo por meio do link no QR Code ou acessar o link disponibilizado e importar o arquivo com as atividades.	90% sim 10% não
Jogar e avançar nas fases até o término das atividades.	90% sim 10% não
Ao término do jogo, conseguiram jogar novamente indo para o início das atividades.	90% sim 10% não
Caso tenham perdido conseguiram repetir a fase utilizando o botão "Tentar novamente".	50% sim 0% não 50% não se aplica

Fonte: elaborado pelo autor.

3.4.2.4 Análise dos resultados relacionados às questões de usabilidade

A aplicação deste teste mostrou que é importante conhecer muito bem o perfil dos usuários para os quais está se desenvolvendo e também o nível de afinidade que eles possuem com a tecnologia. Por mais que o editor possa parecer simples aos olhos de quem é da área da tecnologia, para os usuários a execução de algumas ações precisou ser supervisionada. Ações simples como, por exemplo, extrair as imagens de um arquivo compactado.

Os relatos e as opiniões que foram dadas verbalmente ou por meio do questionário, deixaram claro que é necessário adotar uma prática comum nas empresas de software: os usuários integrantes do público alvo devem participar efetivamente do desenvolvimento. Essa ação dá a segurança, porém não a certeza, de que a entrega atenderá as necessidades do cliente, evitando que as funcionalidades precisem ser alteradas no decorrer do processo.

No início do desenvolvimento do editor havia sido definido que um grupo de atividades que estivessem dentro de um mesmo contexto, seria denominado categoria. Com o uso e consultas na literatura, percebeu-se que a nomenclatura não era adequada na sua totalidade. Então foi definido a utilização do termo assunto.

Após a aplicação do questionário uma das sugestões fornecidas pelos usuários foi a de alterar o termo assunto para conteúdo, mostrando mais uma vez a importância de ter o usuário sempre por perto. O ajuste sugerido foi aplicado, porém, para solucionar esse problema de terminologia definitivamente, poderia ser aplicado a internacionalização ao EasyEdu, criando um dicionário para cada idioma e permitindo a customização pelo usuário. Isso também permitiria que o editor fosse usado em outros idiomas.

Segundo o relato dos usuários, algo que incomodou bastante foi o fato do pop-up de autenticação ser bloqueado pelo Google Chrome. O ícone que informa que um pop-up foi bloqueado é bem discreto e pode deixar o usuário confuso sem saber o que fazer. Esse é um comportamento padrão do navegador e o usuário precisa explicitamente informar que ele confia naquele domínio e que as janelas devem sempre ser exibidas. Uma possível solução para esse problema seria incorporar a interface de autenticação do Google no editor. As observações feitas pelos usuários podem ser visualizadas na Figura 48 do Apêndice D.

O questionário encerrava com as questões de usabilidade, as quais podem ser consultadas no Apêndice D Figura 46 e Figura 47. Nos resultados da Tabela 4, 90% dos usuários acharam o EasyEdu intuitivo e fácil de usar e 80% deles conseguiram executar mais da metade das tarefas sem auxílio. Talvez com um tempo maior de adaptação e uso esse

número aumentaria, levando em consideração que os usuários nunca tinham visto o editor e em torno de uma hora e vinte minutos foram apresentados e passaram a utilizá-lo.

Dos usuários especialistas entrevistados, 100% deles assumiram que a utilização do EasyEdu como um método para apresentar conteúdos e desenvolver exercícios pode auxiliar na compreensão e na fixação do assunto abordado. Como complemento é apresentado o comentário que foi extraído do questionário e está presente na Figura 48 do Apêndice D: "Acredito que ferramentas deste tipo devam ser mais usadas em sala de aula, pois nós, pedagogos, estamos muito engessados nas práticas em sala, e atividades assim são um belo começo para as mudanças.". Estes fatos asseguram que objetivo inicial do trabalho de criar uma ferramenta diferenciada que auxiliasse no processo de ensino e aprendizagem foi alcançado.

Tabela 4 – Respostas do questionário para as questões de usabilidade

Quantas das atividades solicitadas quantas foram executadas sem auxílio.	30% todas 40% a maior parte 10% metade das tarefas 20% menos da metade das tarefas 0% nenhuma tarefa
Acharam o editor web intuitivo e fácil de usar.	90% sim 10% não
Acharam que a utilização do editor web para apresentar conteúdos e desenvolver exercícios, pode auxiliar na compreensão e na fixação do assunto abordado.	100% sim 0% não
Avaliaram o editor web para jogos multitoque como:	60% muito bom 40% bom 0% regular 0% insatisfatório

Fonte: elaborado pelo autor.

3.4.3 Análise geral sobre o desenvolvimento do editor web para jogos multitoque

Esta seção foi dedicada a apresentar algumas experiências obtidas no decorrer do desenvolvimento deste trabalho.

No início do projeto, definiu-se que as atividades disponibilizadas pelo EasyEdu seguiriam a mesma linha de alguns exemplos de jogos criados nas aulas de Sistemas Multimídias. Os jogos permitiam arrastar e soltar elementos (*drag and drop*) e foram feitos para a mesa multitoque, explorando este que é o seu principal recurso. A possibilidade de testar esses jogos contribuiu para validar a ideia antes de iniciar o desenvolvimento.

Na especificação do jogo, identificou-se a necessidade de possuir um algoritmo que trabalhasse com a colisão de dois elementos. Situação que ocorreria quando o aluno arrastasse uma opção de resposta para dentro do espaço criado para receber a resposta correta. Para não

investir tempo desenvolvendo esse tipo de rotina, buscou-se uma alternativa já pronta que pudesse otimizar o tempo de desenvolvimento.

Dentre as opções, foi optado pela utilização da biblioteca Phaser.io que é um framework para desenvolvimento de jogos em HTML5 para navegadores desktop e mobile, suportando a renderização com Canvas e WebGL (PHASER, 2017). O Phaser.io é muito completo, possuindo uma infinidade de funcionalidades para trabalhar com jogos. Rotinas até complexas demais para o simples problema que se estava tentando resolver. O ditado popular "usar uma bazuca para matar uma formiga" coube bem nessa situação.

A curva de aprendizagem é bem acentuada, mesmo o framework tendo certo nível de abstração. Muito tempo de programação foi envolvido para atingir um resultado minimamente aceitável. A versão inicial do jogo de letras chegou a ser desenvolvida utilizando o Phaser.io, porém, com o passar do tempo percebeu-se que mantendo essa abordagem não seria possível entregar o que havia sido prometido. Então voltaram os esforços em pesquisar uma forma de desenvolver jogos que permitissem *drag and drop*, dessem suporte ao multitoque e funcionassem em desktops e dispositivos móveis.

Assim a opção foi desenvolver nativamente e essa decisão foi motivada pelo fato de que os navegadores já ofereciam suporte ao *drag and drop* do HTML5. Essa funcionalidade é basicamente a que os jogos do EasyEdu utilizariam. Uma versão do jogo das imagens chegou a ser desenvolvida utilizando os recursos nativos do HTML5, que no final possuía o comportamento esperado. Entretanto, ao efetuar testes em dispositivos móveis identificou-se que este recurso ainda não era suportado nesses navegadores (CAN I USE, 2017)**.**

Para fazer com que o comportamento nos dispositivos móveis fosse o mesmo, uma das alternativas foi utilizar uma biblioteca criada por um brasileiro e disponibilizada no GitHub (DRAGDROPTOUCH, 2016). A biblioteca cumpriu o que prometeu, mas apenas um toque por vez era processado, o que comprometia o atingimento do objetivo do trabalho de ser multitoque.

