UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

EDIBOX: EDITOR DE JOGOS MULTIPLATAFORMA

MARCOS DOUGLAS HOPPE

BLUMENAU
2017

MARCOS DOUGLAS HOPPE

EDIBOX: EDITOR DE JOGOS MULTIPLATAFORMA

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

EDIBOX: EDITOR DE JOGOS MULTIPLATAFORMA

Por

MARCOS DOUGLAS HOPPE

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. Maurício Capobianco Lopes, Dr. – FURB
Membro:	Profa. Joyce Martins, M. Sc. – FURB

Blumenau, 11 de dezembro de 2017

Dedico este trabalho a minha família, aos meus amigos e colegas de trabalho que me apoiarem ao longo desta caminhada e a todos aqueles que de alguma maneira me ajudaram na realização desta monografia.

AGRADECIMENTOS

A Deus por ter me dado saúde e força para superar todas as dificuldades.

À minha família que me apoiou ao longo deste curso, não me deixando desmotivar-se.

Aos meus amigos e companheiros de trabalho que fizeram parte da minha formação e que vão continuar presentes em minha vida.

Ao meu orientador, Dalton Solano dos Reis, por acreditar na realização deste trabalho e por todo apoio e incentivo prestados para o desenvolvimento desta monografia.

A minha namorada pelo apoio e incentivos para não desistir.

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

A todos que direta e indiretamente fizeram parte da minha formação, o meu muito obrigado!

A imaginação é mais importante que o conhecimento.

Albert Einstein

RESUMO

Este trabalho apresenta o desenvolvimento de um editor de jogos educativos multiplataforma que visa auxiliar os educadores no processo de ensino e aprendizagem. No editor, o educador pode construir jogos didáticos com os conteúdos trabalhados em sala de aula, através dos *templates* disponíveis, auxiliando desta forma os educandos através destas atividades digitais a compreenderem os conteúdos abordados e a desenvolverem novas habilidades. As atividades podem ser compartilhadas e editadas. O EdiBox está integrado com a plataforma Firebase possibilitando que os conteúdos desenvolvidos sejam armazenados e acessados em tempo real de qualquer dispositivo conectado à rede de internet. O Firebase também disponibiliza serviços de autenticação e muitos outros para a expansão do editor. Para o desenvolvimento foi utilizado o Framework Ionic 3 que possibilitou a criação desta aplicação híbrida, tornando-a multiplataforma. O Ionic utiliza internamente a plataforma Apache Cordova para ter acesso aos recursos nativos dos dispositivos móveis, através dos inúmeros *plugins* disponibilizados e a plataforma Angular 4 que apresenta componentes de interface de usuário para o desenvolvimento web. Mesmo que ainda existam melhorias e funcionalidades a serem realizadas, o editor obteve bons resultados como visto na pesquisa realizada com os especialistas, mostrando que a utilização do EdiBox pode auxiliar na produção de aulas mais interativas e na compreensão e fixação dos assuntos abordados. Com isso, é possível assegurar que o objetivo principal do trabalho foi alcançado.

Palavras-chave: Jogos educacionais. Ionic. Firebase. Multiplatarforma.

ABSTRACT

This work presents the development of a multiplatform educational game publisher that aims to assist educators in the teaching and learning process. In the editor, the educator can build didactic games with the contents addressed in the classroom through the available templates, thus helping the students through the digital activities to understand the covered contents and to develop new skills. Activities can be shared and edited. EdiBox is integrated with Firebase enabling the developed content to be stored and accessed in real time from any device connected to the internet network. Firebase also provides authentication services and many other tools for publisher expansion. For the development, the Ionic Framework 3 was used, and it made possible the creation of a hybrid and multiplatform application. Ionic internally uses the Apache Cordova platform to access the native features of mobile devices through the numerous plugins available and the Angular 4 platform that features user interface components for web development. Even though there are still improvements and functionalities to be carried out, the editor obtained good results as seen in the research conducted with specialists, showing that the use of the EdiBox can help in the production of more interactive classes and in the understanding and fixation of the subjects addressed. With this, it is possible to ensure that the main goal of the work has been achieved.

Key-words: Educational games. Ionic. Firebase. Multiplatform.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API – Application Programming Interface

CLI – Command Line Interface

CSS – Cascading Style Sheets

HTML – Hyper Text Markup Language

RF – Requisito Funcional

RNF – Requisito Não Funcional

SDK – Software Development Kit

UI – Interface de Usuário

URL – Uniform Resource Locator

SUMÁRIO

1 INTRODUÇÃO

Atualmente, os jogos digitais estão evoluindo constantemente, tornando-se para seus jogadores cada vez mais divertidos, interativos e realistas (PAULA, 2015). Assim sendo, muitas companhias mundiais estão investindo em novos dispositivos, recursos e tecnologias que levarão as pessoas a experiências imersivas cada vez mais próximas da sua realidade (LANDIM, 2015). As inovações tecnológicas estão mudando a cada dia a vida e a cultura da sociedade.

Os jogos digitais bem construídos, que contêm um contexto histórico, social e cultural e que sejam desafiadores, podem ser mediadores na relação das crianças com o conhecimento, ampliando, com o uso da tecnologia, a motivação e o interesse pelas aulas, mostrando-lhes que aprender é estimulante, lúdico e gratificante, além de explorar a criatividade por meio de experiências novas que as crianças não vivenciaram no seu ambiente social (LEITE; UGGIONI, 2013). Quando se fala em jogos como forma de auxiliar a aprendizagem do educando deve-se levar em consideração a abordagem utilizada, já que ela é o principal fator que determinará o sucesso na construção do conhecimento (PAULA; VALENTE, 2016).

Nas unidades educacionais, segundo Santos (2016), a informática e suas tecnologias auxiliam no desenvolvimento cognitivo e no processo de aprendizagem dos educandos por meio dos jogos digitais. Mas, devido às limitações por restrição de material educativo de qualidade, custo para aquisição e dificuldades de aplicação no dia a dia do educando, os mesmos apresentam características planares e sem o uso das tecnologias. Os jogos planares são desenvolvidos sobre um plano reto, ou seja, em cima da mesa e geralmente não apresentam sobreposição de peças, como por exemplo jogos de tabuleiros (xadrez) ou jogos de encaixe (quebra-cabeça).

Diante do exposto, este trabalho desenvolveu um editor de jogos educativos para múltiplas plataformas, onde os educadores podem construir jogos lúdicos para sua disciplina, adicionando o conteúdo que está sendo trabalhado em sala de aula ao jogo. Desta forma, proporciona o desenvolvimento de múltiplas habilidades e a construção do conhecimento nos educandos de forma interativa e motivadora.

1.1 OBJETIVOS

O objetivo deste trabalho é disponibilizar um editor de jogos educacionais.

Os objetivos específicos são:

a) fornecer um ambiente que possibilite ao educador construir jogos didáticos;
b) auxiliar os educadores no processo de ensino e aprendizagem;

c) disponibilizar a aplicação nas plataformas móvel e web.

1.2 ESTRUTURA

Este trabalho está estruturado em quatro capítulos. O primeiro é dedicado à introdução ao tema e aos objetivos geral e específicos. O segundo capítulo detalha a fundamentação teórica, visado fornecer um embasamento a respeito dos principais assuntos abordados no trabalho e necessários para o bom entendimento, além disto são exibidos três trabalhos correlatos a este. No terceiro capítulo é apresentado o desenvolvimento da aplicação, no qual são listados os requisitos especificados por meio de diagramas, detalhada a implementação com as principais técnicas e ferramentas utilizadas e a operacionalidade da aplicação. Na sequência do capítulo são apresentadas as análises dos resultados, comentando as dificuldades encontradas durante o desenvolvimento, além da realização de testes de interface, usabilidade e comparativo com os trabalhos correlatos. Por fim, o quarto capítulo expõe as conclusões obtidas no presente trabalho e apresenta sugestões de extensões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo explorar os principais assuntos e fundamentos necessários para a realização deste trabalho. A seção 2.1 demonstra a utilização dos jogos na educação. Na seção 2.2 é apresentado o potencial do *framework* Ionic. A seção 2.3 aborda a ferramenta de *back-end* Firebase. A última seção 2.4 descreve três trabalhos correlatos ao proposto.

2.1 JOGOS NA EDUCAÇÃO

Atualmente os jogos digitais encontram-se em evidência no contexto do entretenimento. Conforme as tecnologias evoluem, as empresas de jogos digitais conseguem desenvolver jogos que apresentam grande interatividade, realidade, qualidade gráfica e jogabilidade para um nicho de mercado cada vez maior de jogadores (PAULA, 2015).

A introdução de tecnologias digitais no ambiente escolar vem causando uma verdadeira revolução na educação. No entanto, essas mudanças não devem ser encaradas como algo simples, visto que tornam as atividades corriqueiras mais fáceis do que livros e cadernos, porém deve-se levar em consideração a abordagem utilizada, já que ela determinará a construção do conhecimento (PAULA; VALENTE, 2016).

Esta mudança de paradigma escolar tem grandes desafios, já que precisa ocorrer uma verdadeira integração entre os jogos e a educação, visto que segundo Paula e Valente (2016), as experiências com as Tecnologias da Informação e Comunicação (TIC) no contexto escolar demonstram que os alunos têm fora do ambiente escolar experiências bem mais ricas, as quais dentro das escolas não são proporcionadas, desmotivando-os já que os jogos digitais são extremamente voltados para a busca do conhecimento e memorização do conteúdo como no ensino formal.

Jogando e brincando as crianças podem reinventar o mundo, explorando sua criatividade e alargando a fronteira entre fantasia e realidade, posto que neste ambiente elas podem ser os personagens, recriar o cenário, compreender os diferentes papeis sociais, além de estimular a interatividade e socialização com outras crianças, fortalecendo os valores sociais. Já nos jogos digitais, as brincadeiras têm regras, personagens, cenários, objetivos, obstáculos e apresentam condições de vitória ou derrota que desafiam estas crianças a experiências ainda mais criativas e estimulantes (LEITE; UGGIONI, 2013).

Certos jogos digitais camuflam os aspectos pedagógicos dizendo ser educativos. Observa-se que os aspectos geralmente explorados são a destreza e a competitividade e não a capacidade criadora, de análise crítica, de colaboração e o desenvolvimento da imaginação

que abrangem diferentes conceitos das áreas de conhecimento. Essa discussão é bem abrangente, pois não se pode afirmar até que ponto os jogos podem ajudar na aprendizagem ou se os educadores estão prontos para explorarem este recurso e como irão fazê-lo, mas

> Acreditamos que diante de jogos digitais desafiadores, contextualizados, interdisciplinares, dinâmicos, sempre em diálogo com múltiplos contextos históricos, sociais e culturais, que agem como mediadores da relação da criança com o conhecimento, o uso da tecnologia pode aumentar a motivação e o interesse de crianças, favorecendo a percepção de que aprender seja gostoso, lúdico, e estimule a criatividade. Deste modo, os estudantes podem processar informações, analisar opções, tomar decisões e resolver situações-problema de forma cooperativa e significativa. (LEITE; UGGIONI, 2013, p. 1).

Paula e Valente (2016) afirmam que os jogos digitais não são novidades para o ensino de conteúdo específico em atividades extraclasse. Acredita-se que um dos motivos para essa escolha é a capacidade de motivação que esses artefatos possuem e devido ao ambiente tecnológico que o aluno está acostumado a vivenciar.

As inovações tecnológicas estão mudando a cada dia a vida e a cultura de uma determinada sociedade, diante disto, sabe-se que as crianças e os jovens estão cada vez mais próximas de aparatos tecnológicos. Portanto, pode-se dizer que as tecnologias estão mudando a prática da docência que deve integrar essas tecnologias para ficar de acordo com as tendências. Também se viu uma grande desigualdade social marcada pelo acesso a esses dispositivos, já que o educando de baixa renda tem menos contato com estes recursos (ALMEIDA, 2014).

Diante desta desigualdade, o ambiente escolar pode proporcionar a interação igualitária das tecnologias, porém é indispensável que o professor tenha conhecimento sólido em informática, para dominar as inúmeras tecnologias inovadoras que aparecem no mercado, uma vez que os jogos são ferramentas básicas assim como a internet que traz várias possibilidades aos alunos de adquirirem conhecimento e socialização (ALMEIDA, 2014). A atividade realizada na internet pode trazer ao aluno um rico conhecimento, além disto, ele adquire novas experiências de socialização, já que interage com pessoas do mundo inteiro. Citando como exemplo um jogo de RPG online, no qual são encontrados milhares de jogadores, de vários locais e culturas diferentes, buscando o mesmo objetivo. Nestes jogos podem ser desenvolvidas múltiplas habilidades que serão levadas para a vida adulta destas crianças.

2.2 FERRAMENTA IONIC PARA DESENVOLVIMENTO MULTIPLATAFORMA

A empresa Ionic foi fundada em 2012, em um tempo na qual existia pouco desenvolvimento de aplicativos para dispositivos móveis usando tecnologias web. Sendo assim, o *framework* Ionic visa a criação de aplicações híbridas para este mercado, tornando-o

mais fácil para os desenvolvedores que precisam conhecer apenas uma linguagem de programação e com isto produzem aplicativos para múltiplas plataformas utilizando um único código (IONIC, 2017).

O *framework* foi construído sobre a plataforma Apache Cordova, que permite acesso aos recursos nativos do dispositivo, como câmera, GPS e acelerômetro, dentre outros, e o Angular, que é uma plataforma JavaScript de código aberto, mantido pelo Google, o qual é amplamente utilizado para o desenvolvimento de aplicações web no estilo Model-View-Controller (MVC), fornecendo componentes de interface de usuário (UI) de qualidade (CABRAL, 2016).

O Ionic está atualmente na versão 3 e utiliza o conceito de Aplicações de Página Única (Single Page Application - SPA), herdado do Angular. As SPAs são aplicações completas desenvolvidas em JavaScript e se parecem com aplicações desktop, sendo que grande parte do funcionamento da aplicação fica no lado do cliente, melhorando consideravelmente a performance e a experiência do usuário, já que realiza menos requisições ao servidor, além disso ajuda na manutenção dado que separa o *front-end* do *back-end* (KLAUS, 2016).

As aplicações híbridas utilizam a linguagem web para a construção de aplicativos, porém somente Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS) e JavaScript não são suficientes para rodar perfeitamente em dispositivos móveis. Então está aplicação é encapsulada em um *webview*, que é o componente na qual são exibidos os conteúdos web em dispositivos móveis. Para a integração com os recursos nativos utiliza o Apache Cordova que tem o papel de realizar a comunicação entre a aplicação e o dispositivo (LOPES, 2016).

O Cordova é um *framework* de código aberto e gratuito, baseado na licença Apache. Este basicamente é uma distribuição do PhoneGap que foi criado pela empresa Nitobi e vendido em 2011 para a Adobe Systems Inc. e logo após teve seu código doado para a Apache Software Foundation. Devido ao PhoneGap já ser uma marca registrada e ter um ambiente integrado à Adobe, nasceu desta forma a marca Cordova (LOPES, 2016). Segundo os autores Matos e Silva (2016), o Cordova possui vários *plugins* para garantir o acesso às funcionalidades específicas do dispositivo em que está sendo executada a aplicação, tornando-a semelhante com uma aplicação nativa. Porém, para deixar a UI mais parecida com aplicativos nativos, utiliza outras ferramentas em conjunto, como o Angular.

A plataforma Angular é amplamente integrada ao Ionic para geração de aplicações híbridas, sendo que funciona na maioria dos navegadores modernos. O *framework* é estruturado principalmente em módulos e componentes que podem ser reaproveitados,

gerando uma clara e fácil manutenção do código. Diante disto, o Angular também tem uma comunidade global e atuante (CAMPOS, 2017). O componente encapsula uma estrutura HTML, CSS e comportamento JavaScript e pode ser utilizado como *tags* do HTML customizadas (PIRES, 2017).

O Ionic e o Angular contam com ferramentas para auxílio na construção das páginas, como interface de linha de comando (CLI) para criação, testes e depuração do código. O Ionic Lab no qual é possível executar os simuladores de cada plataforma e o Ionic Creator é uma ferramenta on-line que possibilita a criação de protótipos funcionais (MATOS; SILVA, 2016).

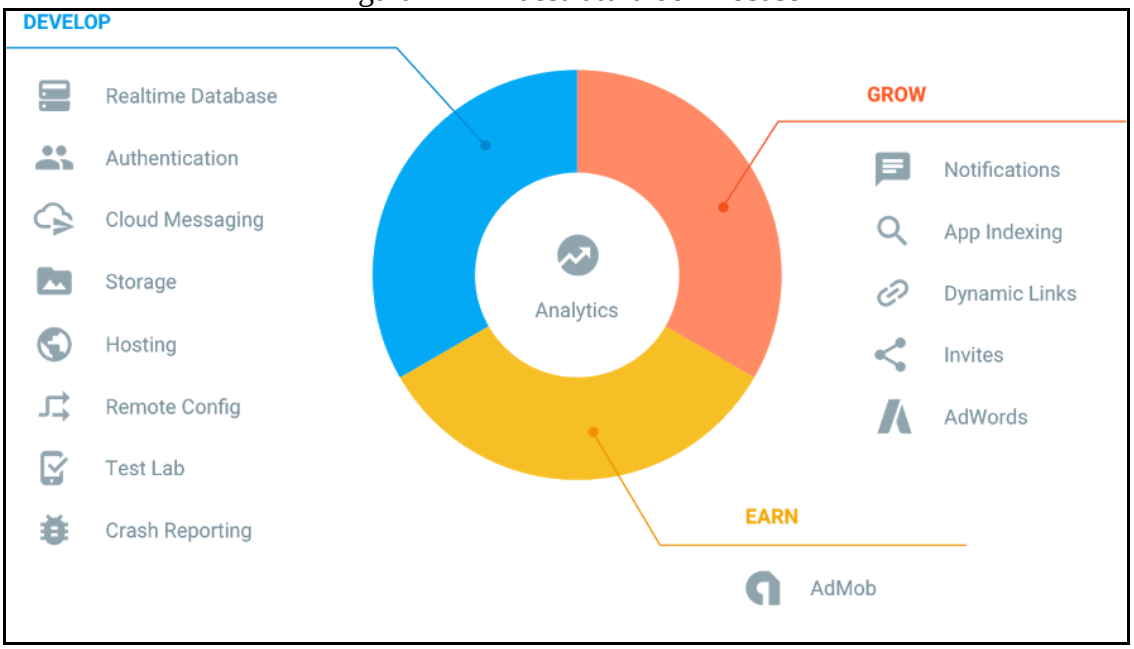
2.3 FIREBASE COMO SERVIÇOS DE BACKEND

A Google adquiriu em 2014 o Firebase que atualmente oferece diversos serviços baseados na nuvem. A Application Programming Interface (API) do Firebase disponibiliza serviços de *back-end* para desenvolvedores de aplicações móveis (Android e iOS) e aplicações web (BATTISTELLI, 2017).

A plataforma oferece várias funcionalidades, dentre as principais pode-se citar o armazenamento de dados, banco de dados em tempo real, notificações, autenticação de usuários, hospedagem de páginas, ferramentas de análises de desempenho e laboratório de teste. No recurso de Realtime Database os dados são armazenados em um banco de dados não relacional (NoSQL) com a estrutura de JSON e são rapidamente sincronizados com todos os clientes conectados em tempo real. No Storage é armazena as mídias do usuário, como áudio, imagens e vídeos. Por fim, a autenticação de usuários pode ser realizada por alguns provedores de identidades federais como: Google, Facebook e Twitter, além de possibilitar a criação de uma conta para da aplicação com e-mail e senha (FIREBASE, 2017c).

O Software Development Kit (SDK) do Firebase suporta o desenvolvimento nas seguintes linguagens de programação: C++, Java, JavaScript, Node.js, Objective-C e Swift (BATTISTELLI, 2017). Segundo Viana (2017), a plataforma Firebase contém quatro segmentos de serviços: Analytics, Develop, Grow e Earn, conforme ilustrados na Figura 1.

Figura 1 – Infraestrutura do Firebase



Fonte: Viana (2017, p. 1).

O Analytics é utilizado para gerar as métricas da aplicação e mensurar o comportamento do usuário mediante a captura automática de determinados eventos-chave que são relevantes para os negócios (VIANA, 2017). O SDK do Firebase captura mais de 500 eventos diferentes, das quais podem ser personalizados. Atualmente, o serviço só está disponível para Android e iOS e os resultados destes eventos podem ser visualizados em um painel em forma de gráficos estatísticos (FIREBASE, 2017d).

O Develop tem como objetivo poupar tempo dos desenvolvedores e auxiliar na construção de um aplicativo de qualidade gerando receitas ao negócio. Disponibiliza alguns recursos, são eles: Cloud Messaging, Authentication, Realtime Database, Storage, Hosting, Remote Config, Test Lab e Crash Reporting. No entanto, alguns destes recursos não são disponibilizados em todas as plataformas. O Test Lab funciona somente no Android, o Remote Config e Crash Reporting não tem suporte para web e o Hosting é exclusivo para web (VIANA, 2017).

O segmento Grow, segundo Viana (2017), são recursos voltados ao envolvimento e conquista dos usuários com a aplicação. Dentre os recursos existem as Notifications que podem ser enviadas a determinados grupos de usuários, Firebase Invites para indicações e compartilhamento de conteúdo por e-mail ou SMS, o Dynamic Links permite o acesso direto ao conteúdo do link no aplicativo nativo, a App Indexing garante a navegação direta para o conteúdo pesquisado no Google e o AdWords auxilia a apresentação dos anúncios direcionados aos usuários.

O último segmento Earn disponibilizada pelo recurso AdMob, permite monetizar a aplicação móvel com publicidade segmentada dentro do aplicativo, desta forma, gera receitas sem prejudicar a experiência do usuário. Os anúncios podem ser exibidos como banners ou vídeos e são adicionados perfeitamente aos componentes da UI, pois possuem redimensionamento inteligente e o desenvolvedor escolhe onde colocá-los (FIREBASE, 2017a).

O Firebase oferece um plano gratuito para aqueles que estejam interessados em conhecer seu funcionamento, além de outros planos flexíveis baseados no consumo de sua aplicação. O plano gratuito que foi utilizado neste trabalho também chamado de Spark, inclui todas as funcionalidades de um plano pago, porém com algumas restrições: no item (a) Realtime Database pode-se ter até 100 conexões simultâneas com armazenamento de informação de 1GB; no (b) Storage tem-se 5GB de armazenamento e 20 mil operações de upload e 50 mil operações de download por dia; no (c) Hosting apresenta domínio personalizado e tecnologia de segurança (Secure Socket Layer - SSL); e o (d) Test Lab é um serviço que está disponível por 10 dias inicialmente. No entanto, a integração com o Google Cloud Platform não é possível nesta versão gratuita (FIREBASE, 2017b).

2.4 TRABALHOS CORRELATOS

Nesta seção são apresentados três trabalhos correlatos, que possuem características semelhantes à proposta deste trabalho. A seção 2.4.1 descreve o EasyEdu (CORSO, 2017), um editor web de jogos para auxílio da educação. A seção 2.4.2 aborda um aplicativo multiplataforma de escrita colaborativa (SILVA, 2015). Por fim, a seção 2.4.3 detalha o aplicativo Tangram (PLAYOSMO, 2017a), que é um jogo educativo de encaixe para crianças.

2.4.1 EasyEdu

O trabalho elaborado por Corso (2017) desenvolveu uma página web na qual pode-se construir jogos educativos para os alunos aprenderem brincando. Desta forma, leva o uso da tecnologia para dentro da sala de aula convencional.

O EasyEdu é um editor de jogos voltado a auxiliar os alunos e educadores na obtenção de conhecimentos, como pode ser observado na Figura 2. Neste contexto, os educadores podem usar layouts customizáveis para criar atividades que foram abordadas em sala de aula e depois compartilhar com seus alunos, possibilitando que os mesmos exercitem o conteúdo de forma lúdica.

Segundo o autor, existem dois layouts de jogos disponíveis que podem ser utilizados para construção de atividades. Estes layouts permitem adicionar arquivos multimídia como imagem e som. As atividades desenvolvidas estão preparadas para serem utilizadas em mesas interativas. Por este motivo, o editor apresenta recursos para multitoque simultâneos e atividades de partida simples ou dupla, explorando a competitividade entre os alunos.

O compartilhamento das atividades com os alunos é feito através de QRCode, com o qual os alunos podem importar os jogos exportados pelo professor. Todas as atividades são armazenadas em uma pasta no Google Drive. Porém, segundo Corso (2017), o arquivo de QRCode teve que ser armazenado em um repositório público, já que o Google Drive recusa este tipo de arquivo.

Figura 2 – Editor EasyEdu

Fonte: Corso (2017, p. 63).

O editor EasyEdu, como relatado no experimento realizado com os acadêmicos da 7ª fase do curso de Pedagogia da FURB, é uma ferramenta diferenciada para auxiliar no processo de ensino e aprendizagem. Porém, apresenta alguns empecilhos que podem trazer problemas quanto a utilização do editor, visto que o professor necessita ter uma conta de e-mail no Google Drive devido ao armazenamento dos arquivos. O desenvolvimento e compartilhamento das atividades dependem da conexão com a internet, portanto o editor não trabalha off-line, apesar de apresentar atividades prontas que podem ser jogadas e não precisam ser baixadas. O site é responsivo, mas em dispositivos móveis o editor não se comporta do jeito que é esperado devido ao tamanho da tela e os recursos nativos.

2.4.2 Aplicativo multiplataforma de escrita colaborativa em tempo real

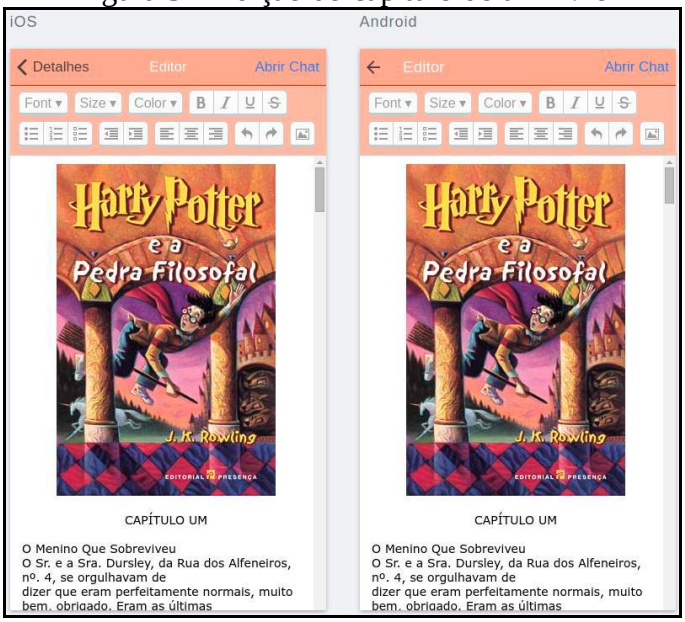
Silva (2015) descreve um aplicativo multiplataforma de escrita colaborativa. O aplicativo proporciona ao escritor escrever seus livros em contato com os leitores que podem

avaliar o conteúdo que está sendo escrito em tempo real e assim colaborar com o escritor para melhorar o nível do livro e alavancar sua carreira. O leitor consegue interagir com os autores dos livros através de um chat que tem como intuito aproximar o leitor do autor do livro para proporcionar uma rica troca de informações e feedbacks.

O desenvolvimento do aplicativo se guiou pela metodologia Mobile First. Analisando esta metodologia, foi escolhido o *framework* Ionic que permite o desenvolvimento de aplicações híbridas para múltiplas plataformas móveis utilizando as mesmas regras de negócio de uma aplicação totalmente web, além de proporcionar a possibilidade de ampliação futura sem limitações e perda de qualidade (SILVA, 2015). O autor expõe que para o serviço de persistência e recuperação de dados em tempo real, utilizou-se a ferramenta Firebase e para a sincronização dos dados a API do Firepad.

Este trabalho permite ao escritor escrever vários livros ao mesmo tempo de uma forma organizada. A interface possui os principais recursos para edição de textos. Na Figura 3 são exibidos os recursos de texto e edição do primeiro capítulo de um livro. Na parte superior é encontrado o botão de chat.

Figura 3 – Edição do capítulo de um livro



Fonte: Silva (2015, p. 72).

Segundo Silva (2015, p.78), houve pontos negativos referentes à interação pois inicialmente se pensava em desenvolver uma sessão de comentários e não um chat, mas devido à complexidade ficou impossível realizá-la. Outro ponto foi a necessidade do utilizador estar sempre conectado com à internet para conseguir visualizar e editar os textos, já que os mesmos são armazenados na nuvem.

2.4.3 Tangram - OSMO

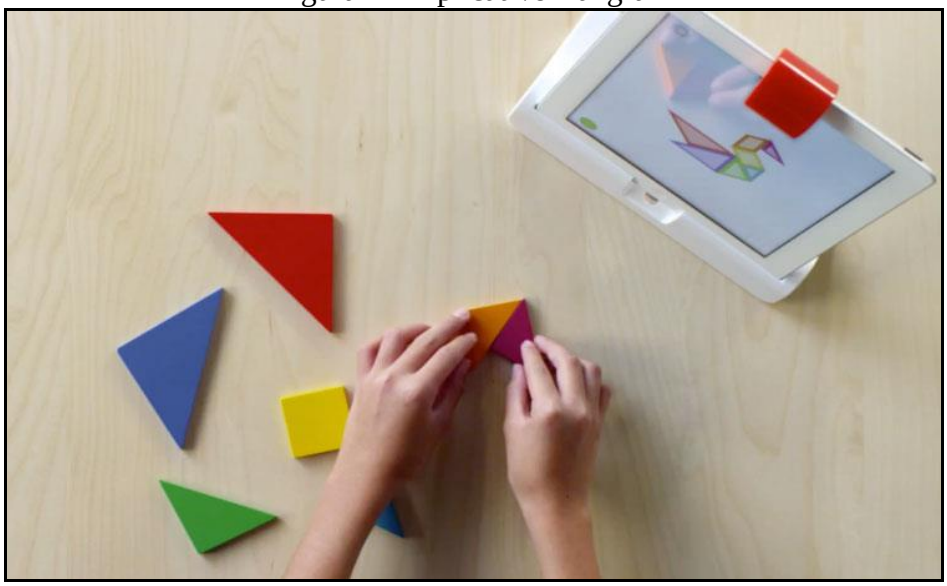
A fabricante de brinquedos digitais educativos, Tangible Play, fundada em 2013 (CRUNCHBASE, 2017) por dois ex-engenheiros do Google (PEREZ, 2014), lançaram uma nova linha de brinquedos chamada Osmo, projetada para crianças de 5 a 12 anos de idade (PLAYOSMO, 2017a). A Osmo inova nos seus aplicativos educacionais e na forma como as pessoas, em especial as crianças, interagem com os dispositivos tecnológicos, misturando as brincadeiras do mundo real com a tela do dispositivo, despertando assim a curiosidade e auxiliando o desenvolvimento de múltiplas habilidades (BRUSTEIN, 2014).

Atualmente, existem oito jogos disponíveis para a Osmo (PLAYOSMO, 2017a). Todos os jogos digitais são baseados em torno de uma tecnologia que eles chamam de inteligência artificial reflexiva, utilizada em conjunto com o kit Osmo. Este kit inclui uma base para o apoio do tablet na posição vertical e uma peça plástica vermelha (espelho refletor) que é encaixada na câmera frontal do dispositivo, tornando-a reflexiva para capturar tudo o que é posicionado na superfície plana à frente do dispositivo. Sendo assim, os aplicativos compatíveis podem reconhecer as peças físicas do jogo (CANALTECH, 2014).

Um dos oito jogos é o aplicativo Tangram, que é um jogo de quebra-cabeça de origem chinesa que contém sete peças de madeira colorida (2 triângulos grandes, 1 triângulo médio, 2 triângulos pequenos, 1 paralelogramo e 1 quadrado). Essas peças precisam ser encaixadas sem sobrepô-las para formar as figuras, tornando assim o jogo uma ótima maneira de exercitar habilidades de resolução de problemas espaciais e visuais (PLAYOSMO, 2017b). Segundo a Playosmo (2017b), o aplicativo contém 500 quebra-cabeças diferentes para serem montados.

Na Figura 4 pode-se observar as mãos da criança jogando o jogo Tangram e as peças que acompanham o kit Osmo. No jogo a criança deve encaixar as peças físicas na frente do dispositivo correspondentes com a imagem colorida apresentada na tela. À medida que ela avança nos níveis, o jogo vai ficando mais difícil até que ele somente mostre a silhueta da figura totalmente montada.

Figura 4 – Aplicativo Tangram



Fonte: Ilex (2014, p. 1).

Os aplicativos são exclusivos para sistema operacional iOS, mas será expandido para outras plataformas nos próximos anos e desenvolvido mais jogos para o crescimento da Osmo (ROETTGERS, 2016). Segundo a autora Perez (2014), os aplicativos não funcionam em qualquer superfície, pois dependendo da iluminação e do contraste da superfície com as peças, o jogo pode capturar algumas peças incorretamente.

Segundo Playosmo (2017a), a Osmo foi adotada em mais de 22 mil escolas espalhadas por 42 países do mundo, onde os alunos experimentam, exploram, criam e colaboram com os aplicativos. Na opinião dos educadores os jogos ajudam na aprendizagem social e emocional de uma criança, enquanto alguns ensinam conceitos diferentes, como inteligência espacial e pensamento criativo (PEREZ, 2014).

3 DESENVOLVIMENTO DO EDITOR

Neste capítulo são descritas as etapas do desenvolvimento do editor de jogos multiplataforma. Na seção 3.1 são apresentados os Requisitos Funcionais (RF) e os Requisitos Não Funcionais (RNF). A seção 3.2 demonstra a especificação do projeto através de diagramas e modelos que representam as principais funcionalidades. A seção 3.3 detalha a implementação, descrevendo e exibindo os trechos relevantes do código, ferramentas utilizadas e a operacionalidade em nível de usuário. Por fim, a seção 3.4 aborda os resultados obtidos e a análise do experimento realizado.

3.1 REQUISITOS

A seguir estão listados os Requisitos Funcionais e Requisitos Não Funcionais atendidos pelo editor.

O editor deverá ter os seguintes requisitos:

a) permitir aos usuários o cadastramento e autenticação no aplicativo (RF);
b) possibilitar ao usuário a criação de álbuns de jogos (RF);
c) permitir ao usuário a criação de jogos (RF);
d) permitir ao usuário editar e excluir ou mover um jogo (RF);
e) permitir que o usuário adicione conteúdo multimídia aos jogos (RF);
f) possibilitar ao usuário poder jogar os jogos (RF);
g) propiciar o compartilhamento dos jogos construídos com outros usuários cadastrados no sistema (RF);
h) permitir que o usuário possa tirar fotos com câmera do dispositivo, se disponível (RF);
i) trabalhar com mídias de imagem .JPEG (RNF);
j) funcionar nas plataformas Android, iOS e Web (RNF);
k) ser implementado nas linguagens HTML5, SCSS e TypeScript (RNF);
l) utilizar o *framework* Ionic para o desenvolvimento do aplicativo (RNF);
m) utilizar a plataforma Angular (RNF);
n) utilizar a plataforma Apache Cordova para ter acesso aos recursos nativos dos dispositivos (RNF);
o) utilizar o SDK Firebase, que fornece vários serviços de *back-end* para aplicações multiplataforma (RNF);
p) suportar múltiplas conexões simultâneas (RNF);
q) ser desenvolvido no ambiente de desenvolvimento Visual Studio Code (RNF).

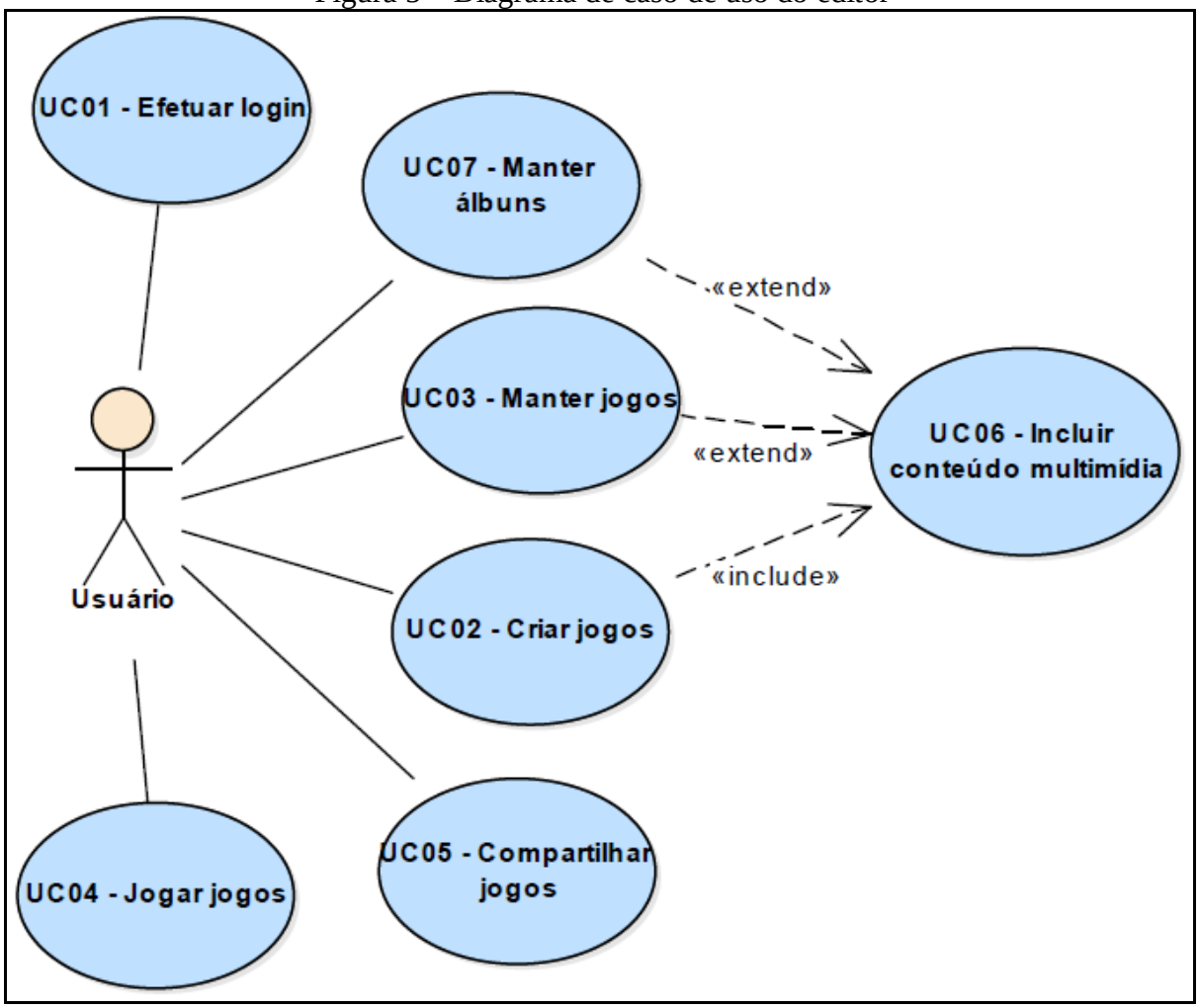
3.2 ESPECIFICAÇÃO

Para o entendimento das funcionalidades e arquitetura do aplicativo, esta seção apresenta os diagramas da Unified Modeling Language (UML) desenvolvidos na ferramenta Draw.io e Enterprise Architect. Nas próximas seções são apresentados os diagramas de caso de uso, de classes, de atividade e de arquitetura da aplicação.