Na proposta desenvolvida na disciplina de TCCI, foi colocado como Requisito Não Funcional a utilização do Hammer.js para tratamento do multitoque. Depois de não obter sucesso com a utilização do Phaser.io e do desenvolvimento nativo, uma prova de conceito foi realizada utilizando o Hammer.js. O êxito na utilização da biblioteca possibilitou que as atividades já criadas fossem migradas para o novo método. Caso o RNF da proposta tivesse sido seguido desde o início, provavelmente mais layouts poderiam ter sido entregues. Como o jogo da memória e o quebra-cabeça que havia sido pensado inicialmente. Talvez assim tivesse um tempo para realizar outro experimento.

Uma das funcionalidades mais interessantes implementadas, foi a integração com o Google, mais especificamente utilizando os recursos do Drive. Esse recurso agregou valor ao EasyEdu e fez dele um diferencial. Em vários sites é possível o usuário se autenticar com a conta do Google, mas não são muitos que utilizam o Drive como servidor de arquivos e banco de dados, assim como faz o EasyEdu.

A documentação das APIs do Google é bem completa, mas carece de exemplos de uso. Durante o desenvolvimento, teve-se a impressão que integrações de aplicações web com o Google Drive não são muito comuns, pois nem em fóruns de programação como o Stack Overflow, por exemplo, existiam questões abertas. Alguns dos problemas encontrados foram solucionados consultando o código de aplicativos em Java desenvolvidos nativamente para Android e criando rotinas equivalentes em JavaScript.

Dessa integração apenas o item de acentuação não foi implementado. O método `POST` cria o arquivo de controle com o encode `UTF-8`, porém quando a descrição de um conteúdo criado contém acentos, o arquivo é criado com encode `ASCII`. Após criar o metadata do conteúdo, o arquivo de controle do editor é atualizado e acaba tendo o seu encode alterado para `ASCII`, corrompendo os dados existentes. A rotina para criar e atualizar um arquivo é a mesma, a única diferença é o tipo de chamada `HTTP`. Até o determinado momento não foi encontrada uma solução para o problema.

Outro problema encontrado com a utilização do Google Drive foi o acesso a um arquivo com conteúdo quando a opção de compartilhamento escolhida fosse o QR Code. Essa situação é descrita com mais detalhes na seção 3.3.1.1. Antes de adotar como alternativa a utilização do repositório público Myjson (MYJSON, 2017) foi tentado gerar o QR Code com os dados do objeto do conteúdo. A solução funcionava até certo ponto, porém, para arquivos com mais dados o leitor do dispositivo não conseguia reconhecer o código gerado.

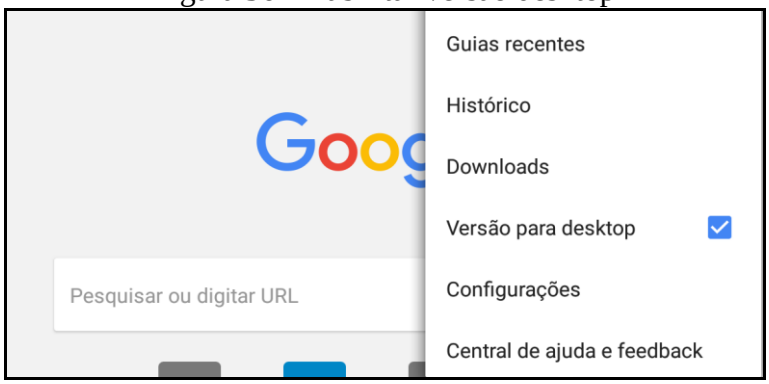
A última situação identificada nos testes e que também foi relatada no experimento, ocorre em dispositivos nos quais a altura da tela é muito menor em relação à largura. Essa condição faz com que a barra de rolagem vertical seja exibida, e para visualizar as opções de resposta, é necessário rolar a página para poder arrastar a opção. A ação de ter que movimentar a página, faz com que mesmo a opção de resposta estando correta, ela não fique no local.

Esse problema está relacionado com o uso da função `document.elementFromPoint(x, y)` para pegar o elemento onde o movimento de `pan` foi finalizado. A função considera as coordenadas do conteúdo que está sendo exibido no momento do início do movimento (MDN, 2017)**.** Caso a tela seja mexida, as coordenadas não

correspondem mais ao estado inicial e a rotina não conseguirá pegar o elemento onde o movimento foi finalizado. Seria necessário encontrar uma função que considerasse as coordenadas x e y da página inteira, não apenas do conteúdo que está sendo exibido. Após o término da implementação foi encontrada a função `document.caretPositionFromPoint(x, y)`, mas seria necessário realizar alguns testes para ver se o comportamento é o mesmo.

Como forma de contorno, no desktop é possível diminuir o zoom da página até que as opções de resposta sejam exibidas ou ocultar a barra de rolagem. Também, pode-se colocar o conteúdo exibido em tela cheia, no caso do Google Chrome a tecla de atalho é F11. Para os dispositivos móveis, deve-se utilizar o modo paisagem para a visualização e habilitar a versão desktop do site nas configurações da página assim como ilustrado na Figura 36.

Figura 36 - Habilitar versão desktop



Fonte: digitalizado pelo autor.

3.4.4 Comparativo entre o trabalho desenvolvido e os correlatos

Nesta seção é apresentado o Quadro 25 em que é realizada a comparação das principais características dos trabalhos correlatos descritos no capítulo anterior e o trabalho desenvolvido, EasyEdu.

Quadro 25 – Comparação dos trabalhos correlatos

Característica/ Trabalhos relacionados	Calpa (2012)	Maciel (2015)	Assis e Souza (2012)	EasyEdu
Multimídia (áudio e imagem)	X	X		X
Arrastar ao toque	X	X	X	X
Pode ser customizado ou adaptado a diferentes usuários		X	X	X
Pode ser customizado ou adaptado a diferentes conteúdos		X	X	X
Necessita de plataforma específica	X	X		
Ferramenta educacional	X	X	X	X
Interação multitoque	X	X	X	X
Integração com o Google Drive				X

Fonte: elaborado pelo autor.

Com base no Quadro 25 é possível perceber que todos os trabalhos relacionados possuem um cunho educacional, explorando o lado lúdico e permitindo a interação multitoque. Os trabalhos oferecem recursos de multimídia que possibilitam o uso de arquivos de áudio e imagem. Exceto o trabalho desenvolvido por Assis e Souza (2012), que por ser manual não permite que sejam adicionados recursos de áudio as atividades.

Os trabalhos citados permitem ser customizados e adaptados a diferentes conteúdos e usuários. O trabalho de Maciel (2015) permite que sejam criados jogos por meio da inclusão de novas pranchas. O trabalho de Assis e Souza (2012) possibilita adaptar a ideia do dominó a outros temas, como por exemplo a língua inglesa. O EasyEdu possui tipos de layouts genéricos, permitindo que o professor crie conteúdos e atividades variados para diferentes tipos de alunos. O único trabalho que não permite ser alterado é o de Calpa (2012), pois o jogo foi desenvolvido exclusivamente para pessoas com autismo.

Sobre os recursos tecnológicos necessários para utilizar cada um dos trabalhos, podemos observar que o trabalho de Calpa (2012) foi desenvolvido para funcionar somente na mesa multitoque. O motor de jogos 2D de Maciel (2015) foi disponibilizado apenas para a plataforma Android. O dominó das funções inorgânicas de Assis e Souza (2012) não depende de nenhuma plataforma para ser jogado, mas por outro lado não tem a praticidade provida pela tecnologia. O EasyEdu é o único que foi desenvolvido para web e que possui integração com o Google Drive. Isso possibilita ao usuário ter acesso às atividades independentemente da plataforma que utilize, desde que esteja conectado à internet.

4 CONCLUSÕES

O objetivo deste trabalho foi desenvolver uma ferramenta que pudesse contribuir para o processo de ensino e aprendizagem, explorando o lado lúdico e trazendo a tecnologia para dentro das salas de aula. Como resultado, esta monografia apresentou o desenvolvimento de um editor web para jogos multitoque, que fornece layouts customizáveis pelo professor, permitindo que sejam criados jogos de encaixe de letras e imagens, utilizando recursos multimídia. O desenvolvimento e a disponibilização do EasyEdu possibilitaram o atingimento do primeiro objetivo específico definido para o trabalho.

As atividades geradas pelo editor podem ser disponibilizadas para que os alunos as pratiquem no modo de jogo partida única ou partida dupla, incentivando a interação. Para melhor aproveitamento dos recursos que os jogos desenvolvidos proporcionam, recomenda-se a utilização de dispositivos que tenham suporte multitoque. Caso não seja possível, pode-se utilizar um computador convencional. Permitir que os alunos pratiquem as atividades desenvolvidas, tornou possível o atingimento do segundo e do terceiro objetivos específicos do trabalho. O EasyEdu foi validado com os alunos da turma de pedagogia da FURB, pois estes encaixam-se no perfil de especialistas. Os resultados obtidos apresentados na seção 3.4.2 foram gratificantes, fazendo com que o último objetivo específico proposto para o trabalho fosse alcançado.

As ferramentas utilizadas para o desenvolvimento do trabalho se mostraram adequadas para os fins aplicados. O AngularJS foi escolhido devido a vivência no framework, também por ser muito bem aceito e empregado no desenvolvimento de aplicações web, o que facilita os trabalhos de extensão. A integração com o Google para armazenar os dados no Drive não havia sido pensada inicialmente, mas ao colocá-la em prática identificou-se que ela agregou valor na entrega. As rotinas utilizadas na integração poderiam facilmente ser desenvolvidas com outra linguagem de back-end, mas perderíamos esse diferencial.

Outro ponto de destaque foi a utilização do QR Code para o compartilhamento das atividades, o que possibilita a troca de informação de forma prática. Mesmo não tendo experiência com aplicações multitoque, não foram encontrados problemas no emprego do Hammer.js. A utilização é simples e materiais de apoio são facilmente encontrados na internet.

Durante o desenvolvimento foram encontrados alguns problemas, que estão descritos por ordem de criticidade. A alteração do encode do arquivo de controle da aplicação por exemplo, quando eram informados caracteres acentuados. Os detalhes podem ser consultados

na seção 3.4.3. Outra situação comentada na seção 3.4.3 é com relação à função utilizada para obter o elemento onde o movimento do multitoque foi encerrado. O último caso foi a necessidade de enviar o JSON com os dados do conteúdo para um repositório público, devido ao bloqueio do Google para armazenar esse tipo de arquivo. Nesse caso os detalhes podem ser consultados na seção 3.3.1.1.

O desenvolvimento e a conclusão deste trabalho propiciaram ao autor a oportunidade, mesmo de uma maneira muito singela, contribuir para que o professor possa tornar as aulas menos monótonas e o ensino mais atrativo para os alunos, deixando-as mais motivadas para o aprendizado. Além disso, foi possível aplicar os conhecimentos adquiridos durante o curso de graduação, juntamente com a experiência trazida do meio profissional. Esses dois fatores se complementaram, garantido êxito no resultado.

4.1 EXTENSÕES

Durante o desenvolvimento deste trabalho, observaram-se algumas funcionalidades que poderiam ser implementadas em trabalhos futuros. São elas:

a) desenvolver um aplicativo híbrido ou nativo que permita praticar as atividades localmente aproveitando os recursos do dispositivo;
b) permitir a criação de atividades, utilizando Realidade Aumentada;
c) disponibilizar mais layouts, como jogo da memória e quebra-cabeça;
d) disponibilizar mais opções de conteúdos padrão, como sistema solar, órgãos do corpo humano;
e) permitir que o próprio professor possa criar o layout;
f) armazenar o tempo de resolução da atividade na Google Play Store, para que possa ser criado um ranking do exercício;
g) disponibilizar o conteúdo off-line;
h) permitir exportar e importar mais de um conteúdo;
i) permitir criar mais do que duas partidas consecutivas, para que os recursos da mesa multitoque sejam explorados ao máximo;
j) incorporar a interface de autenticação do Google ao EasyEdu para resolver o problema do bloqueio do pop-up;
k) aplicar internacionalização ao EasyEdu, permitindo que seja utilizado em outros idiomas
l) permitir rotação de atividades durante a execução
m) criar roteiro de atividades.

REFERÊNCIAS

ANGULARJS. **Developer Guide:** Data binding. [S.l.], [2010?]. Disponível em: <https://docs.angularjs.org/guide/databinding>. Acesso em: 05 maio 2017.

AQUA. **Como a tecnologia touchscreen está mudando o mundo.** [S.l.], 2014. Disponível em: <http://www.aqua.com.br/noticias/como-tecnologia-touchscreen-esta-mudando-o-mundo>. Acesso em: 01 nov. 2015.

ASSIS, Pablo de. **Como funcionam as telas sensíveis ao toque (touch screen).** [S.l.], 2009. Disponível em: <http://www.tecmundo.com.br/projetor/2449-como-funcionam-as-telas-sensiveis-ao-toque-touch-screen-.htm?utm_source=404corrigido&utm_medium=baixaki>. Acesso em: 01 nov. 2015.

ASSIS, Junior P.C.; SOUZA, A. P. **Jogo de dominó das funções inorgânicas:** uma ferramenta para o ensino da química geral na 1ª série do ensino médio de uma escola da rede particular da cidade de Manaus-AM. [Manaus], 2012. Disponível em: <http://www.abq.org.br/cbq/2012/trabalhos/6/1106-9162.html>. Acesso em: 13 set. 2015.

AUDINO, Daniel Fagundes. **Objetos de aprendizagem hipermídia aplicado à cartografia escolar no sexto ano do ensino fundamental em geografia**. 2012. 152 f. Dissertação (Pós-graduação em Geografia) - Universidade Federal de Santa Catarina, Florianópolis. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/99501/303100.pdf?sequence=1&isAllowed=y>. Acesso em: 15 abr. 2017.

BEHAR, Patricia Alejandra et al. **Modelos pedagógicos em educação a distância.** Porto Alegre: Artmed, 2009.