3.2.1 Diagrama de caso de uso

Nesta subseção foram desenvolvidos os casos de uso levando em consideração a elicitação dos RF e RNF da aplicação. Como ilustrado na Figura 5, esses casos de usos descrevem de forma simplificada as funcionalidades que podem ser realizadas pelo usuário no editor. Os detalhamentos dos casos de uso estão disponíveis no Apêndice A.

Figura 5 – Diagrama de caso de uso do editor



Fonte: elaborado pelo autor.

3.2.1.1 Matriz de rastreabilidade RF x UC

O Quadro 1 apresenta a matriz de rastreabilidade entre os requisitos da seção 3.1 e os casos de uso da Figura 5.

Quadro 1 – Matriz de rastreabilidade

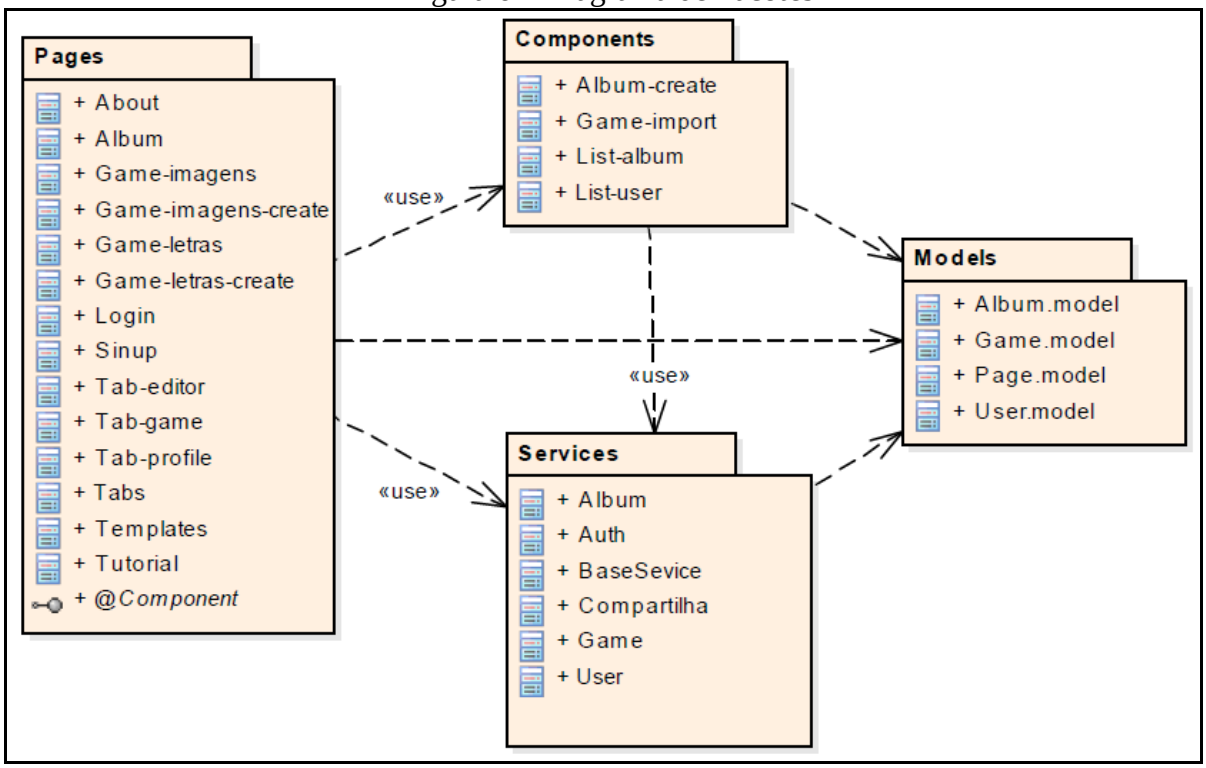
Requisitos funcionais	Caso de Uso
permitir aos usuários o cadastramento e autenticação no aplicativo	UC01
possibilitar ao usuário a criação de álbuns de jogos	UC07
permitir ao usuário criação de jogos	UC02
permitir ao usuário editar e excluir ou mover um jogo	UC03
permitir que o usuário adicione conteúdo multimídia aos jogos	UC02 e UC03
possibilitar ao usuário poder jogar as atividades	UC04
permitir o compartilhamento dos jogos construídos com outros usuários cadastrados no sistema	UC05
permitir que o usuário possa tirar fotos com câmera do dispositivo, se disponível	UC06

Fonte: elaborado pelo autor.

3.2.2 Diagrama de classe

Na Figura 6 são exibidas as relações entre os pacotes da aplicação. Estes pacotes servem para agrupar as classes que constituem as páginas e os serviços do editor gerando uma melhor visualização do escopo geral. Em seguida, são detalhadas as classes destes pacotes.

Figura 6 – Diagrama de Pacotes

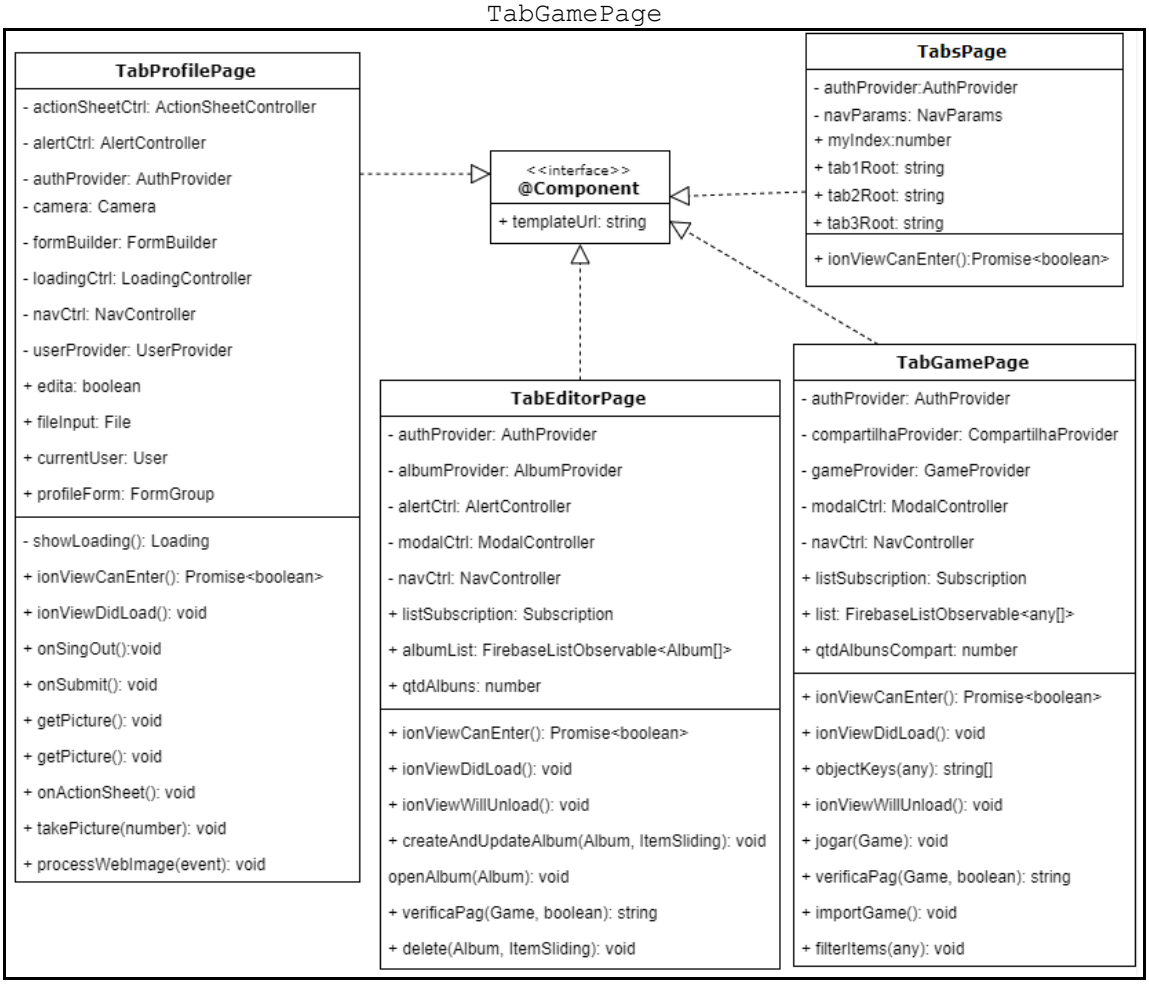


Fonte: elaborado pelo autor.

3.2.2.1 Diagrama de classe do pacote `pages`

Nesta subseção é apresentado o pacote `pages` que possui as classes anotadas como `@Component`. Estas classes são responsáveis pela exibição, navegação e controle das páginas desenvolvidas. Os diagramas de classes podem ser visualizados nas Figura 7, Figura 8, Figura 9 e Figura 10.

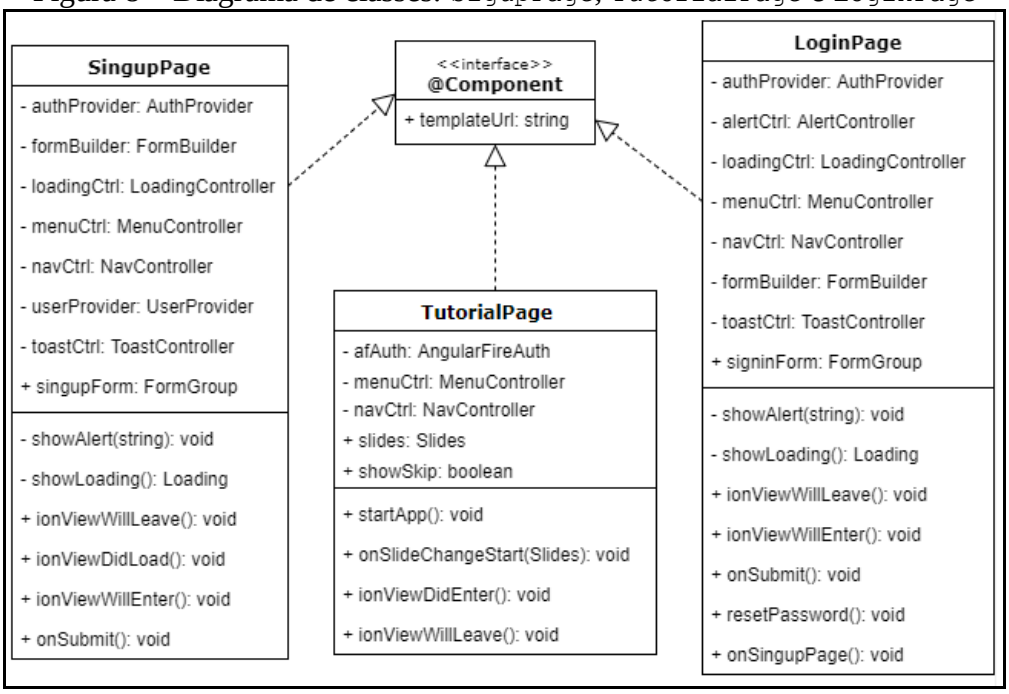
Figura 7 – Diagrama de classes: `TabsPage`, `TabProfilePage`, `TabEditorPage` e `TabGamePage`



Fonte: elaborado pelo autor.

A Figura 7 representa as páginas encontradas na barra de navegação para acesso rápido. Nela tem-se a classe `TabsPage` que gerencia as páginas `TabProfilePage`, `TabGamePage` e `TabEditorPage`, determinando qual delas será exibida para o usuário. A entidade `TabProfilePage` apresenta as informações de perfil do usuário, assim como a possibilidade de edição e desautenticação da aplicação. A `TabGamePage` lista todos os jogos desenvolvido no editor que podem ser rapidamente jogados. Por fim, a `TabEditorPage` exibe os álbuns que contém os jogos que foram desenvolvidos.

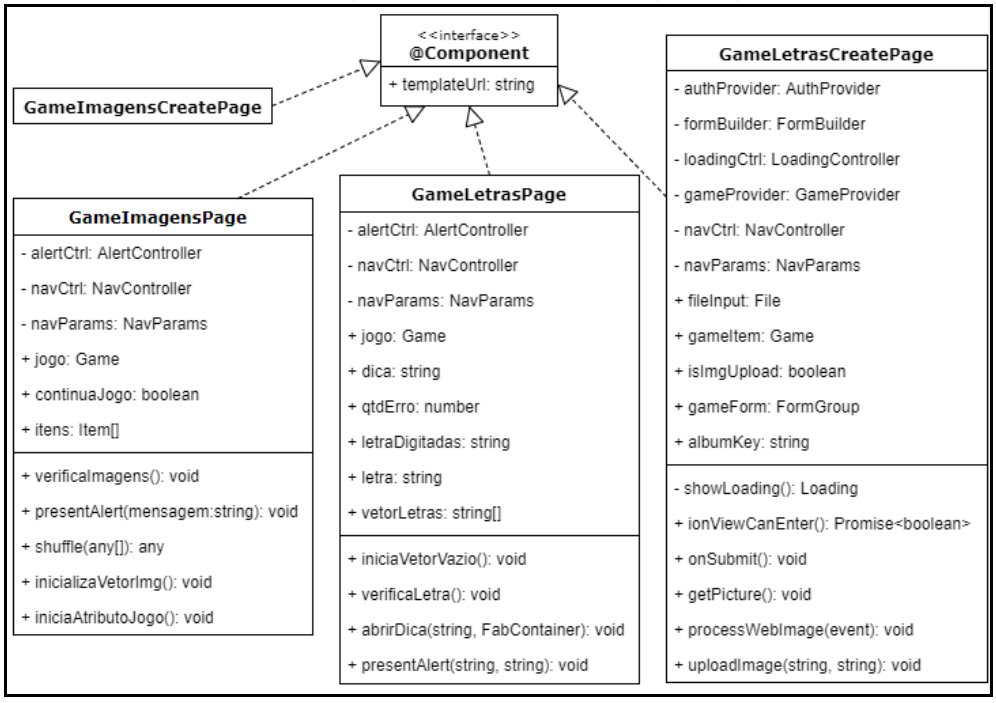
Figura 8 – Diagrama de classes: `SigupPage`, `TutorialPage` e `LoginPage`



Fonte: elaborado pelo autor.

A `TutorialPage` é a página inicial da aplicação responsável por fornecer algumas instruções sobre como utilizar o editor de jogos multiplataforma. A `LoginPage` é responsável por fornecer uma interface gráfica para que o usuário possa entrar no aplicativo através de seu usuário e senha. Caso não tenha uma conta, o usuário pode ir para a `SingupPage` onde terá a possibilidade de se cadastrar na aplicação.

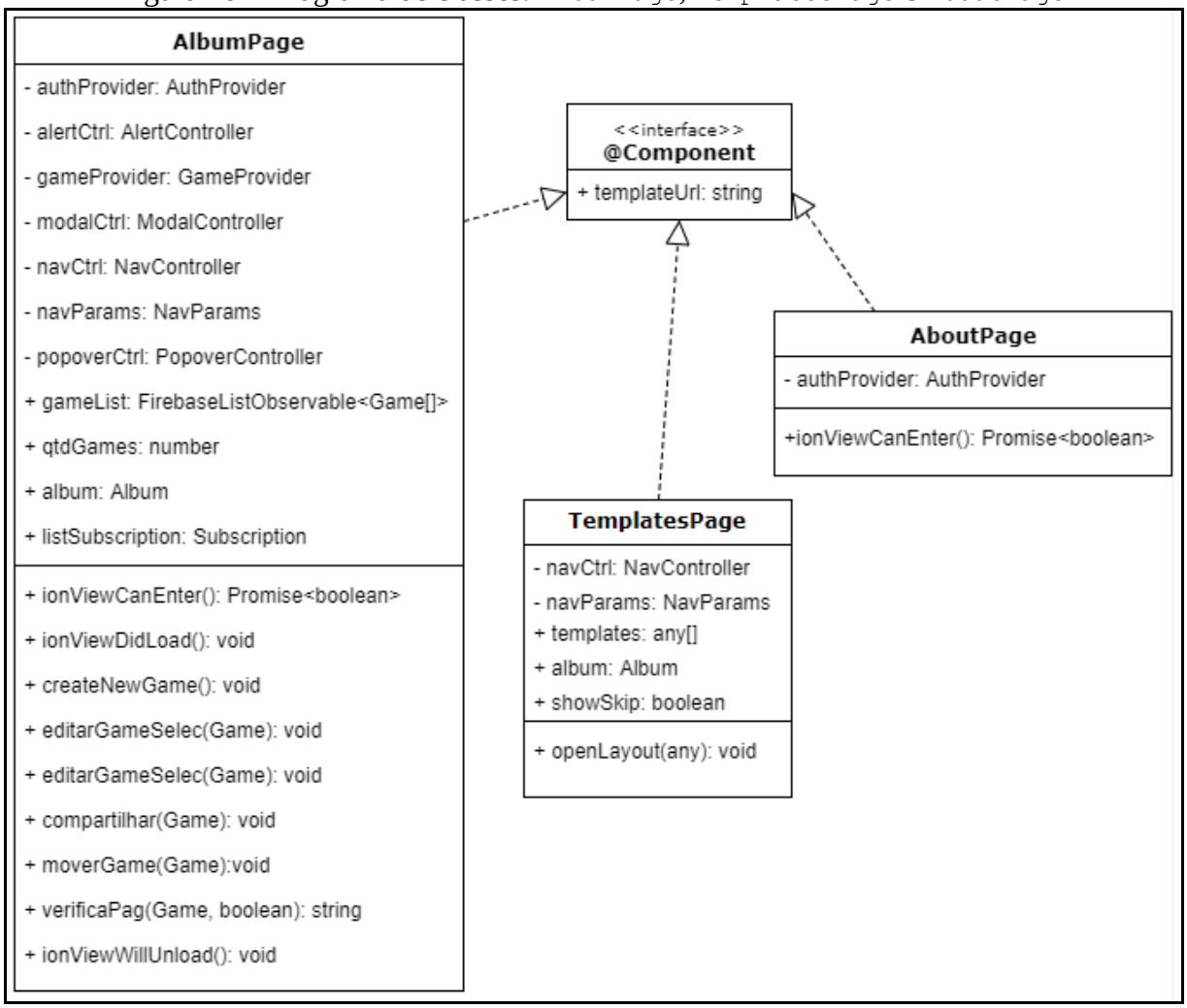
Figura 9 – Diagrama de classes: `GameLetrasCreatePage`, `GameLetrasPage`, `GameImagensCreate` e `GameImagemPage`



Fonte: elaborado pelo autor.