BIELSCHOWSKY, Carlos Eduardo; PRATA, Carmem Lúcia. Portal educacional do professor no Brasil. **Revista de Educación**, [Brasília], n. 352, Maio/Ago. 2010. Disponível em: <http://portaldoprofessor.mec.gov.br/storage/materiais/0000013441.pdf>. Acesso em: 07 abr. 2017.

BRUZZI, Demerval Guildarducci et al. **Linux educacional:** objetos de aprendizagem. [S.l], [2017?]. Disponível em: <http://webeduc.mec.gov.br/linuxeducacional/curso_le/modulo4.html>. Acesso em: 03 abr. 2017.

CAN I USE. **Drag and drop.** [S.l.], [2017?]. Disponível em: <https://caniuse.com/#search=drag%20and%20drop>. Acesso em: 15 mar. 2017.

CALPA, Greis Francy Mireya Silva. **PAR (Peço, Ajudo, Recebo):** Um jogo colaborativo em mesa multi-toque para apoiar a interação social de usuários com autismo. 2012. 106 f. Dissertação (Mestrado em Informática) - Curso de Informática, Departamento de Informática, PUC-Rio, Rio de Janeiro, 2012. Disponível em: <http://webserver2.tecgraf.puc-rio.br/~abraposo/pubs/alunos/dissertacao_GreisSilva_set12.pdf>. Acesso em: 30 ago. 2015.

CAREO. **Careo overview & goals.** [S.l.], [2002?]. Disponível em: <http://www.careo.org/documents/overview.html>. Acesso em: 01 abr. 2017.

DE MELLO, E. F. F.; TEIXEIRA, A. C. A interação social descrita por Vigotski e a sua possível ligação com a aprendizagem colaborativa através das tecnologias em rede. In: Seminário ANPED SUL, IV, 2012, Caxias do Sul. **Anais eletrônicos...** Caxias do Sul: [s.n.], 2012. p. 1362-1365. Disponível em: <http://www.portalanpedsul.com.br/admin/uploads/2012/Educacao_Comunicacao_e_Tecnologias/Trabalho/06_03_38_6-7515-1-PB.pdf>. Acesso em: 05 nov. 2015.

DRAGDROPTOUCH. **Polyfill that enables html5 drag drop support on mobile (touch) devices**. [S.l.], [2016?]. Disponível em: <https://github.com/bernardo-castilho/dragdroptouch>. Acesso em: 08 abr. 2017.

FALKEMBACH, G. A. M. **O Lúdico e os Jogos Educacionais**. Universidade Federal do Rio Grande do Sul, Centro Interdisciplinar de Novas Tecnologias na Educação, [Porto Alegre], 2007. Disponível em: <http://penta3.ufrgs.br/midiasedu/modulo13/etapa1/leituras/arquivos/Leitura_1.pdf>. Acesso em: 20 set. 2015.

FIALHO, N. N. Os jogos pedagógicos como ferramentas de ensino. In: CONGRESSO NACIONAL DE EDUCAÇÃO DA PUCPR – EDUCERE, III, 2008, Curitiba. **Anais eletrônicos...** Curitiba: CHAMPAGNAT, 2008. p. 12298-12306. Disponível em: <http://www.pucpr.br/eventos/educere/educere2008/anais/pdf/293_114.pdf>. Acesso em: 30 ago. 2015.

GALO, Bruno; SERRANO, Filipe; ROCHA, Juliana. **Evolução:** do touchscreen ao multitouch. [S.l.], 2009. Disponível em: <http://www.estadao.com.br/noticias/geral,evolucao-do-touchscreen-ao-multitouch,1461>. Acesso em: 13 set. 2015.

GOOGLE. **Google Drive:** armazenamento na nuvem e backup de arquivos para fotos, documentos e muito mais. [S.l.], [2017?a]. Disponível em: <https://www.google.com/intl/pt-BR_ALL/drive/>. Acesso em: 10 de abr. 2017.

GOOGLE DEVELOPERS. **API Discovery Document**. [S.l.], [2017?a]. Disponível em: <https://developers.google.com/api-client-library/javascript/features/discovery>. Acesso em: 20 de mar. 2017.

_____. **Using OAuth 2.0 to Access Google APIs**. [S.l.], [2017?b]. Disponível em: <https://developers.google.com/identity/protocols/OAuth2>. Acesso em: 15 abr. 2017.

_____. **Supported MIME Types**. [S.l.], [2017?c]. Disponível em: <https://developers.google.com/drive/v3/web/mime-types>. Acesso em: 05 de abr. 2017.

_____. **What is Google Picker?**. [S.l.], [2017?d]. Disponível em: <https://developers.google.com/picker/?hl=pt-BR>. Acesso em: 15 mar. 2017.

_____. **Google Picker Class Reference**. [S.l.], [2017?e]. Disponível em: <https://developers.google.com/picker/docs/reference>. Acesso em: 10 mar. 2017.

G SUITE. **G Suite Update Alerts:** Deprecating web hosting support in Google Drive. [S.l.], [2015?]. Disponível em: <https://gsuiteupdates.googleblog.com/2015/08/deprecating-web-hosting-support-in.html>. Acesso em: 20 de abr. 2017.

GRUPO DE PESQUISA EM COMPUTAÇÃO GRÁFICA, PROCESSAMENTO DE IMAGENS E ENTRETENIMENTO DIGITAL. **MM – Mesa Multi-Touch Screen**. [Blumenau], 2014. Disponível em: <http://gcg.inf.furb.br/?page_id=3376>. Acesso em: 25 out. 2015.

HAMMER.JS. **Getting started**. [S.l.], [2017?]. Disponível em: <https://hammerjs.github.io/getting-started/>. Acesso em: 25 abr. 2017.

INTERNET ENGINEERING TASK FORCE. **RFC 6749**: The OAuth 2.0 Authorization Framework. [S.l.], [2017?a]. Disponível em: <https://tools.ietf.org/html/rfc6749>. Acesso em 30 de abr. 2017.

LODASH. **Why lodash?**. [S.l.], [2017?]. Disponível em: <https://lodash.com>. Acesso em: 25 abr. 2017.

MACIEL, Gilson Rodrigues. **Motor de jogos 2D de encaixe de imagens na plataforma Android.** 2015. 94 f. Trabalho de Conclusão de Curso (Bacharel em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau, Brasil.

MDN. **Document.elementFromPoint().** [S.l.], [2017?]. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/API/Document/elementFromPoint>. Acesso em: 30 maio 2017.

MICROSOFT. **Experience Things in a Whole New Way.** [S.l.], 2012. Disponível em: <https://www.microsoft.com/en-us/pixelsense/whatissurface.aspx>. Acesso em: 01 nov. 2015.

MOMENT.JS. **Where to use it.** [S.l.], [2017?]. Disponível em: < https://momentjs.com/docs>. Acesso em: 30 abr. 2017.

MYJSON. **About.** [S.l.], [2017?]. Disponível em: <http://myjson.com/about>. Acesso em: 30 abr. 2017.

PHASER. **Phaser - HTML5 Game Framework.** [S.l.], [2017?]. Disponível em: <https://github.com/photonstorm/phaser>. Acesso em: 13 mar. 2017.