As entidades `GameLetrasCreatePage` e `GameImagensCreatePage` são responsáveis por apresentar uma interface para o usuário entrar com as informações dos jogos e arquivos multimídia, criando desta forma as atividades que foram abordadas em sala de aula. As classes `GameLetrasPage` e `GameImagensPage` representam a lógica dos respectivos jogos. O jogo de imagem não está totalmente implementado, o que pode acarretar em algumas mudanças nos atributos ,métodos e a impossibilidade de construí-lo. A `GameImagensCreatePage` foi abstraída da Figura 9, pois é semelhante a `GameLetrasCreatePage`.

Figura 10 – Diagrama de classes: `AlbumPage`, `TemplatesPage` e `AboutPage`



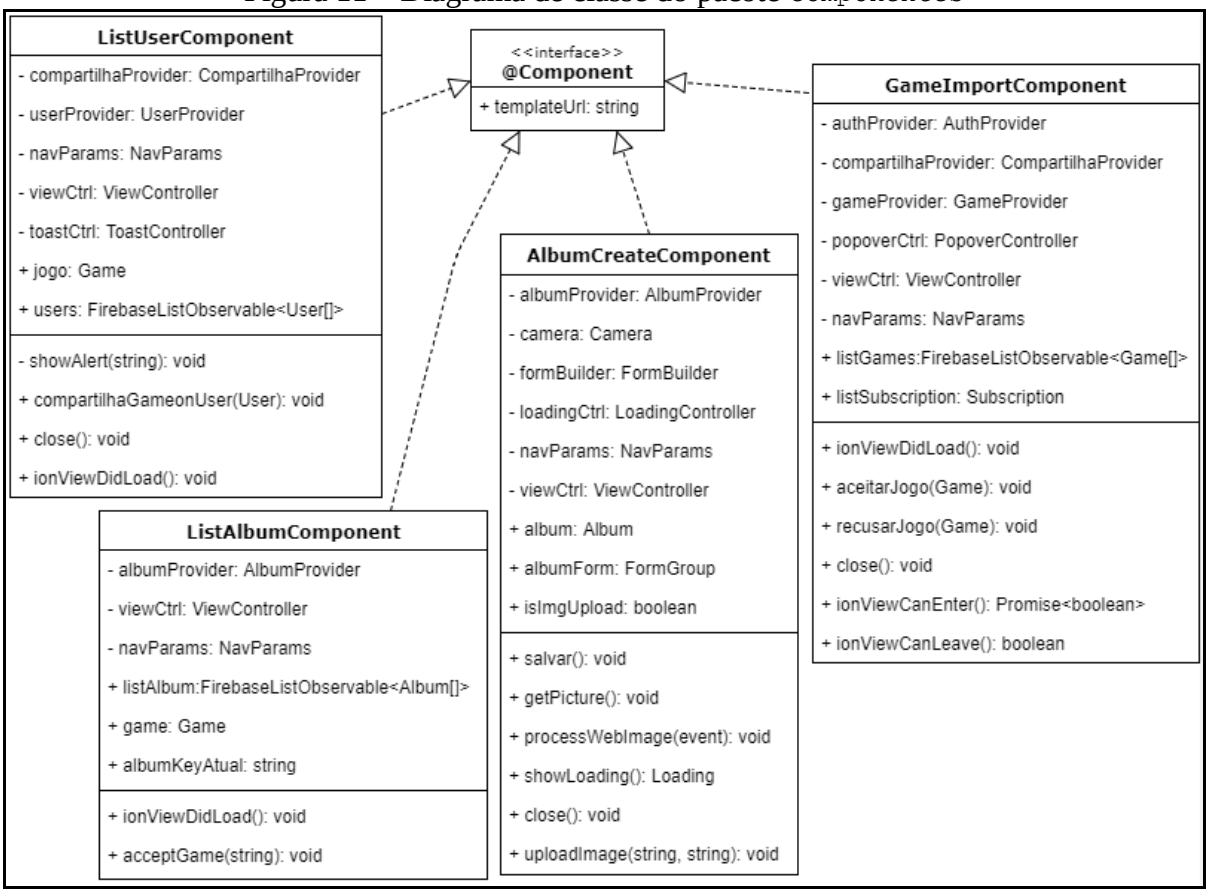
Fonte: elaborado pelo autor.

A página `AlbumPage` concentra várias funções da aplicação, pois é nesta página que são criados os jogos. Esses métodos são responsáveis pela criação, edição, exclusão, compartilhamento e movimentação dos jogos. A `TemplatesPage` apresenta os *templates* disponíveis para a criação dos jogos e a `AboutPage` expõe as informações do EdiBox: editor de jogos multiplataforma.

3.2.2.2 Diagrama de classe do pacote `componentes`

Na Figura 11 são apresentados os componentes da aplicação. Os componentes podem ter o mesmo comportamento de uma página ou fazem parte de uma. O componente `ListtUserComponent` e `ListAlbumComponent` listam todos os usuários e os álbuns respectivamente. O `AlbumCreateComponent` apresenta uma interface para o cadastramento de um novo álbum e o `GameImportComponent` exibe em uma interface com os jogos compartilhados, na qual podem ser aceitos ou recusados pelo usuário.

Figura 11 – Diagrama de classe do pacote `componentes`



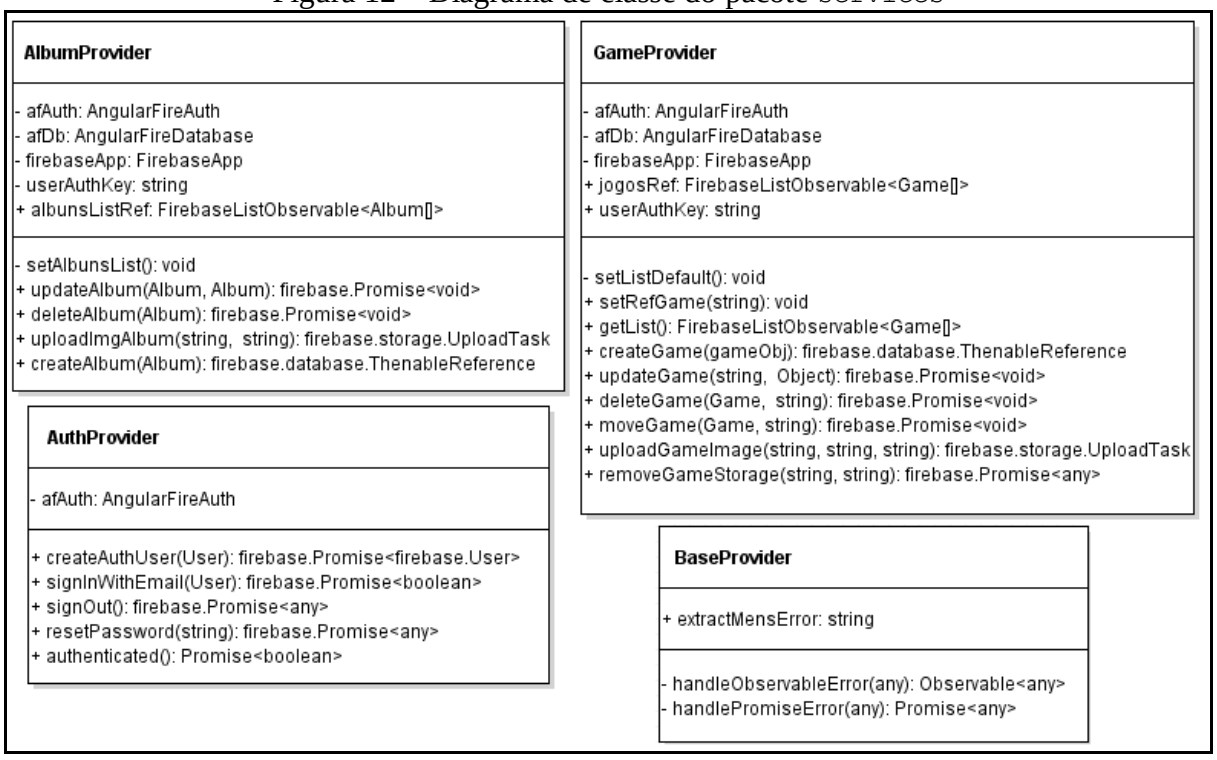
Fonte: elaborado pelo autor.

3.2.2.3 Diagrama de classe do pacote `services`

O pacote `services` possui os serviços implementados ao longo do desenvolvimento do editor de jogos multiplataforma, que podem ser acessados por qualquer classe do projeto através do padrão de projeto de injeção de dependência. Na Figura 12 são apresentados os serviços de `GameProvider` e `AlbumProvider` são responsáveis por buscar, criar, excluir e atualizar as informações dos jogos e álbuns, respectivamente, no banco de dados do Firebase. Além disto, os serviços realizam o *update* e exclusão das imagens do *storage* do Firebase. O

serviço `baseProvider` extrai a mensagem de erro recebida pelo servidor de back-end, estas mensagens podem ser do tipo *Observable* ou *Promise* e são convertidas para texto. O serviço `AuthProvider` realiza a autenticação do usuário, criação de novas contas e redefinição de senhas esquecidas.

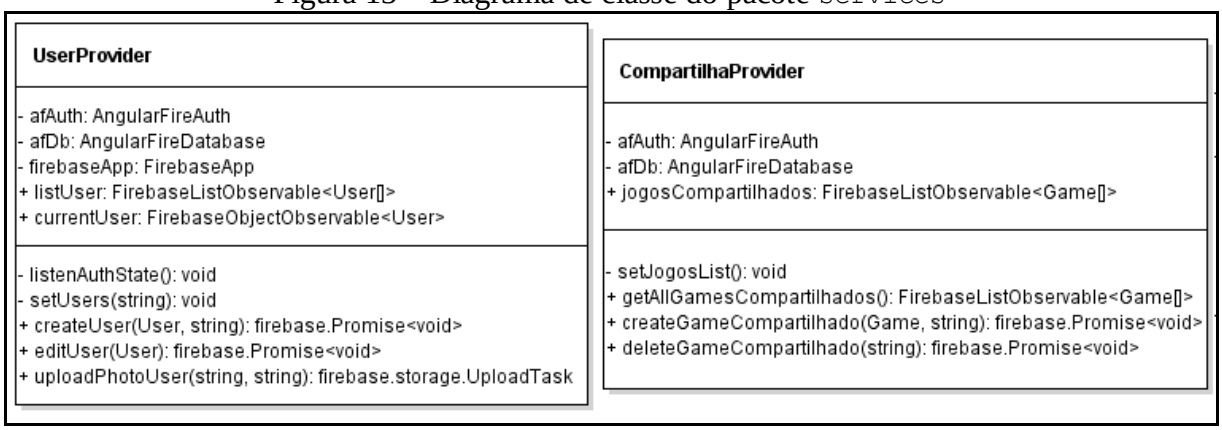
Figura 12 – Diagrama de classe do pacote `services`



Fonte: elaborado pelo autor.

A Figura 13, no serviço `CompartilhaProvider` é representado o compartilhamento no qual possui os métodos para aceitação e exclusão dos jogos. O método `createGameCompartilhado` recria o jogo no banco de dados com a chave de acesso do usuário que irá receber o jogo, então o usuário que recebe pode editar o jogo sem alterar o original. O `UserProvider` cadastra o usuário no banco de dados com o mesmo `token` de autenticação, além de retornar as informações do mesmo permitindo atualizá-las.

Figura 13 – Diagrama de classe do pacote `services`



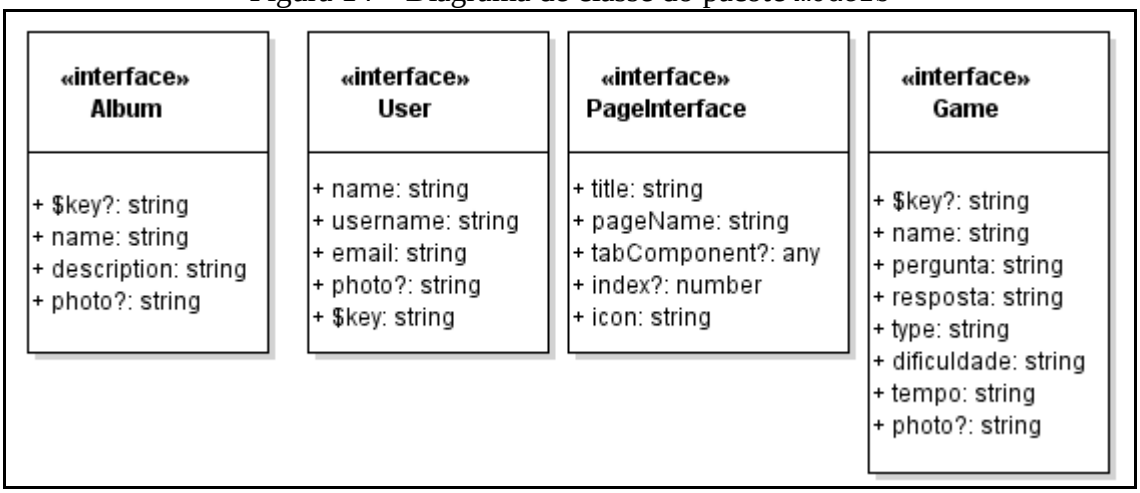
Fonte: elaborado pelo autor.

3.2.2.4 Diagrama de classe do pacote `models`

O pacote `models` possui as entidades utilizadas pelo editor multiplatafoma, como pode ser visto na Figura 14. As `interfaces` são as assinaturas de todos os atributos das principais entidades, garantindo que todos os jogos, álbum e usuários contenham as mesmas informações armazenadas no banco de dados.

As classes `Album`, `User` e `Game` possuem chaves de acesso `Key` que utilizam o operador `elvis` (“`?`”), fazendo com que o atributo não seja obrigatório inicialmente, por isto quem define este atributo geralmente é o servidor. Na classe `PageInterface` pôde-se ver o atributo `index` responsável pela navegação na barra inferior do aplicativo, já outras páginas que não estão representadas na barra não terão este atributo. Devido a aplicação conter apenas um jogo, temos apenas uma interface de jogo.

Figura 14 – Diagrama de classe do pacote `models`



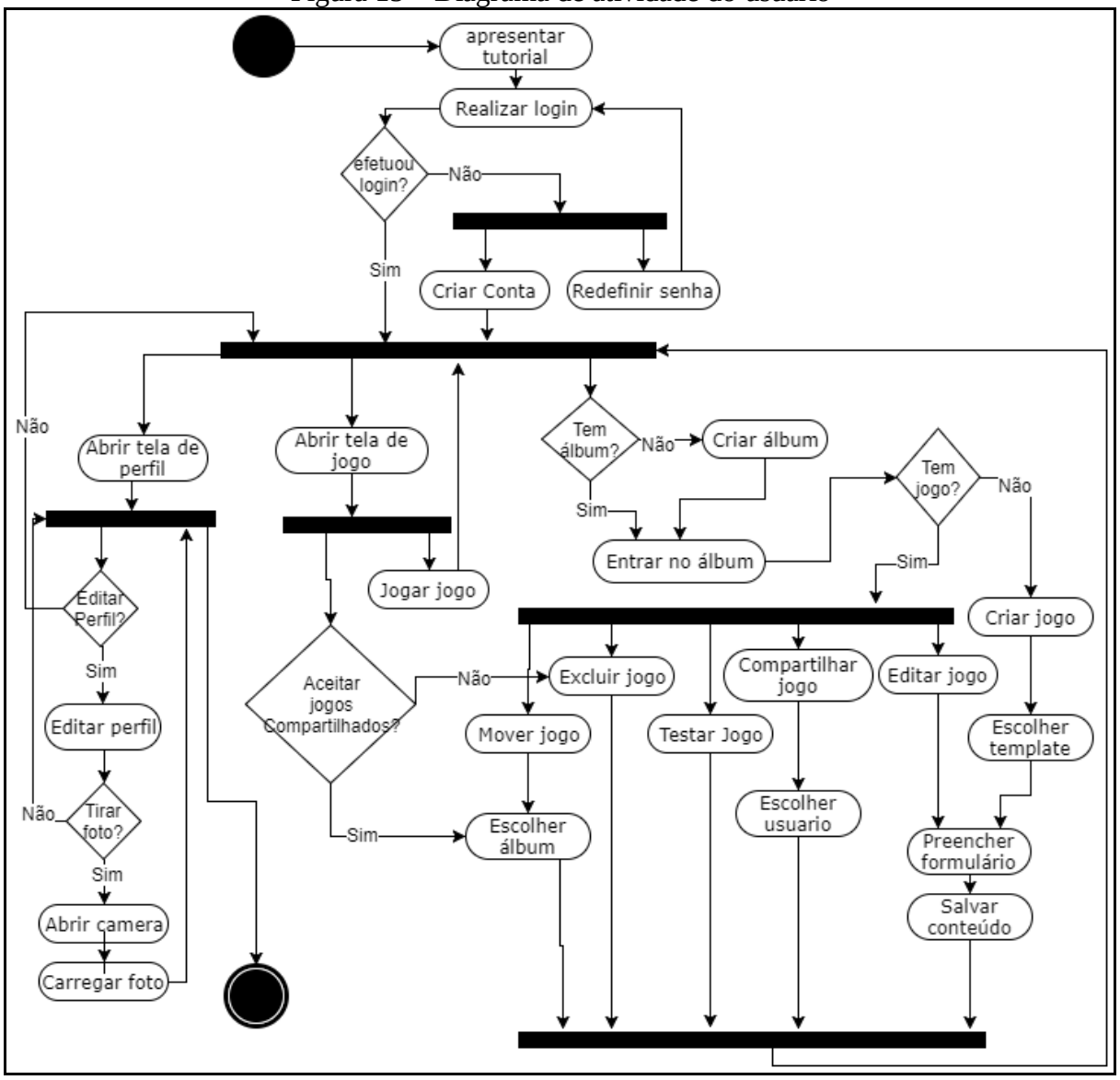
Fonte: elaborado pelo autor.