RIVED. **Conheça o rived.** [S.l.], [1997?]. Disponível em: <http://rived.mec.gov.br/site_objeto_lis.php>. Acesso em: 16 abr. 2017.

ROSS, Paulo Ricardo. Aprendizagem e conhecimento: fundamentos para as práticas inclusivas. **Perspectiva**, Florianópolis, v. 24, n. 3, p. 273-302, jan. 2006. Disponível em: <https://periodicos.ufsc.br/index.php/perspectiva/article/view/10605>. Acesso em: 02 nov. 2015.

SILVEIRA, R. S; BARONE, D. A. C. **Jogos educativos computadorizados utilizando a abordagem de algoritmos genéticos**. 1998. Dissertação (Pós-Graduação em Ciências da Computação) - Universidade Federal do Rio Grande do Sul, Porto Alegre, 1998.

SOARES, Jiane M. A importância do lúdico na educação. [S.l.], 2013. Disponível em:<http://www.planetaeducacao.com.br/portal/imagens/artigos/diario/ARTIGO%20JIANE%20JOGO1.pdf>. Acesso em: 09 maio 2017.

SPINELLI, Walter. **Os objetos virtuais de aprendizagem**: Ação, criação e conhecimento. [S.l], [2007?]. Disponível em: <http://rived.mec.gov.br/comousar/textoscomplementares/textoImodulo5.pdf>. Acesso em: 04 abr. 2017.

VIEIRA, Larissa de S.; OLIVEIRA, Valdiléia X. de. A importância dos jogos e brincadeiras para o processo de alfabetização e letramento. In: ENCONTRO DE PRODUÇÃO CIENTÍFICA E TECNOLÓGICA – EPCT, V, 2010, Campo Mourão. **Anais eletrônicos...** Disponível em: <http://www.fecilcam.br/nupem/anais_v_epct/PDF/ciencias_humanas/21_VIEIRA_OLIVEIRA.pdf>. Acesso em: 05 maio 2017.

WIKIPEDIA. **Marca temporal**. [S.l.], [2017?]. Disponível em: <https://pt.wikipedia.org/wiki/Marca_temporal>. Acesso em: 02 maio 2017.

WILEY, David Arnim. **Learning Object Design and Sequenceing Theory.** 2000. 120 f. Tese (Doutorado) - Curso de Filosofia, Brigham Young University, Provo, Estados Unidos da América.

APÊNDICE A – Exemplo do arquivo de controle do editor

Neste apêndice é demonstrado o formato JSON do arquivo de controle do editor com os dados dos conteúdos padrão (Quadro 26).

Quadro 26 – Arquivo de controle do editor


```
[
   {
      "id":"0B0AX8dmLiwKEa1lLNW13eEtXRGM",
      "name":"Matemática - Figuras geométricas",
      "createdTime":1497321293737
   },
   {
      "id":"0B0AX8dmLiwKEazVld2stR3ctUjA",
      "name":"Biologia - Mamíferos",
      "createdTime":1497321792221
   },
   {
      "id":"0B0AX8dmLiwKERU9wLVJHTXJkSUE",
      "name":"Geografia - Estados brasileiros",
      "createdTime":1497445274777
   }
]
```


Fonte: elaborado pelo autor.

APÊNDICE B – Exemplo do arquivo de controle de um conteúdo

Neste apêndice é demonstrado o formato JSON do arquivo de controle de um conteúdo padrão (Quadro 27).

Quadro 27 – Arquivo de controle de um conteúdo

```
{
   "name":"Geografia - Estados brasileiros",
   "parent":"0B1_Ihk-LLpk9VE5CTzlXQXZUZHc",
   "createdTime":1497448500316,
   "id":"0B1_Ihk-LLpk9UEVvdGNrNVF6bEE",
   "metadataRoot":{
      "id":"0B1_Ihk-LLpk9V2pJTDduZHRndE0",
      "parent":"0B1_Ihk-LLpk9WXBZY2pEaHZPTmM"
   },
   "image":{
      "id":"0B1_Ihk-LLpk9TFdlN3JKRGxTQ0k",
      "name":"brasil.jpg"
   },
   "activities":[
      {
         "correctAnswers":3,
         "level":"EASY",
         "type":"PICTURES",
         "name":"Sul",
         "tip":"Bandeiras estados do Sul",
         "time":"05:00",
         "answerOptions":[
            ...
            {
               "image":{
                  "id":"0B1_Ihk-LLpk9TmVSRDFnWWRwczQ",
                  "name":"santa_catarina.jpg"
               },
               "type":"CORRECT"
            },
            {
               "image":{
                  "id":"0B1_Ihk-LLpk9QVZjZWE0VVQ3M0U",
                  "name":"sao_paulo.jpg"
               },
               "type":"INCORRECT"
            }
            ...
         ],
         "id":1497448752724
         ...
      },
      {
         "level":"MEDIUM",
         "type":"LETTERS",
         "image":{
            "id":"0B1_Ihk-LLpk9SmExMUt0TzVXNVE",
            "name":"amazonas.jpg"
         },
         "name":"AM",
         "answer":"Amazonas",
         "tip":"Estado do Norte",
         "time":"10:00",
         "id":1497449031478,
         ...
      }
   ]
}
```

Fonte: elaborado pelo autor.

APÊNDICE C – Experimento do EasyEdu

Neste apêndice são apresentadas fotos do teste realizado no dia 12 de junho de 2017 no laboratório LIFE com os acadêmicos da 7ª fase do curso de Pedagogia da FURB, da disciplina de Tecnologia Educacional e Aprendizagem do professor Mauricio Capobianco Lopes. É possível observar na Figura 37 e na Figura 38, o editor sendo utilizado pelos usuários especialistas.

Figura 37 - Usuários interagindo com o EasyEdu

Fonte: digitalizado pelo autor.

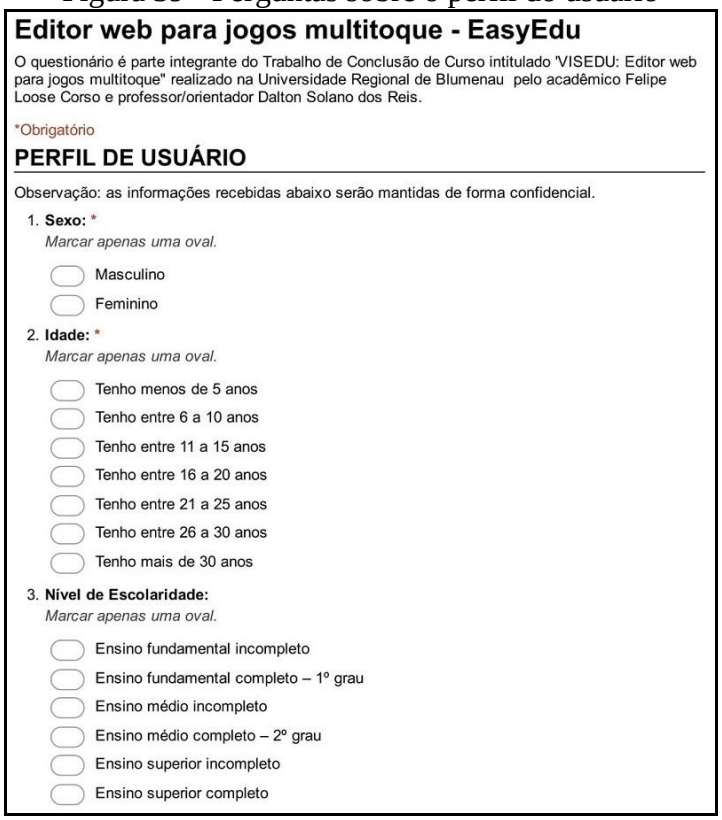
Figura 38 - Usuários interagindo com o EasyEdu

Fonte: digitalizado pelo autor.

APÊNDICE D – Questionário do experimento

Neste apêndice é apresentado o questionário feito para o teste do editor web para jogos multitoque - EasyEdu. A Figura 39 e a Figura 40 mostram as perguntas de perfil do usuário. A Figura 41, Figura 42 e Figura 43 apresentam a seção de instruções para o uso do editor com o papel de professor. A Figura 44 e a Figura 45 apresentam a seção de instruções para o uso do EasyEdu com o papel de aluno. A Figura 46 e a Figura 47 apresentam as perguntas sobre usabilidade do editor. Por fim, a Figura 48 traz as observações feitas pelos entrevistados.

Figura 39 - Perguntas sobre o perfil do usuário



Editor web para jogos multitoque - EasyEdu

O questionário é parte integrante do Trabalho de Conclusão de Curso intitulado 'VISEDU: Editor web para jogos multitoque" realizado na Universidade Regional de Blumenau pelo acadêmico Felipe Loose Corso e professor/orientador Dalton Solano dos Reis.

*Obrigatório

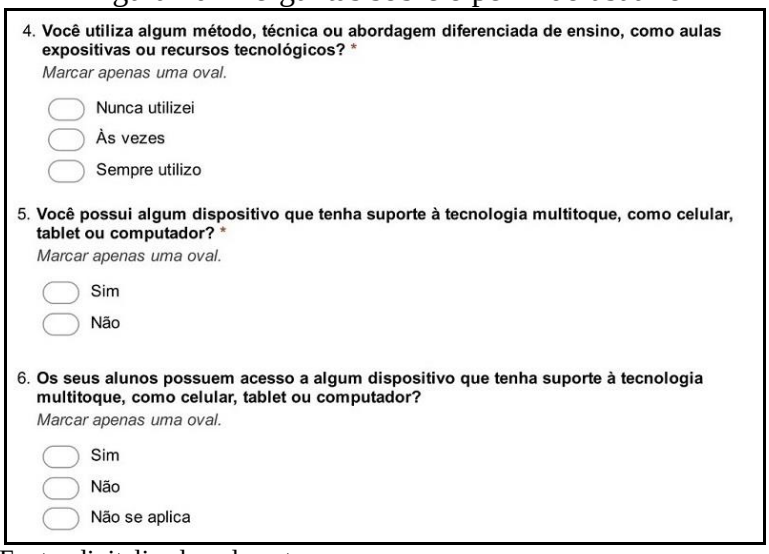
PERFIL DE USUÁRIO

Observação: as informações recebidas abaixo serão mantidas de forma confidencial.

1. **Sexo:** *
   Marcar apenas uma oval.
   - ◯ Masculino
   - ◯ Feminino
2. **Idade:** *
   Marcar apenas uma oval.
   - ◯ Tenho menos de 5 anos
   - ◯ Tenho entre 6 a 10 anos
   - ◯ Tenho entre 11 a 15 anos
   - ◯ Tenho entre 16 a 20 anos
   - ◯ Tenho entre 21 a 25 anos
   - ◯ Tenho entre 26 a 30 anos
   - ◯ Tenho mais de 30 anos
3. **Nível de Escolaridade:**
   Marcar apenas uma oval.
   - ◯ Ensino fundamental incompleto
   - ◯ Ensino fundamental completo – 1º grau
   - ◯ Ensino médio incompleto
   - ◯ Ensino médio completo – 2º grau
   - ◯ Ensino superior incompleto
   - ◯ Ensino superior completo

Fonte: digitalizado pelo autor.

Figura 40 - Perguntas sobre o perfil do usuário



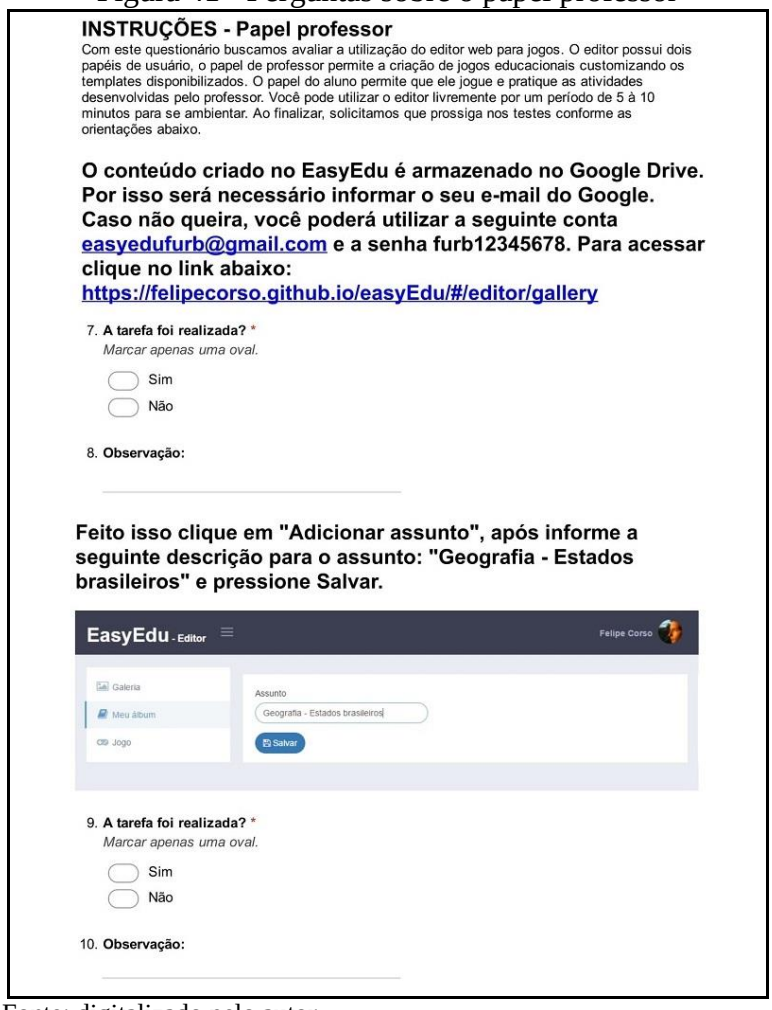
4. **Você utiliza algum método, técnica ou abordagem diferenciada de ensino, como aulas expositivas ou recursos tecnológicos?** *
Marcar apenas uma oval.
- Nunca utilizei
- Às vezes
- Sempre utilizo

5. **Você possui algum dispositivo que tenha suporte à tecnologia multitoque, como celular, tablet ou computador?** *
Marcar apenas uma oval.
- Sim
- Não

6. **Os seus alunos possuem acesso a algum dispositivo que tenha suporte à tecnologia multitoque, como celular, tablet ou computador?**
Marcar apenas uma oval.
- Sim
- Não
- Não se aplica

Fonte: digitalizado pelo autor.