3.2.3 Diagrama de atividade

Na Figura 15 é exibido o diagrama de atividades, com as ações que o usuário pode realizar no editor. Inicialmente são apresentadas as rotinas de autenticação, criação e redefinição de senha. Logo após o usuário se autenticar na aplicação, ele poderá escolher três caminhos. A primeira, a do editor que é a principal rotina da aplicação, onde poderá ser criado editado ou escolhido um álbum. Entrando no álbum são exibidas todas as opções possíveis para criação de um jogo como: criar novo jogo a partir do *template* disponível, editar, testar, excluir, mover para outro álbum ou compartilhar o jogo com outros usuários. Na segunda opção, a tela de jogo, é possível jogar os jogos ou importar os jogos compartilhados. Por fim,

na terceira tela, a de perfil, pode ser editado o seu perfil do usuário, adicionando uma foto se desejar. Também por esta tela é possível sair do aplicativo.

Figura 15 – Diagrama de atividade do usuário

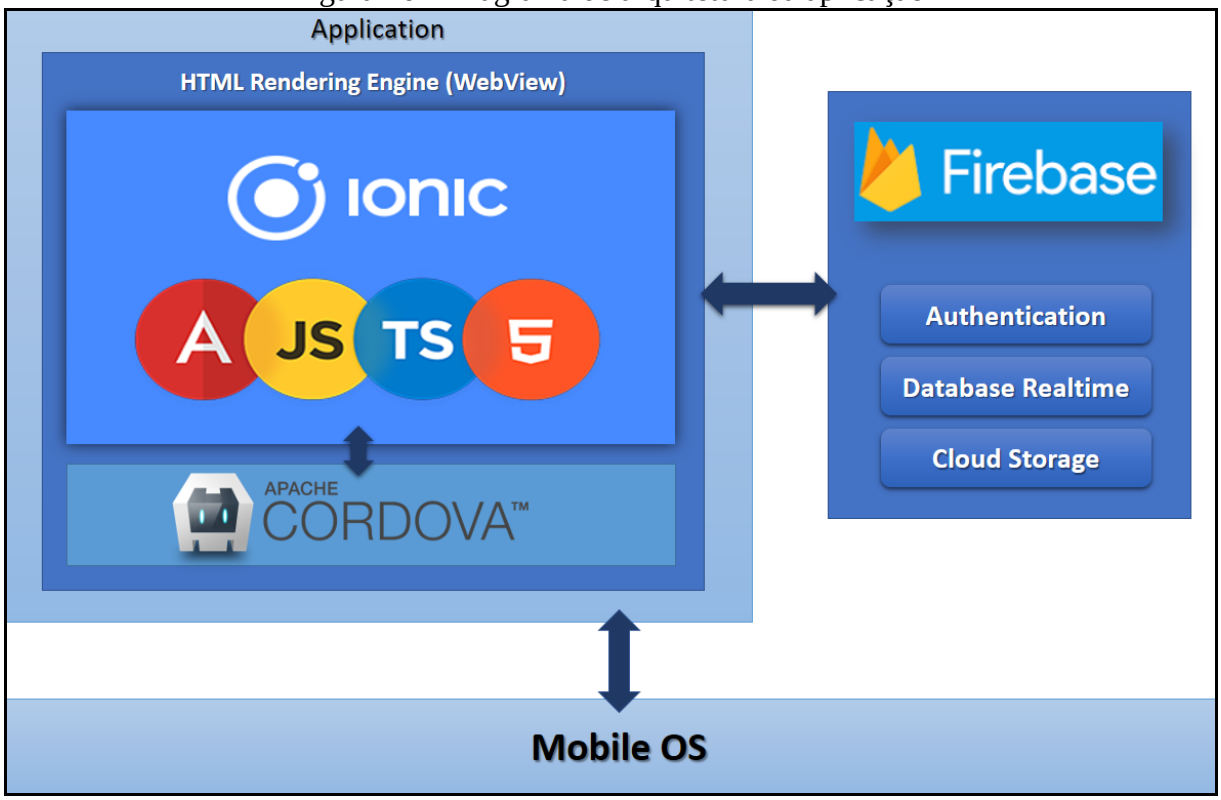


Fonte: elaborado pelo autor.

3.2.4 Diagrama de arquitetura

Nesta subseção é apresentado o diagrama de arquitetura do editor de jogos como exibido na Figura 16.

Figura 16 – Diagrama de arquitetura da aplicação



Fonte: elaborado pelo autor.

Para entender a estrutura principal do trabalho, o diagrama de arquitetura apresenta as camadas principais do editor multiplataforma. A aplicação utiliza os serviços de back-end do Firebase como a autenticação, banco de dados e armazenamento e o Ionic como front-end.

Descrevendo a construção da aplicação de dentro para fora, mostra que a aplicação foi desenvolvida em Ionic, e contém várias tecnologias web para sua implementação, sendo estas tecnologias o Angular, JavaScript, TypeScript e HTML5. Para a integração com os dispositivos o Ionic necessita do apoio do Cordova que faz a comunicação com os recursos nativos dos dispositivos móveis. Seguindo com as camadas, toda esta estrutura web é encapsulada em uma *webview*, fazendo com que a aplicação execute em dispositivos móveis.

3.3 IMPLEMENTAÇÃO

Nesta seção são mostradas as técnicas e ferramentas utilizadas para o desenvolvimento do projeto e a operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

Para disponibilizar o editor de jogos para o maior número de pessoas e dispositivos optou-se pela implementação utilizando o *framework* Ionic 3 com o SDK Firebase de *back-end* que garante o acesso aos arquivos de qualquer lugar ou dispositivo. Além do mais, o Ionic

utiliza como componentes principais as plataformas Angular e Apache Cordova. O Angular é responsável por toda a parte web e o Cordova garante a integração com os recursos nativos dos dispositivos, gerando assim, uma aplicação híbrida de qualidade já que estes componentes são amplamente utilizados.

Diante disso, pôde-se desenvolver a mesma aplicação para as principais plataformas do mercado, utilizando os mesmos recursos de linguagem desenvolvidos, porém algumas funcionalidades são diferentes da versão web devido aos *plugins* do Cordova.

3.3.1.1 Integração do Ionic com o Firebase

Ao iniciar o projeto pensava-se na funcionalidade do usuário construir seus jogos utilizando arquivos multimídia de sua preferência e armazenasse em algum lugar que fosse possível recuperá-las em qualquer dispositivo. Diante disto, o editor seria multiplataforma e foi descartada a possibilidade de armazenamento local a não ser para pequenas atividades que mais tarde com conexão à internet seriam sincronizadas. Portanto, identificou-se a necessidade de possuir um servidor *back-end* com serviços na nuvem que recebessem os arquivos, armazenasse e retornasse remotamente.

Para atender a este objetivo foram estudadas algumas alternativas como o S3 da Amazon, Azure da Microsoft e até mesmo o armazenamento no Google Drive, porém todos apresentaram algum empecilho. Em alguns casos a curva de aprendizagem era muito acentuada e o tempo de implantação era significativo para integrar com o Ionic. Optou-se pelos serviços da plataforma Firebase do Google que é paga, mas na sua versão *free* apresenta os recursos necessário para o início do projeto como: recursos de autenticação, banco de dados hospedado na nuvem com até 100 conexões simultâneas, 10 Gb de armazenamento, hospedagem de sites e notificações entre outros serviços não utilizados no projeto.

Para a utilização dos serviços é necessário cadastrar-se na plataforma do Firebase para criar um projeto e gerenciar os recursos e serviços que são ofertados pelo mesmo, com isto obtém-se as credenciais de integração que irão ser utilizadas no projeto. No projeto Ionic são necessários instalar os pacotes `angularfire2`, `firebase` e `promessa-polyfill` via CLI do Node JS. Logo após, deve ser configurado o módulo principal do projeto, como apresentado no Quadro 2 importando as configurações e serviços do servidor *back-end* e disponibilizar para todo o projeto do editor.

Quadro 2 – Configuração do módulo principal do projeto

```
01 ...
02 import { AngularFireAuthModule } from 'angularfire2/auth';
03 import { AngularFireDatabaseModule, AngularFireDatabase } from
04 'angularfire2/database';
05 import { AngularFireModule, FirebaseAppConfig } from 'angularfire2';
06
07 export const FIREBASE_APP_CONFIG: FirebaseAppConfig = {
08   apiKey: "xxxxxxxx",
09   authDomain: "xxxxxxxxx",
10   databaseURL: "xxxxxxx",
11   projectId: "xxxxxxxx",
12   storageBucket: "xxxxxxxxx",
13   messagingSenderId: "xxxxxxxxx"
14 };
15
16 @NgModule({
17   ...
18   imports: [
19     AngularFireAuthModule,
20     AngularFireDatabaseModule,
21     AngularFireModule.initializeApp(FIREBASE_APP_CONFIG),
22     BrowserModule,
23     IonicModule.forRoot(MyApp)
24   ],
25   ...
26 providers: [
27     AngularFireDatabase,
28     ...
29     StatusBar,
30     SplashScreen,
31     { provide: ErrorHandler, useClass: IonicErrorHandler },
32   ]
33 })
34 export class AppModule {}
```

Fonte: elaborado pelo autor.

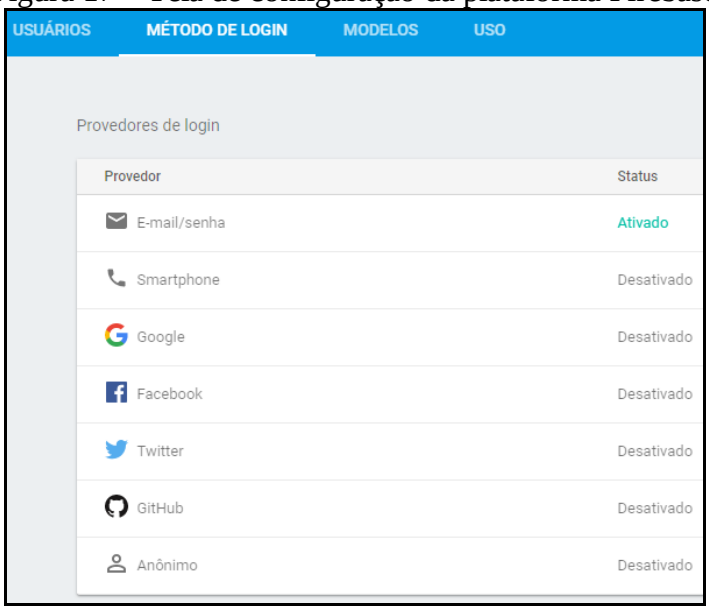
Devido ao Firebase ter suporte a aplicações nativas, as linhas 7 a 14 apresentam as credenciais de integração web já que o Ionic produz projetos híbridos. Nas linhas 19 a 21 são importados os módulos de autenticação, banco de dados e de inicialização do Firebase utilizados na aplicação e na linha 27 o serviço de banco de dados.

3.3.1.2 Autenticação do Firebase

No começo do projeto foi pensado em utilizar o serviço de autenticação do próprio Ionic para logar no aplicativo, porém segundo o diretor executivo Lynch (2017) o mesmo deixará de ser ofertado a partir do dia primeiro de fevereiro de 2018. Para deixar o projeto estável e utilizável após o término deste trabalho, optou-se por utilizar o Firebase que oferece os mesmos serviços de *login* com endereço de e-mail e senha, além de provedores de identidade federados como o Google, Facebook, Twitter e outros. Este projeto utiliza apenas a autenticação por e-mail e senha.

Considerando uma boa prática de programação foi desenvolvido um serviço que ficará responsável pela autenticação, desautenticação e criação do usuário. Na plataforma online do Firebase, apresentado na Figura 17 o método de *login*, nesta tela deve-se ativar o serviço desejado e gerenciá-lo, configurando as mensagens de e-mail, redefinição de senha, quantidades de acessos, domínios da aplicação e outras configurações de segurança. Já no aplicativo Ionic, após importar e configurar a integração com o servidor no módulo principal da aplicação, é só importar nas outras classes a dependência do serviço para utilizá-lo.

Figura 17 – Tela de configuração da plataforma Firebase



Fonte: elaborado pelo autor.

No Quadro 3 são exibidos os principais métodos responsáveis pela autenticação do usuário. O método `createAuthUser` cria um novo usuário, o método `signInWithEmail` autentica o usuário através do e-mail e senhas digitados, `signOut` desloga e o `resetPassword` envia uma notificação para o e-mail do usuário para alteração de senha. Os tratamentos de erros são realizados por uma outra classe abstrata que extrai a mensagem de erro enviada pelo servidor *back-end* removendo informações desnecessárias e elimina ter que realizar vários tratamentos de erros por todo o projeto.

Quadro 3 – Autenticação do usuário

```
01 ...
02 constructor(private afAuth: AngularFireAuth) {
03   super();
04 }
05
06 createAuthUser(user: { email: string, password: string }):
07 firebase.Promise<firebase.User> {
08   return this.afAuth.auth.createUserWithEmailAndPassword(user.email,
09 user.password)
10    .catch(this.handlePromiseError);
11 }
12
13 signInWithEmail(user: { email: string, password: string }):
14 firebase.Promise<boolean> {
15     return this.afAuth.auth.signInWithEmailAndPassword(user.email,
16 user.password)
17     .then((authUser: firebase.User) => {
18       return authUser != null;
19     }).catch(this.handlePromiseError);
20 }
21
22 signOut(): firebase.Promise<any> {
23   return this.afAuth.auth.signOut()
24     .catch(this.handlePromiseError);
25 }
26
27 resetPassword(email: string): firebase.Promise<any> {
28   return this.afAuth.auth.sendPasswordResetEmail(email)
29     .catch(this.handlePromiseError);
30 }
31 ...
```

Fonte: elaborado pelo autor.

3.3.1.3 Persistência de dados em tempo real

O banco de dados do Firebase é um banco não relacional conhecido como NoSQL. Os bancos NoSQL permitem uma escalabilidade mais barata e menos trabalhosa do que os bancos relacionais e trabalham com um esquema de chave/valor.

Os principais métodos responsáveis por buscar, criar, excluir e editar um álbum no banco de dados estão representados no Quadro 4. Estes métodos também são responsáveis por exibir os jogos e usuários, porém estão implementados em outros serviços por existirem particularidades.

Devido ao banco de dados ser atualizado em tempo real o método `setAlbunsList` traz uma lista de referências `observables` do tipo `álbum` ordenada pelo `nome` como pode ser visto na linha 5 a 10 (Quadro 4). Antes de buscar esta lista é realizado um teste para ver se o usuário está autenticado na aplicação, pois se houver alguma mudança de usuário esta referência aos álbuns mudará automaticamente garantindo a estabilidade da aplicação. Para todos os outros métodos esta referência precisa ser inicializada antes, já que ela representa o caminho dos dados no banco.

Para criação de um novo álbum no banco de dados, deve-se ter adquirido a referência dos álbuns anteriormente para simplesmente utilizar o comando `push` passando o objeto do tipo álbum como parâmetro e assim adicionar a esta lista que foi buscada. O Firebase gerenciará automaticamente este dado gerando uma chave exclusiva e cronológica para ele garantindo a consistência dos dados no banco. O método `updateAlbum` atualiza o nó da estrutura de banco de dados sem a necessidade de substituir todos os dados. Como mostrado na linha 22 (Quadro 4) é necessário somente passar o objeto e os campos que serão atualizados, no entanto nesta ocasião está sendo passado todo o objeto. O método `deleteAlbum` é simples, porém devido a utilização do serviço de *storage* do Firebase e por causa do álbum ser um agrupador de jogos, quando o álbum é excluído deve-se apagar todos os jogos vinculados a ele e os arquivos armazenados no *storage,* tornando-o mais complicado. Inicialmente é verificado se este álbum contém uma foto, para ser excluída do *storage*. Após esta verificação na linha 30 (Quadro 4) é realizada a exclusão do álbum utilizando o comando `remove` passando a chave do nó que será excluído. Se a exclusão ocorrer corretamente são eliminados todos os jogos do banco de dados também. Devido ao banco de dados estar desnormalizado precisa excluir em dois lugares para deixar o banco organizado.

Optou-se por desnormalizar o banco, para que a aplicação busque apenas os dados necessários e quando ela precisar, ou seja não é necessário buscar todos os dados de uma só vez, atrasando o carregamento da página e deixando a aplicação lenta. Como exemplo, se o usuário entrar somente na tela de perfil, a aplicação só buscará as informações do usuário e guardará na memória ignorando todo o resto.

Quadro 4 – Serviço persistência do álbum no banco de dados

```
01 ...
02 private setAlbunsList(): void {
03   this.afAuth.authState
04    .subscribe((authUser: firebase.User) => {
05    if (authUser) {
06      this.albunsListRef = this.afDb.list(`/albuns/${authUser.uid}`, {
07        query: {
08          orderByChild: 'name'
09        }
10       });
11       this.userAuthKey = authUser.uid;
12    }
13   });
14 }
15
16 createAlbum(album: Album): firebase.database.ThenableReference {
17   return this.albunsListRef.push(album)
18     .catch(this.handleObservableError);
19 }
20
21 updateAlbum(album, newObjGame: Album): firebase.Promise<void> {
22   return this.albunsListRef.update(album, newObjGame)
23     .catch(this.handlePromiseError);
24 }
25
26 deleteAlbum(album: Album): firebase.Promise<void> {
27   if (album.photo) {
28     this.removeAlbumStorage(album.$key);
29   }
30   return this.albunsListRef.remove(album.$key).then(() => {
31 this.afDb.object(`/games/${this.userAuthKey}/${album.$key}`).remove()
32    .catch(this.handlePromiseError);
33   }).catch(this.handlePromiseError);
34 }
35 ...
```

Fonte: elaborado pelo autor.

Existe uma outra forma de inserção e atualização dos dados no banco e esta foi utilizada na classe do usuário e pode ser visualizada no Quadro 5. Este método `createUser` retorna apenas a referência de um único objeto, visto que o comando `set` não pode ser utilizado em uma lista de `observables`, devido a necessidade de ter um caminho específico para a criação ou a edição do registro. Na edição, é apagado do banco de dados todas as informações lá contidas e gravadas as novas passadas por parâmetro e na criação deve-se ter o cuidado, pois o mesmo não cria automaticamente uma chave como o método `push` descrito acima.

Quadro 5 – Criação do usuário com método set

```
01 ...
02 createUser(user: User, uuid: string): firebase.Promise<void> {
03   return this.afDb.object(`/users/${uuid}`).set(user)
04     .catch(this.handlePromiseError);
05 }
06 ...
```

Fonte: elaborado pelo autor.

3.3.1.4 Abertura da câmera no dispositivo móvel

Na construção dos jogos, o usuário carregará imagens que podem ser fotos tiradas com a câmera do dispositivo móvel. Para que isto seja possível é utilizado o plugin `cordova-plugin-camera` da plataforma Apache Cordova e o `@ionic-native/câmera` do próprio Ionic para se ter acesso ao recurso da câmera do dispositivo independente do sistema operacional. O Quadro 6 apresenta o código fonte utilizado, tendo como parâmetro de entrada do método o `canSourceType`, que é a opção de abertura da câmera ou galeria. Na linha 2 é definido o objeto `cameraOptions` com as configurações da câmera disponíveis na documentação. A qualidade da imagem foi definida com o valor de 50, em uma escala de 0 a 100, para que a imagem tenha um tamanho de dados menor, visando acelerar a transferência da imagem para o servidor *back-end*, principalmente quando o usuário estiver utilizando uma conexão de internet limitada ou lenta. Na linha 11, o método `getPicture` abre o aplicativo de câmera nativa do dispositivo móvel com as configurações definidas anteriormente. Após o usuário tirar a foto, a imagem é passada como parâmetro para o formulário do jogo na linha 13 no formato `base64`, onde posteriormente será enviada ao servidor.