Figura 41 - Perguntas sobre o papel professor



INSTRUÇÕES - Papel professor

Com este questionário buscamos avaliar a utilização do editor web para jogos. O editor possui dois papéis de usuário, o papel de professor permite a criação de jogos educacionais customizando os templates disponibilizados. O papel do aluno permite que ele jogue e pratique as atividades desenvolvidas pelo professor. Você pode utilizar o editor livremente por um período de 5 à 10 minutos para se ambientar. Ao finalizar, solicitamos que prossiga nos testes conforme as orientações abaixo.

O conteúdo criado no EasyEdu é armazenado no Google Drive. Por isso será necessário informar o seu e-mail do Google. Caso não queira, você poderá utilizar a seguinte conta easyedufurb@gmail.com e a senha furb12345678. Para acessar clique no link abaixo: https://felipecorso.github.io/easyEdu/#/editor/gallery

7. **A tarefa foi realizada?** *
Marcar apenas uma oval.
- Sim
- Não

8. **Observação:**

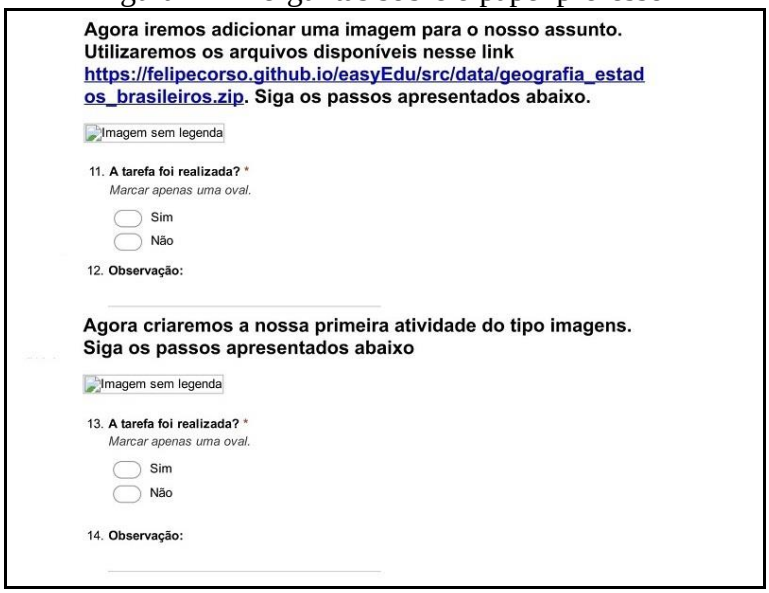
Feito isso clique em "Adicionar assunto", após informe a seguinte descrição para o assunto: "Geografia - Estados brasileiros" e pressione Salvar.

9. **A tarefa foi realizada?** *
Marcar apenas uma oval.
- Sim
- Não

10. **Observação:**

Fonte: digitalizado pelo autor.

Figura 42 - Perguntas sobre o papel professor



Agora iremos adicionar uma imagem para o nosso assunto. Utilizaremos os arquivos disponíveis nesse link https://felipecorso.github.io/easyEdu/src/data/geografia_estados_brasileiros.zip. Siga os passos apresentados abaixo.

Imagem sem legenda

11. **A tarefa foi realizada?** *
Marcar apenas uma oval.
Sim
Não

12. **Observação:**

Agora criaremos a nossa primeira atividade do tipo imagens. Siga os passos apresentados abaixo

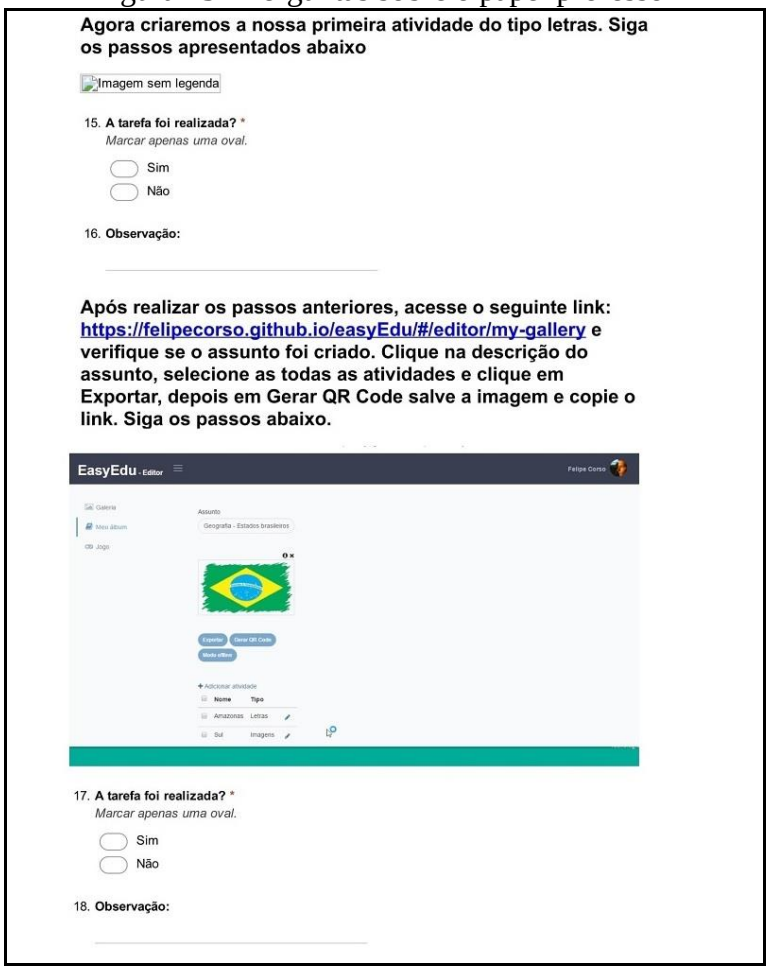
Imagem sem legenda

13. **A tarefa foi realizada?** *
Marcar apenas uma oval.
Sim
Não

14. **Observação:**

Fonte: digitalizado pelo autor.

Figura 43 - Perguntas sobre o papel professor



Agora criaremos a nossa primeira atividade do tipo letras. Siga os passos apresentados abaixo

Imagem sem legenda

15. **A tarefa foi realizada?** *
Marcar apenas uma oval.
Sim
Não

16. **Observação:**

Após realizar os passos anteriores, acesse o seguinte link: https://felipecorso.github.io/easyEdu/#/editor/my-gallery e verifique se o assunto foi criado. Clique na descrição do assunto, selecione as todas as atividades e clique em Exportar, depois em Gerar QR Code salve a imagem e copie o link. Siga os passos abaixo.

17. **A tarefa foi realizada?** *
Marcar apenas uma oval.
Sim
Não

18. **Observação:**

Fonte: digitalizado pelo autor.

Figura 44 - Perguntas sobre o papel aluno

INSTRUÇÕES - Papel aluno

Após concluir a criação do assunto iremos praticar as atividades jogando.