Quadro 6 – Método de abertura da câmera

```
01 takePicture(canSourceType: number): void {
02   let cameraOptions: CameraOptions = {
03     correctOrientation: true,
04     quality: 50,
05     destinationType: this.camera.DestinationType.DATA_URL,
06     encodingType: this.camera.EncodingType.JPEG,
07     mediaType: this.camera.MediaType.PICTURE,
08     sourceType: canSourceType
09   };
10
11   this.camera.getPicture(cameraOptions)
12     .then((imageData) => {
13       this.profileForm.patchValue({ photo: ''
14 + imageData });
15       this.isUpload = true;
16     }, (err: Error) => {
17       console.log('Erro ao abrir a câmera: ', err);
18   });
19 }
```

Fonte: elaborado pelo autor.

3.3.1.5 Armazenamento do Firebase

O armazenamento no *storage* utiliza uma estrutura semelhante ao do banco de dados, porém não existe tantos métodos para trabalhar com ele. Como visto no Quadro 7, os únicos métodos são de `upload` e `delete`. As linhas 4 e 5 (Quadro 7) referenciam o *storage* do Firebase. O comando da linha 6 (Quadro 7) `child` recebe como parâmetro o caminho para o arquivo no *storage*, sendo a última informação do caminho o nome do arquivo, na qual no

editor está sendo colocado como a chave do álbum e o último parâmetro do método é o `putString`, define o que será passando, no caso uma imagem com tipo `data/base64`, ou seja, texto. Neste último parâmetro pode ser passado diretamente um `file`. O método `removeAlbumStorage` usa a mesma estrutura do método acima com alteração no último comando. O comando `delete` apaga apenas um arquivo por vez, não tendo a opção de apagar uma pasta inteira, apresentando assim uma complexidade maior para manipulação de vários arquivos ao mesmo tempo.

Quadro 7 – Métodos de upload de arquivos

```
01 ...
02 uploadImgAlbum(imageUrl: string, albumId: string):
03 firebase.storage.UploadTask {
04   return this.firebaseApp.storage()
05     .ref()
06     .child(`albuns/${this.userAuthKey}/${albumId}`)
07     .putString(imageUrl, 'data_url');
08 }
09
10 removeAlbumStore(nameImage: string) {
11   this.firebaseApp.storage()
12     .ref()
13     .child(`albuns/${this.userAuthKey}/${nameImage}`)
14     .delete()
15     .catch(this.handlePromiseError);
16 }
```

Fonte: elaborado pelo autor.

Na página de profile, é implementado o método exibido no Quadro 8 que envia os arquivos para o servidor, chamando o serviço que retorna uma `UploadTask` explicado no Quadro 7. Esta tarefa tem três ações que são de ocorrência de erro no carregamento do arquivo, ocorrência de sucesso, quando o arquivo foi totalmente enviado para o *storage* sendo desta forma necessário atualizar o banco de dados com a Uniform Resource Locator (URL) da imagem, e a última ação possível é a de carregamento onde pode-se pausar o envio ou capturar o progresso do envio e apresentar uma barra de porcentagem para o usuário.

Quadro 8 – Enviar arquivos para armazenamento

```
01 uploudPhoto() {
02   let loading: Loading = this.showLoading();
03   let uploadTask =
04 this.userProvider.uploadPhotoUser(this.profileForm.get('photo').value,
05 this.currentUser.$key);
06
07   uploadTask.on('state_changed', (snapshot:
08 firebase.storage.UploadTaskSnapshot) => {
09   let uploadProgress = Math.round((snapshot.bytesTransferred /
10 snapshot.totalBytes) * 100);
11
12   }, (error: Error) => {
13     console.log("erro ao carregar imagem");
14   });
15
16   uploadTask.then((UploadTaskSnapshot:
17 firebase.storage.UploadTaskSnapshot) => {
18       this.profileForm.patchValue({ photo:
19 uploadTask.snapshot.downloadURL });
20       this.userProvider.editUser(this.profileForm.value);
21       loading.dismiss();
22       this.edita = !this.edita;
23   });
24 }
```

Fonte: elaborado pelo autor.

3.3.1.6 Principais funcionalidades do jogo de letras

O único jogo disponível para construção no editor é o jogo do tipo `letra`. Quando o jogo inicia, é chamando o método `iniciaVetorVazio` para remover os espaços em branco da resposta com `replace` colocando “-”, criando um vetor com o tamanho da resposta e inicializá-lo com "_" para as letras e "-" para os traços mantendo os números, acentuação e caracteres especiais. Estes caracteres são adicionados para que os campos do vetor mantenham o tamanho de uma letra.

O método principal `verificaLetra` é chamado quando o usuário digita alguma letra no `<input>` do HTML. Na linha 14 (Quadro 9) é verificado se o usuário digitou algum caractere, já que o mesmo será transformado para maiúscula com a função `toUpperCase` e se não tratado gerará um erro. Se o usuário digitar mais de um caractere a função pegará somente o primeiro caractere. Na linha 16 (Quadro 9) é verificado em maiúsculo se está letra existe na resposta, se não existir é contada como erro, já nas linhas 18 a 22 (Quadro 9) são realizados os tratamentos para apresentar as letras incorretas digitadas pelo usuário. Na linha 24 (Quadro 9) é verificado se o jogo terminou pela quantidade de erros que é determinada pela dificuldade. Se a letra digitada existir na `resposta` é realizado um `for` na linha 28 (Quadro 9), varrendo toda a `resposta` para encontrar os índices onde o caractere digitado é igual a letra. Esta estrutura de repetição foi criada em razão de existirem casos que o caractere é encontrado em

mais de uma posição. Nesta situação com a posição da letra encontrada no vetor, é adicionando no `vetorLetras` a letra correspondente e na linha 31 (Quadro 9) é verificado se ainda existem letras a serem preenchidas, se não houver o jogo termina.

Quadro 9 – Jogo letras

```
01 ...
02 iniciaVetorVazio() {
03   this.jogo.resposta = this.jogo.resposta.replace(" ","-");
04     for (var i = 0; i < this.jogo.resposta.length; i++) {
05       if (this.jogo.resposta[i].indexOf("-")) {
06         this.vetorLetras.push("_");
07       } else {
08         this.vetorLetras.push("-");
09       }
10     }
11 }
12
13 verificaLetra() {
14   if (this.letra != "") {
15     this.letra = this.letra[0].toUpperCase();
16     if (this.jogo.resposta.toUpperCase().indexOf(this.letra) == -1) {
17       this.qtdErro++;
18      if (this.qtdErro == 1) {
19         this.letraDigitadas = this.letra;
20     } else {
21        this.letraDigitadas = this.letraDigitadas + ", " + this.letra;
22      }
23        this.letra = "";
24      if ("" + this.qtdErro == this.jogo.dificuldade) {
25         this.presentAlert("Game Over", "Tente novamente!");
26      }
27   } else {
28      for (var i = 0; i < this.jogo.resposta.length; i++) {
29        if (this.jogo.resposta.toUpperCase()[i] == this.letra) {
30           this.vetorLetras[i] = this.jogo.resposta[i];
31           if (this.vetorLetras.indexOf("_") == -1) {
32             this.presentAlert("Parabéns!!!", "Você ganhou o jogo");
33           }
34        }
35      }
36      this.letra = "";
37     }
38   }
39 }
40 ...
```

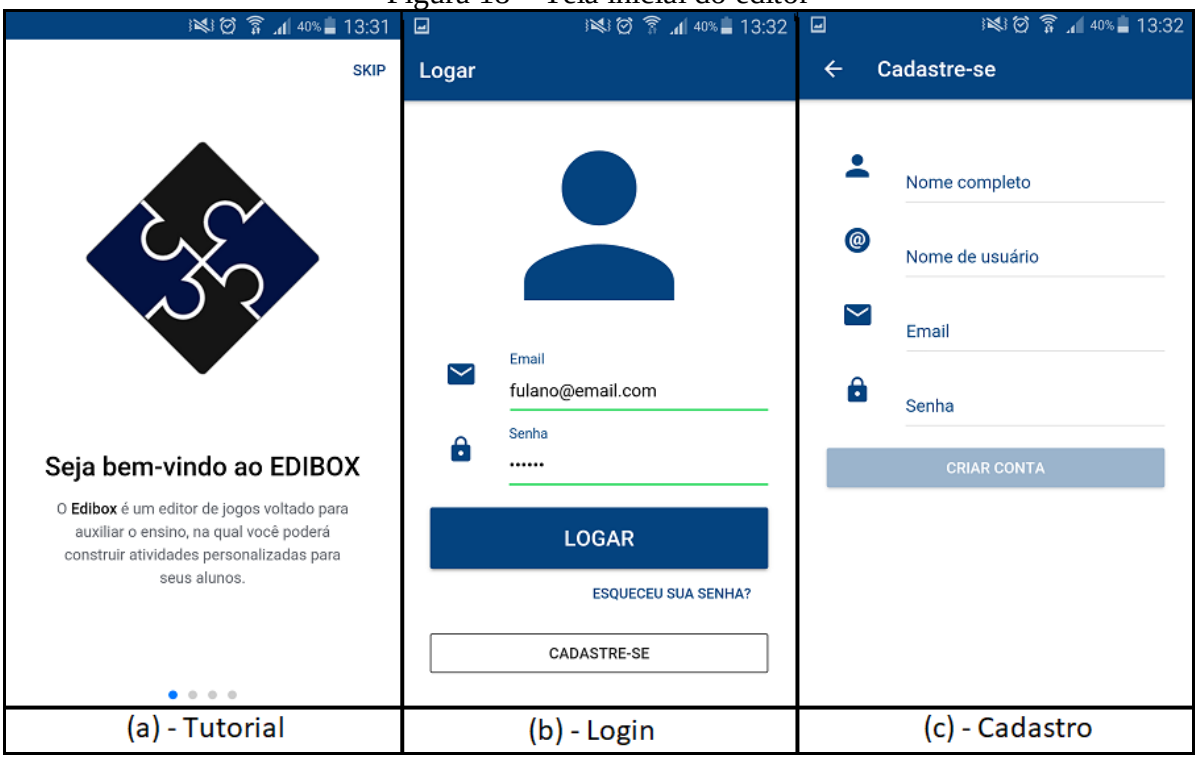
Fonte: elaborado pelo autor.

3.3.2 Operacionalidade da implementação

Esta subseção demonstra a operacionalidade do aplicativo híbrido em nível de usuário. Para isto foi utilizado um celular Samsung Galaxy S4 modelo GT-I9515L com sistema operacional Android 5.0.1 para apresentar as figuras desta subseção. Também para o funcionamento do aplicativo é necessário ter conexão com a internet, pois todos os dados estão armazenados no servidor do Firebase.

Ao iniciar o aplicativo, é apresentado um breve tutorial (Figura 18 (a)), exibindo e explicando algumas funcionalidades do editor, direcionando para a tela de *login* (Figura 18 (b)), se não estiver autenticado no aplicativo. Nesta tela o usuário informa o seu e-mail e senha já cadastrados para ter acesso ao editor e poder buscar e guardar os conteúdos e os jogo no Firebase. Se o usuário `esquecer sua senha`, ele tem a opção de enviá-la novamente via e-mail cadastrado, para que receba um e-mail de redefinição de senha e assim voltar a utilizar a aplicação. Caso não tenha uma conta no editor o usuário poderá `cadastrar-se` criando assim uma nova conta. Nesta tela todos os campos são obrigatórios (Figura 18 (c)).

Figura 18 – Tela inicial do editor



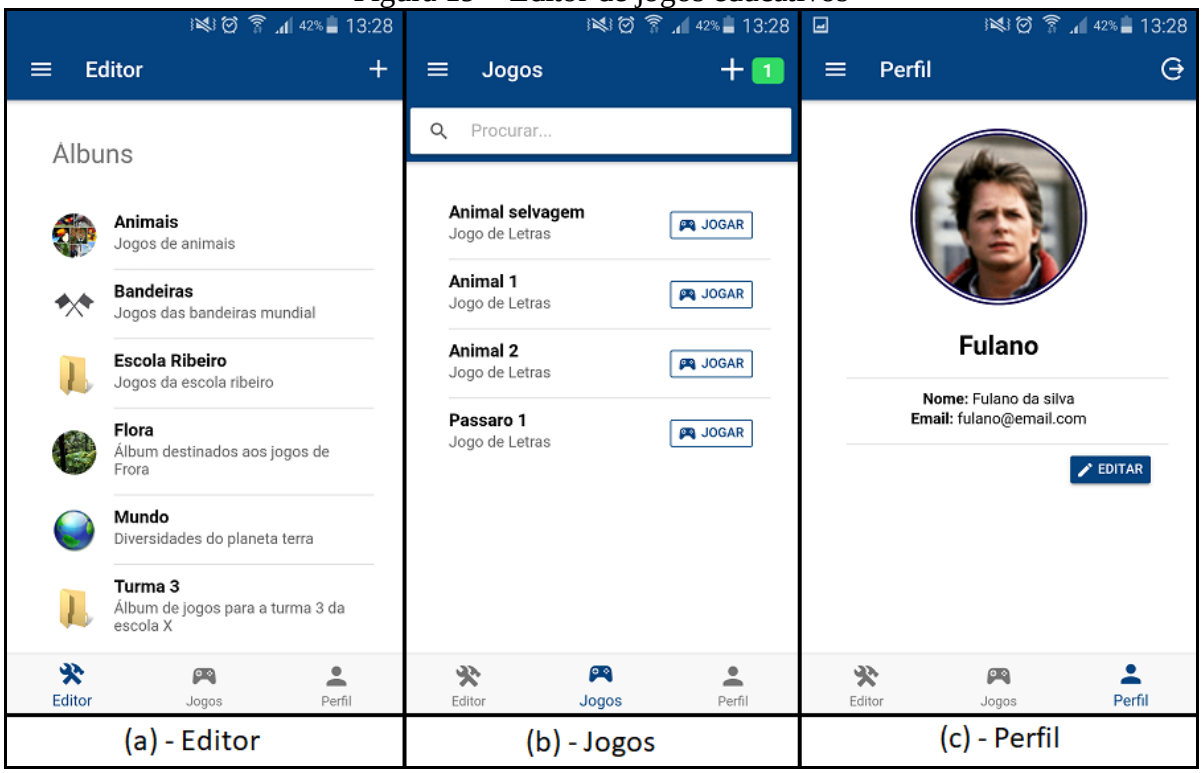
Fonte: elaborado pelo autor.

Entrando no editor existem duas formas de navegabilidade entre as páginas, pelo menu lateral, que contém todas as páginas do editor ou pela barra inferior, que dá acesso rápido às principais telas do mesmo. Com o usuário autenticado ele é direcionado para a tela de game Figura 19 (b) na qual são listados todos os jogos que o usuário já construiu para poder jogá-los. Nesta página, na barra superior, existe a opção de importar ou excluir jogos que foram compartilhados por outros usuários.

Na Figura 19 (a) é apresentada a página do editor, na qual são apresentados os álbuns que são agrupadores de jogos. Nesta tela é possível criar mais álbuns clicando no ícone "+". Arrastando o álbum para o lado direito são exibidos os botões de `edição` e `exclusão` do álbum. Na Figura 19 (c) é apresentada a tela de perfil do usuário com suas informações

cadastradas e sua foto de perfil que pode ser alterada, bem como, na parte superior há um botão para sair do editor.

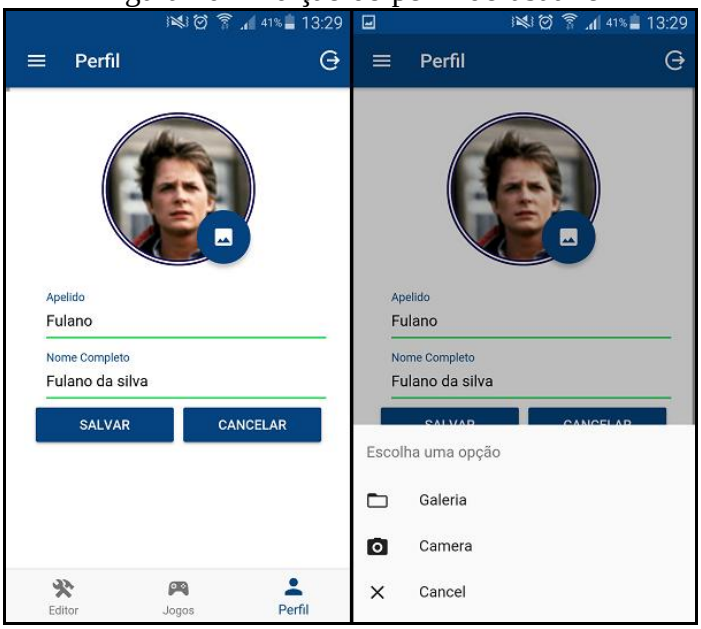
Figura 19 – Editor de jogos educativos



Fonte: elaborado pelo autor.

Na Figura 20 é exibida a tela de perfil, na qual o usuário pode editar suas informações pressionando o botão `editar` na qual habilitará todos os campos editáveis. No botão em cima da imagem, quando apertado abrirá um menu abaixo para escolha da abertura de câmera ou da galeria para importação da foto. Em dispositivos sem câmera, abrirá o *explorer* do sistema. Após realizar as alterações desejadas é só pressionar em salvar para que o sistema envie todas as informações para o banco.

Figura 20 – Edição do perfil do usuário



Fonte: elaborado pelo autor.

Já no álbum, na parte superior são mostradas as informações como `nome`, `descrição`, `imagem` e um botão `novo jogo` para criação de novos jogos. Logo abaixo são listados os jogos já criados como ilustrado na Figura 21. Para cada jogo são apresentados cinco ações, possíveis que o usuário pode realizar: teste, onde o usuário poderá jogar o jogo em outra tela; compartilhamento com usuários, a qual abre uma tela listando todos os usuários cadastrados na aplicação e o usuário escolhe com qual pessoa ele deseja compartilhar o seu jogo; mover para outro álbum, a qual abre uma pequena tela para a escolha do álbum que o usuário deseja adicionar o jogo; editar que abre a mesma tela de cadastro, porém com os campos preenchidos e assim podem ser alteradas as informações; e a última opção é a de exclusão do jogo.

Figura 21 – Álbum de jogos

Fonte: elaborado pelo autor.

Ao clicar em um novo jogo o usuário é direcionado para uma tela onde escolherá um dos *templates* de jogos disponíveis para utilizar como ilustrado na Figura 22. Existe apenas um *template* disponível para criação, o de letras. Porém, cada *template* define um tipo de jogo e cada jogo tem seus recursos e campos.

Figura 22 – Escolha de templates

Fonte: elaborado pelo autor.

O jogo do tipo `letras` tem como objetivo adivinhar qual é a palavra que está oculta. Para isto, o usuário formula uma questão que esteja relacionada com o assunto estudado e compartilha para que outras pessoas tentem descobrir. Diante disto, para a construção deste jogo, como ilustrado na Figura 23, o usuário preencherá um formulário informando um `nome`

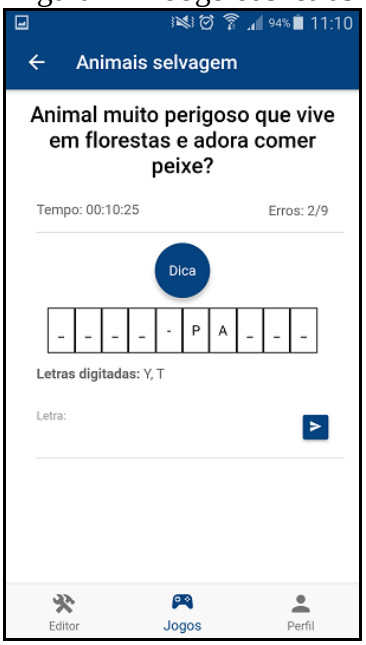
para o jogo. No campo `pergunta` ele fará a pergunta que o usuário tentará descobrir a resposta. No campo `dificuldade` há a possibilidade de definir a dificuldade para resolução do jogo, na qual é definida a quantidade de erros máximos que o jogador pode cometer. Na dificuldade `fácil` pode cometer até 9 erros, no `médio` 6 erros e no `difícil` somente 3 erros. Logo abaixo também pode ser definido um tempo máximo para a resolução do jogo. Para isso, o valor deve ser preenchido no campo `Tempo`, no formato de horas, minutos e segundos, ambos com dois dígitos. O último dado seria a `dica`, que é em forma de imagem, onde o usuário carrega uma imagem ou tira uma foto, ajudando assim a resolver a jogo criado.

Figura 23 – Cadastro do jogo de letras

Fonte: elaborado pelo autor.

A Figura 24 apresenta o jogo. Nela é apresentado o nome do jogo, o tempo de resolução, a dificuldade representada pelos erros, e a pergunta. No campo `dica`, quando clicado, aparecerão duas opções, imagem ou áudio (o áudio ainda não está implementado). Logo abaixo é apresentado as lacunas das letras que formam a resposta da questão, e a seguir são apresentadas as letras erradas e a entrada para digitar as letras. O campo só reconhece a primeira letra, ignorando o resto se for colocado.

Figura 24 – Jogo das letras



Fonte: elaborado pelo autor.

3.4 ANÁLISE DOS RESULTADOS

Nesta seção são analisados os resultados obtidos com o desenvolvimento desta aplicação. Inicialmente são descritos os principais problemas enfrentados pelo autor no desenvolvimento do editor. Em seguida, tem-se os testes de interface e usabilidade, além do experimento com convidados. Por fim, é apresentada a comparação entre os trabalhos correlatos e o desenvolvido.

3.4.1 Descrição geral dos problemas enfrentados

Inicialmente discutiu-se a ideia de estender o trabalho do Corso (2017), acrescentando a realidade aumentada ao editor, porém o mesmo apresentou alguns empecilhos que seriam custosos para resolver no tempo disponível e com a pouca experiência do autor com as tecnologias utilizadas. Percebe-se que a utilização em dispositivos móveis, na sua grande maioria com telas pequenas, fazia com que o editor se comportasse de maneira inadequada. Outro ponto observado foi a utilização de dois servidores para armazenamento, um para a atividade e outro para o compartilhamento. Diante destes fatos e outros menos pertinentes optou-se por recriar o editor de jogos, tomando o cuidado com as tecnologias e plataformas empregadas, visto que para estender o trabalho, tem que se alterar grande parte dos serviços, adaptar todas as páginas e ainda arrumar alguns problemas que viriam a acontecem em dispositivos móveis devido ao acesso aos recursos nativos.

O trabalho atingiu o seu objetivo, porém aconteceram divergências da proposta inicial visto que trabalhar com várias tecnologias diferentes sem um conhecimento prévio causou muitos transtornos e inviabilizou fazer todos os objetivos propostos inicialmente em um curto período de tempo. Sendo assim, optou-se em trabalhar com o *framework* Ionic em conjunto com o Firebase.

O Ionic permitiu a criação do aplicativo para várias plataformas com a mesma base de código em um tempo onde se tem poucas ferramentas como esta. Existem alguns recursos que não funcionaram como se esperava, sendo elas, o suporte a movimentação de elementos na tela (*drag and drop*), demasiada abstração dos componentes e diversas atualizações dos componentes e alguns *plugins* do Ionic Native que não desempenham um papel muito bom em navegadores web.

Uma das dificuldades encontradas na construção dos jogos de encaixe, que inicialmente se pensava em disponibilizar, foi a falta de suporte a *drag and drop*, na qual não se conseguia arrastar os componentes pela tela e soltar em determinadas regiões. Desta forma, seriam custosas as adaptações e a implementação de todos os métodos visto que teria que se pensar em todas as plataformas envolvidas. Diante disto, procurou-se soluções de terceiros que atendessem ao cenário, mas não se obteve êxito ou a ferramenta não atendia a todas as especificações.

No *framework* é possível realizar muito do que é feito na web, desde a criação de novos componentes até a estilização dos atuais, no entanto o Ionic abstrai bastante as coisas do desenvolvedor e é constantemente atualizado para trazer novos recursos. Essas mudanças acabam acentuando a curva de aprendizado, pois muitas vezes parâmetros deixam de existir e funcionalidades trabalham diferente do que eram antes. Na criação de novos recursos ou componentes teve que se tomar cuidados para que a aplicação ficasse estável em todas as plataformas e tamanho de telas, custando um certo tempo para o desenvolvimento.

Ao utilizar Ionic Native que é um facilitador para a utilização dos *plugin* do Apache Cordova, percebeu-se que alguns destes *plugins* não se comportavam da maneira adequada nos navegadores web e dispositivos móveis mesmo apresentando suporte. No desenvolvimento do jogo de letras, para agilizar a entrada de dados, pretendia-se abrir o teclado nativo do dispositivo para entrada dos caracteres, no entanto o *plugin* do teclado após chamado o método de abertura, não abriu, somente com a existência de um campo `<input>` do HTML onde o usuário clicava para escrever. Além disto, o *plugin* da câmera funcionou incorretamente nos navegadores web, abrindo a câmera sobre os componentes da tela causando transtornos para o utilizador, visto que muitos computadores portáteis têm câmeras

frontais. Entretanto estes empecilhos podem ter acontecidos por conta da falta de experiência do autor com a tecnologia.

Outras dificuldades foram encontradas nos serviços de *back-end* do Firebase, no qual algumas funcionalidades do editor não ficaram totalmente prontas devido ao tempo disponível para o desenvolvimento. No serviço de armazenamento, após muitas pesquisas, não encontrou-se a opção de mover o arquivo sem ter que fazer o download e upload novamente ao servidor. Também não foi encontrada a opção de apagar vários arquivos ao mesmo tempo. Já no compartilhamento dos jogos com outros usuários, a URL da imagem no *storage* não é alterada, na qual se o usuário que compartilhou mudar a imagem enviada no jogo, o usuário que recebeu o jogo perderá a referência desta imagem. No entanto, para resolver isto seria necessário baixar a imagem e quando compartilhar enviá-la novamente para o armazenamento, gerando assim uma nova URL para o usuário que recebeu. Outra função não realizada foi a de apagar múltiplos arquivos dos jogos já que quando o álbum é excluído, todos os jogos que ele continha também eram removidos. Como sugestão para resolução desta limitação, deve-se criar uma estrutura de repetição passando todos os caminhos absolutos dos arquivos para a exclusão e quando não existir mais arquivos este álbum é automaticamente excluído do armazenamento do Firebase.

Sobre a autenticação observou-se que acontece uma instabilidade no sistema quando o usuário está autenticado em dois dispositivos distintos ao mesmo tempo e em um deste resolve sair da aplicação, causando um erro no dispositivo que está autenticado, devido o mesmo estar acessando os dados do banco de dados que contém regras de acesso. Este problema também ocorre quando ficam armazenadas informações em cache e o usuário desloga, gerando instabilidade.

Alguns componentes nativos do Ionic e as mensagens enviadas pelo servidor são apresentadas em inglês, por não serem tão relevantes aos objetivos do trabalho, já que no futuro pode ser realizada a internacionalização do editor. Por ser uma aplicação híbrida e os custos para o desenvolvimento e junto com a inexperiência do autor, não se consegue ter gráficos de jogos tão estimulantes quanto os jogos criados nativamente. Alguns recursos do editor não foram terminados a tempo de serem entregues, devido a empecilhos inesperados que aconteceram como: a criação do cronômetro recursivo para os jogos; o envio por e-mail do QRCode com o jogo; a barra de busca dos jogos; integrar o *plugin* de mídia para gravação dos áudios; a disponibilização de mais *templates* de jogos para os professores e a diferenciação dos papeis de aluno e professor.

3.4.2 Teste de interface

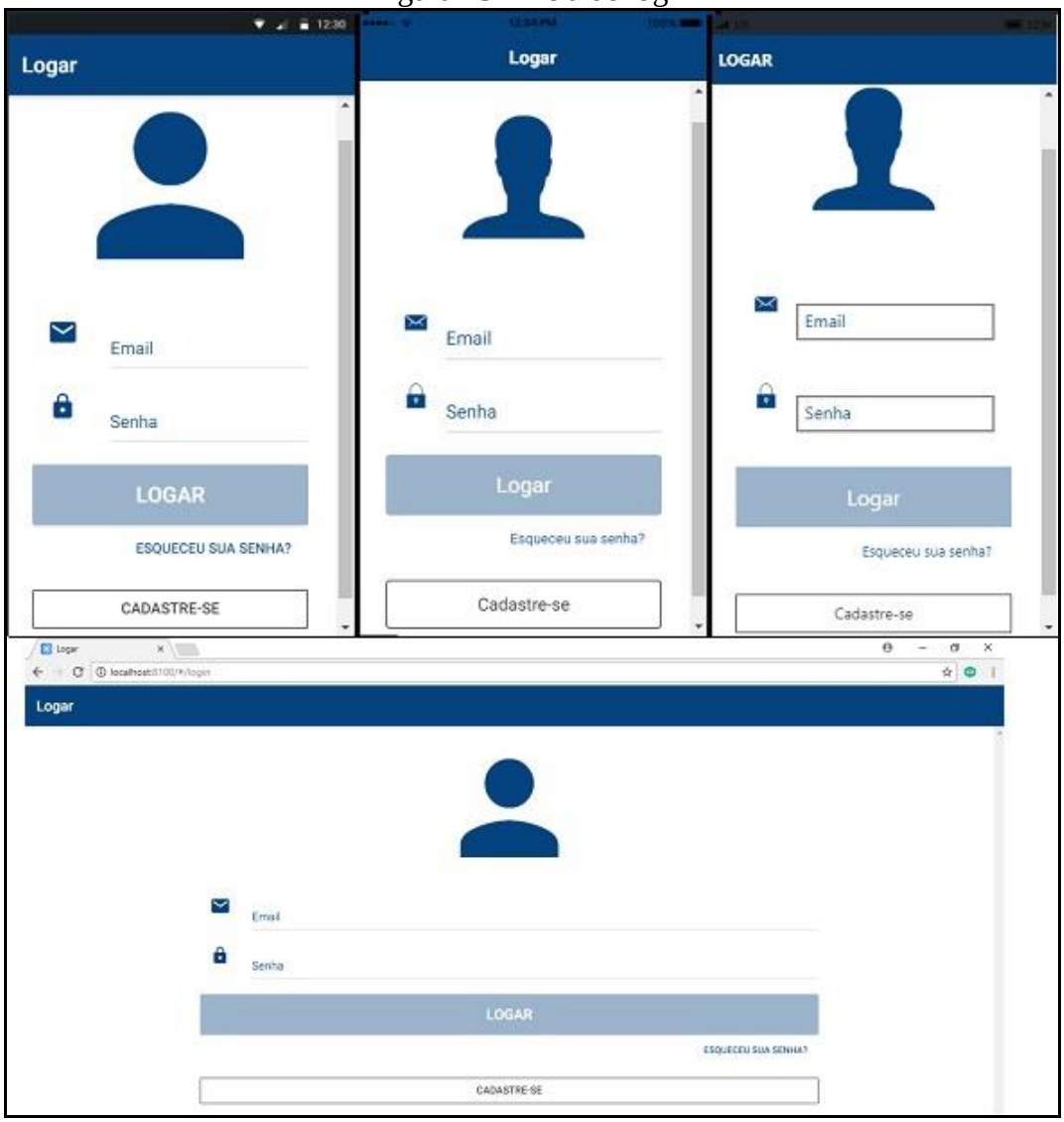
Nesta subseção são comparadas as interfaces gráficas da aplicação entre as plataformas móveis e web e seus diferentes designs. Quanto a interação do usuário com a aplicação, o que se diferencia é a utilização dos *plugins* do Apache Cordova para funcionalidades nativas nos dispositivos móveis.

O desenvolvimento do editor multiplataforma teve como objetivo manter a interface e as experiências semelhantes independentemente da plataforma em que o aplicativo está sendo executado. Portanto, o *framework* Ionic 3 define o estilo dos seus componentes de aplicativos com base na plataforma. Os usuários de iOS utilizam o design nativo do iOS, para os navegadores web e os de Android utilizam o *framework* materialize e para celulares Windows o design próprio do Windows Phone. Contudo, pode ser forçado a utilização de um único estilo para todas as plataformas perdendo as características naturais de cada plataforma.

Todas as imagens abaixo foram tiradas do simulador Ionic e seguirão a mesma ordem de apresentação sendo acima Android (esquerda), iOS (centro), Windows Phone (direita) e abaixo o navegador web Google Chrome.

A tela de *login* é apresentada na Figura 25. Nesta tela destacam-se as diferenças no formulário principalmente do Windows Phone, pois ele apresenta todos os campos com uma borda retangular enquanto que nas outras plataformas apresenta apenas uma linha horizontal abaixo. Na barra superior o título no Android é alinhado à esquerda assim como no navegador, no iOS é centralizado e no Windows Phone é alinhado à esquerda e em letra maiúscula. No Android e no navegador, todos os botões estão em maiúsculo já nas outras plataformas não. No navegador existe uma margem lateral devido a tela ter uma largura maior do que os dispositivos móveis.

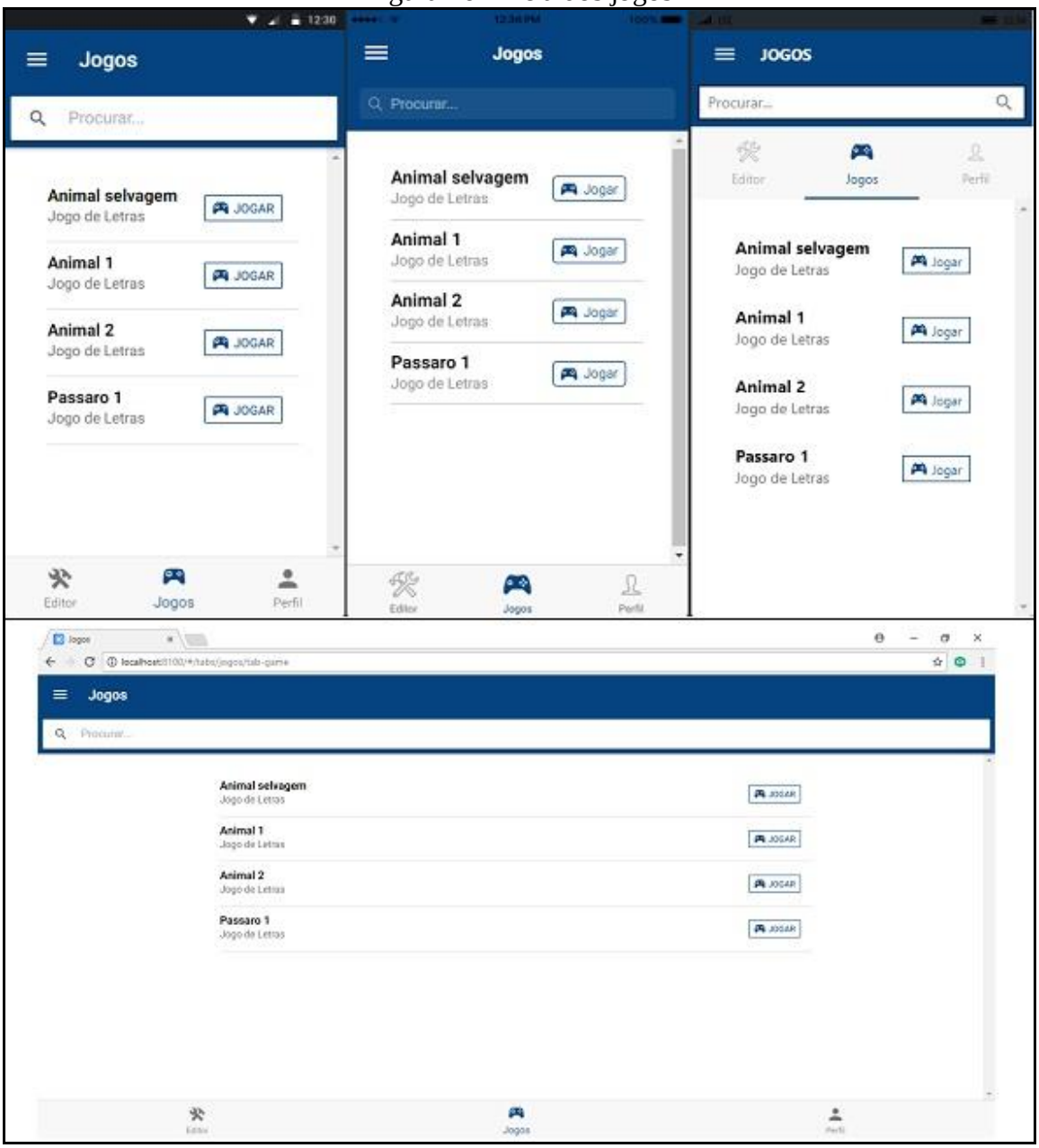
Figura 25 – Tela de login



Fonte: elaborado pelo autor.

Na Figura 26 é ilustrado a tela de `jogos`, na qual observa-se que a barra de navegação principal no Windows Phone é posicionada acima enquanto que nas outras plataformas encontra-se na parte inferior. A barra de pesquisa no iOS tem transparência e os botões têm os cantos um pouco arredondados, já no Windows Phone e Android a barra é apresentada na cor branca e com os ícones em lados opostos com os botões são retangulares. Quanto a lista de jogos o Windows Phone não apresenta as linhas de divisão dos itens que são apresentadas em outras plataformas. Os ícones são diferentes de uma plataforma para a outra, porém são semelhantes.