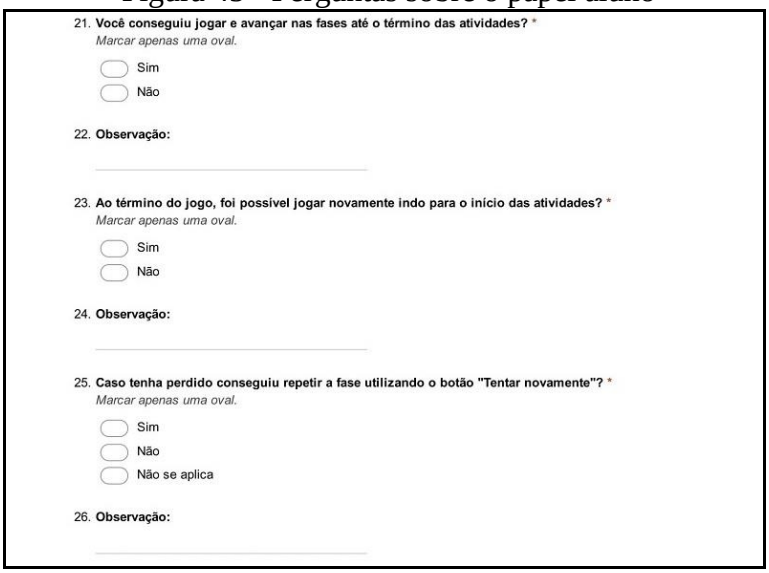
Você pode acessar a atividade por meio do link do QR Code, acessando o endereço https://felipecorso.github.io/easyEdu/#/game/category e importando o assunto ou ainda pelo link que foi copiado para área de transferência.

19. **A tarefa foi realizada?** *
Marcar apenas uma oval.
- Sim
- Não

20. **Observação:**

Fonte: digitalizado pelo autor.

Figura 45 - Perguntas sobre o papel aluno



21. **Você conseguiu jogar e avançar nas fases até o término das atividades?** *
Marcar apenas uma oval.
- Sim
- Não

22. **Observação:**

23. **Ao término do jogo, foi possivel jogar novamente indo para o início das atividades?** *
Marcar apenas uma oval.
- Sim
- Não

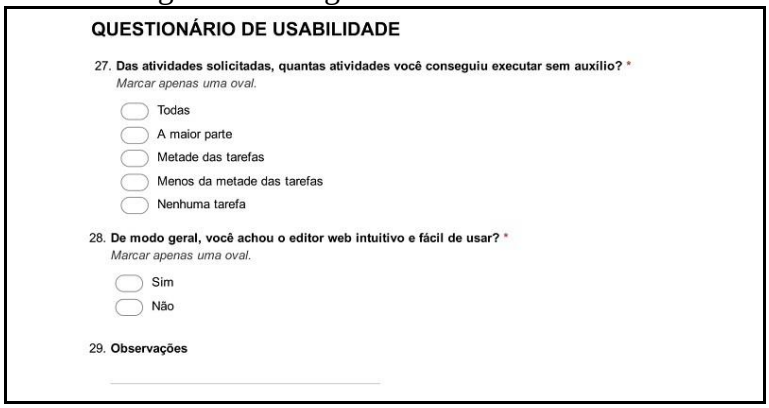
24. **Observação:**

25. **Caso tenha perdido conseguiu repetir a fase utilizando o botão "Tentar novamente"?** *
Marcar apenas uma oval.
- Sim
- Não
- Não se aplica

26. **Observação:**

Fonte: digitalizado pelo autor.

Figura 46 - Perguntas sobre usabilidade



QUESTIONÁRIO DE USABILIDADE

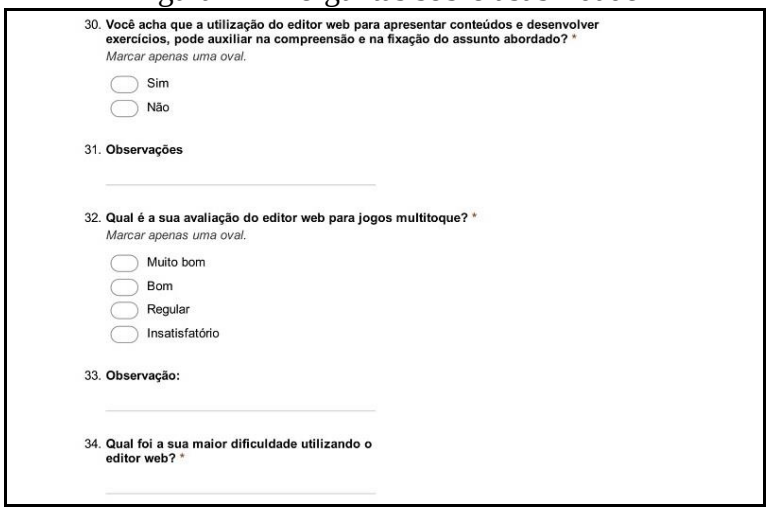
27. **Das atividades solicitadas, quantas atividades você conseguiu executar sem auxílio?** *
Marcar apenas uma oval.
- Todas
- A maior parte
- Metade das tarefas
- Menos da metade das tarefas
- Nenhuma tarefa

28. **De modo geral, você achou o editor web intuitivo e fácil de usar?** *
Marcar apenas uma oval.
- Sim
- Não

29. **Observações**

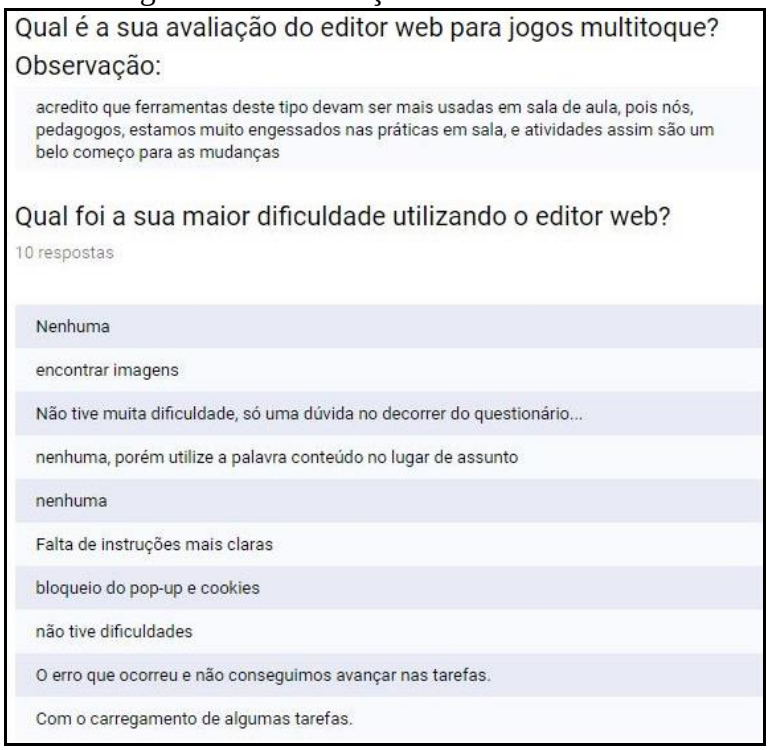
Fonte: digitalizado pelo autor.

Figura 47 - Perguntas sobre usabilidade



Fonte: digitalizado pelo autor.

Figura 48 - Observações dos entrevistados



Fonte: digitalizado pelo autor.