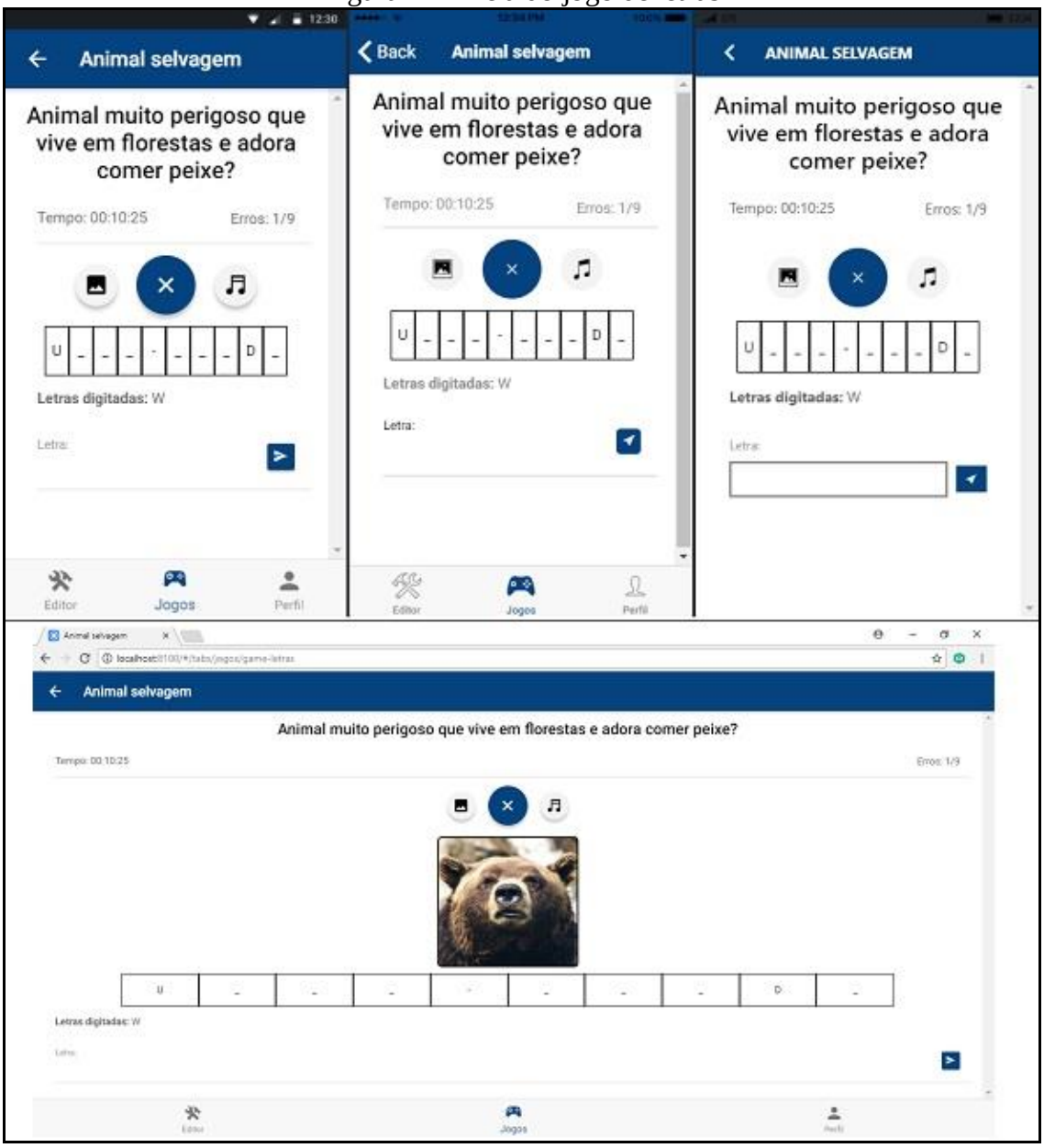
Figura 26 – Tela dos jogos



Fonte: elaborado pelo autor.

O tamanho da tela maior dos navegadores em comparação com o tamanho das telas dos dispositivos móveis resulta em um maior espaço para adicionar mais informações e recursos como visto na Figura 27 o jogo de letras. No Windows Phone não é apresentado a barra de navegação principal já nas outras plataformas são. Outro destaque fica para o botão de voltar que no iOS é apresentado com a palavra `Back`, além dos ícones serem diferentes entre as três plataformas. O botão de dica no Android tem sombreamento e em alguns textos no iOS são apresentados em cor cinza.

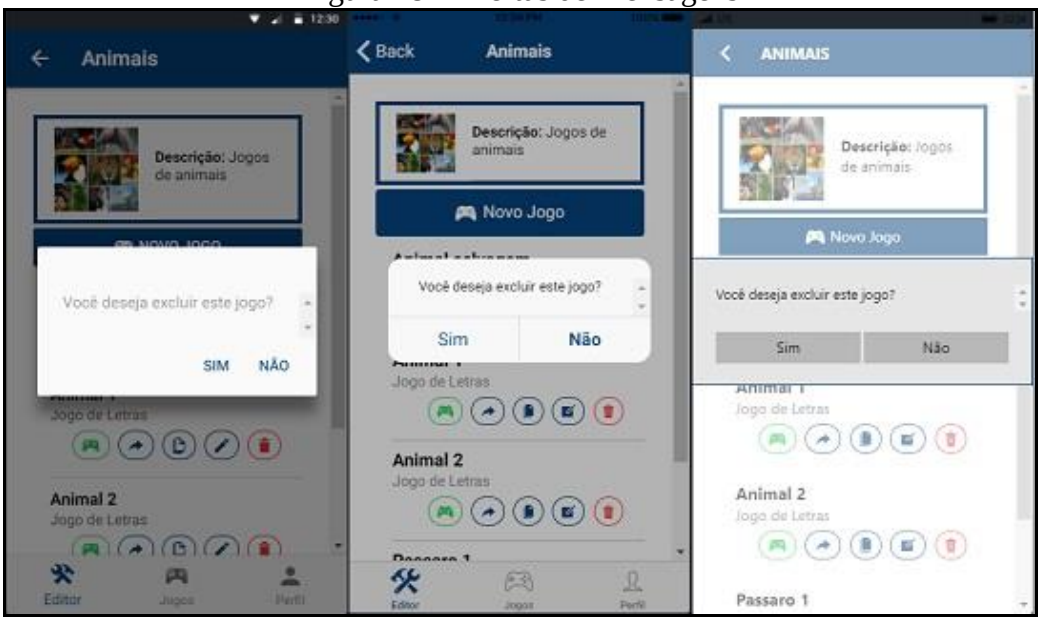
Figura 27 – Tela do jogo de letras



Fonte: elaborado pelo autor.

As mensagens de alerta no navegador são apresentadas centralizadas na tela e é igual ao Android que é retangular com os botões alinhados à direita. No iOS tem-se um visual com os cantos arredondados e os botões justificados e no Windows Phone o alerta é na cor cinza e ocupa toda a largura da tela. Na Figura 28 podem ser observados os diferentes tipos de alertas encontrados em cada plataforma.

Figura 28 – Alertas de Mensagens



Fonte: elaborado pelo autor.

3.4.3 Teste de usabilidade

O experimento aconteceu em duas etapas no dia 29 do mês de novembro de 2017. Na primeira etapa os testes foram realizados com seis professores da Associação de Pais e Amigos do Excepcional (APAE) de Apiúna, onde o autor desta monografia trabalha, de forma individual e acompanhados pelo autor. O seu orientador realizou o experimento com outros três estagiários da FURB. A segunda etapa foi realizada através da disponibilização do editor on-line, na qual qualquer pessoa poderia utilizar e responder o formulário. Como o experimento foi realizado em apenas um dia, o experimento foi realizado com 11 especialistas e teve como objetivo avaliar a usabilidade e a aplicabilidade do editor como complemento aos métodos tradicionais de ensino. O registro fotográfico do experimento pode ser observado no Apêndice B da Figura 29 e Figura 30.

Para o experimento foi aplicado o questionário disponível no Apêndice C, que está dividido em três partes. A primeira parte continha questões relacionadas ao perfil do usuário. A segunda parte guiava o usuário pelo editor para o conhecimento das rotinas de criação e jogabilidade que podem ser realizadas no mesmo. Na última parte do questionário, era solicitado que o usuário opinasse sobre a percepção obtida ao utilizar o aplicativo. As questões tratavam da usabilidade, aplicabilidade no dia a dia e sobre as dificuldades encontradas na utilização do editor.

No início do experimento com os especialistas deu-se uma introdução do problema que motivou a realização deste trabalho, demonstrando os objetivos que se esperava alcançar.

Logo em seguida aconteceu uma breve apresentação do editor e do questionário, partindo deste ponto para as atividades que o questionário orientava.

Após realização a introdução, na primeira parte do formulário buscam determinar o perfil da amostra, estas questões podem ser visualizadas no Apêndice C na Figura 31 e Figura 32. Os resultados obtidos são apresentados na Tabela 1. Com base nos dados é possível concluir que a amostra ficou dividido entre feminino e masculino, com a maioria de pessoas com mais de 25 anos. Um dado relevante da pesquisa aponta que 36,4% das pessoas tem pós-graduação e 18,2% concluíram o ensino superior e o restante está concluindo. Outro ponto levantado foi que todos utilizam recursos tecnológicos e a grande maioria utiliza destes recursos diferenciados para o aprendizado, porém não foi perguntado se eles preferem este tipo de metodologia. No final foi perguntado quais dispositivos eram mais utilizados no dia a dia e observa-se que os usuários utilizam com maior frequência dispositivos móveis.

Tabela 1 – Perfil dos usuários

Sexo	45,5% masculino 54,5% feminino
Idade	0% entre 11 a 15 anos 18,2% entre 16 a 20 anos 9,1% entre 21 a 25 anos 27,2% entre 26 a 30 anos 45,5% mais de 30 anos
Nível de escolaridade	0% ensino fundamental completo – 1º grau 0% ensino médio incompleto 45,4% ensino superior incompleto 18,2% ensino superior completo 36,4% pós-graduado
Você possui algum dispositivo tecnológico, como celular, tablet ou computador.	100% sim 0% não
Você utiliza algum método, técnica ou abordagem diferenciada de ensino, com recursos tecnológicos.	9,1% nunca utilizaram 63,6% as vezes 27,3% sempre utilizam
Indicativo de qual dispositivo tecnológico, é mais utilizado diariamente.	27,2% computador de mesa 45,5% celular/Smartphone 27,3% notebook 0% tablet 0% outros 0% não se aplica

Fonte: elaborado pelo autor.

A segunda parte do questionário continha instruções para o usuário realizar algumas atividades. As questões podem ser consultadas no Apêndice C nas Figura 33 a 39. Como exibido na Tabela 2, todos os especialistas realizaram as atividades que eram propostas pelo questionário. As atividades eram de autenticação, criação de jogos, edição, exclusão e jogatina e foram realizadas com sucesso, no entanto duas pessoas relataram um erro no

compartilhamento de jogos, pois os mesmos não tinham criado um álbum ainda e receberam um jogo de outra pessoa, diante desta situação eles não conseguiu aceitar o jogo.

Tabela 2 – Respostas do questionário para as instruções ao usuário

Criar uma conta e se autenticar no aplicativo	100% sim 0% não
Criar um álbum para adicionar os jogos	100% sim 0% não
Criar um jogo do tipo letras, preenchendo todos os campos obrigatórios.	100% sim 0% não
Compartilhar um jogo com outro usuário.	81,8% sim 18,2% não
Editar e excluir um jogo	100% sim 0% não
Jogar o jogo desenvolvida	100% sim 0% não

Fonte: elaborado pelo autor.

A última etapa do questionário continha questões de usabilidade do editor, onde o usuário pode deixar sua opinião sobre a utilização. Essas respostas são mostradas na Tabela 3. Como descrito anteriormente, todos com exceção de duas pessoas, conseguiram realizar todas as atividades. Nos resultados observados 90,1% acharam o editor fácil de usar e intuitivo. Dos usuários especialistas entrevistados, 100% deles assumiram que a utilização do editor como método para desenvolvimento de exercícios pode auxiliar na compreensão e na fixação dos assuntos abordados em sala de aula. Estes fatos asseguram que objetivo inicial do trabalho foi alcançado, de propor uma ferramenta que pudesse ser realizada para construção de jogos educativos e levasse a tecnologia para dentro da sala de aula. A terceira parte do questionário sobre as questões de usabilidade podem ser consultadas no Apêndice C na Figura 41.

Tabela 3 – Respostas do questionário para as questões de usabilidade

Quantas das atividades solicitadas foram executadas sem auxílio.	81,8% todas 18,2% a maior parte 0% metade das tarefas 0% menos da metade das tarefas 0% nenhuma tarefa
Acharam o EdiBox intuitivo e fácil de usar.	90,9% sim 9,1% não
Acharam que a utilização do editor para apresentar conteúdos e desenvolver exercícios, pode auxiliar na compreensão e na fixação do assunto abordado em sala de aula.	100% sim 0% não
Avaliaram o editor para jogos multiplataforma.	81,8% muito bom 18,2% bom 0% regular 0% insatisfatório

Fonte: elaborado pelo autor.

O questionário de usabilidade termina com uma questão discursiva sobre as dificuldades encontradas no editor. As respostas podem ser vistas no Apêndice C na Figura

42. Nesta figura, percebe-se que algumas pessoas relataram que o editor precisaria ter mais dicas durante sua utilização, pois as atividades muitas vezes são realizadas por pessoas mais velhas que não estão habituadas a utilizarem tecnologias diariamente. Muitos dos problemas relatados aconteceram devido à falta de conhecimento na utilização do editor, visto que eles tiveram apenas de 5 a 10 minutos para testar o aplicativo antes de realizar o questionário.

3.4.4 Comparação com os trabalhos correlatos

Nesta seção é apresentado o Quadro 10 no qual é realizada a comparação das principais características dos trabalhos correlatos com o trabalho apresentado nesta proposta.

Quadro 10 – Comparativo entre os trabalhos correlatos

Características/Trabalhos relacionados	Corso (2017)	Silva (2015)	Playosmo (2017a)	EdiBox
realiza autenticação de usuário	X	X	X	X
ferramenta educacional	X	-	X	X
permite incluir arquivo multimídia	X	X	X	X
gratuito	X	X	-	X
compartilhamento de arquivos	X	-	-	X
utiliza recursos nativos	-	-	X	X
trabalha offline	-	-	X	-
integração com o Firebase	-	X	-	X
multiplataforma	-	X	-	X

Fonte: elaborado pelo autor.

A partir da comparação estabelecida no Quadro 10, é possível notar que todos os trabalhos têm inicialmente um sistema de autenticação de usuário. O trabalho do Corso (2017) é o mais parecido com o trabalho proposto, pois ele também é um editor de jogos que foi desenvolvido para web e tem possibilidade de criação e compartilhamento de jogos via QRcode, enquanto que EdiBox é um aplicativo multiplataforma desenvolvido em Ionic assim como o trabalho de Silva (2015) e pode-se construir jogos e compartilhá-los.

O Playosmo (2017a) é o único que é pago e desenvolvido apenas para iOS, também utiliza os recursos de câmera do dispositivo, assim como o trabalho proposto, porém comparando com os três ele é o único que não depende da conexão com a internet para funcionar. O trabalho de Silva (2015) é o único que não é uma plataforma de jogos de cunho educativo, mas tem integração com o Firebase para o armazenamento e recuperação de arquivos em tempo real semelhante ao trabalho proposto.

4 CONCLUSÕES

O objetivo deste trabalho foi desenvolver um aplicativo educacional que pudesse contribuir com os educadores para um processo de ensino e aprendizagem mais eficiente. Com o resultado criou-se um editor de jogos educativos multiplataforma que pode explorar o lado lúdico e levar a tecnologia para dentro da sala de aula. No editor de jogos o professor pode construir atividades com os conteúdos abordados em sala de aula através do *template* customizável que o editor disponibiliza, compartilhando-os para os alunos praticarem. O desenvolvimento e a disponibilização do EdiBox possibilitaram o atingimento do primeiro objetivo específico definido para o trabalho.

Para o desenvolvimento do aplicativo foram utilizados o *framework* Ionic, que possibilitou com a mesma base de código gerar insumos para as principais plataformas do mercado, atingindo desta forma um dos objetivos específicos. O Ionic integra várias tecnologias web, para gerar aplicações híbridas, como as plataformas Angular e Cordova, mostrando desta forma que o *framework* escolhido foi adequado para o desenvolvimento.

Outra ferramenta que foi utilizada e influenciou bastante no desenvolvimento da aplicação foi o Firebase como servidor *back-end*, que permitiu persistir os dados na nuvem e elimina a necessidade de criar um servidor de autenticação, armazenamento, persistência de dados e outros recursos pertinentes a aplicação. Contudo, foi utilizada uma versão da ferramenta que possui suporte inadequado ao *storage*, causando limitações no carregamento e exclusão dos arquivos de imagem e áudio.

Todos os objetivos propostos foram atingidos, mas a curva de aprendizagem foi bem acentuada, mesmo com as tecnologias utilizadas terem certo nível de abstração. Referente a implementação, pode-se destacar os desafios enfrentados que causaram algumas limitações ao editor, tais como, a diferença na utilização dos recursos móveis e web, a interface gráfica de aplicações híbridas não ser tão rica em detalhes quanto a de aplicativos nativos, a funcionalidade de *drag and drop*, a divisão dos papeis de utilização do editor (aluno/professor), a exclusão de arquivos do *storage*, instabilidade na autenticação quando o usuários está logado em vários dispositivos ao mesmo tempo, a disponibilização de mais *templates* para criações de atividades e a possibilidade de adicionar recursos multimídia diferenciados. Algumas destas limitações foram descritas com mais detalhes na seção 3.4.1. além de outras que foram iniciadas, mas não foram concluídas devido ao tempo.

Por fim, presume-se que este trabalho possa contribuir de uma maneira singela para a área educacional, visto que disponibilizará uma ferramenta de desenvolvimento de jogos

didáticos para múltiplas plataformas, auxiliando desta forma o educador a tornar suas aulas mais lúdicas e interativas, na qual utilizará de recursos tecnológicos que os alunos diariamente convivem, tornando o processo de ensino e aprendizagem mais prazeroso.

4.1 EXTENSÕES

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

a) desenvolver os papeis de professor, aluno, coordenador e outros.
b) implementar *service work* para que a aplicação possa trabalhar off-line;
c) permitir que o próprio professor possa criar seus próprios layouts;
d) criar rotinas de análise e acompanhar do desenvolvimento educacional do aluno;
e) disponibilizar mais *templates*, como o jogo de quebra-cabeça, imagens, matemática e de associação;
f) incorporar as autenticações pelas redes sociais como Google, Facebook e Twitter;
g) desenvolver elementos arrastáveis para criação de jogos de encaixe;
h) incorporar mais recursos multimídia às atividades deixando-as mais interativas;
i) construir e implementar atividades utilizando realidade aumentada;
j) disponibilizar opções de atividades padrões, como sistema solar, geografia e história;
k) aplicar gamificação para desenvolver a competitividade entre os educandos;
l) aplicar internacionalização, permitindo que ele seja utilizado em outros idiomas;
m) aplicar acessibilidade para que o editor possa ser usando por pessoas deficientes.

REFERÊNCIAS

ALMEIDA, Aretusa L. **A importância do uso dos jogos digitais no programa mais educação.** 2014. 28 f. Trabalho de Conclusão de Curso (Licenciatura em Pedagogia) - Centro de Educação, Universidade Estadual da Paraíba, Campina Grande.

BATTISTELLI, Juliana. **Google Firebase for dummies:** o que é e como funciona a plataforma. [S.l.], 2017. Disponível em: <https://blog.mastertech.tech/tecnologia/google-firebase-for-dummies-o-que-e-e-como-funciona-plataforma/>. Acesso em: 23 nov. 2017.

BRUSTEIN, Joshua. **This iPad App Tries to Take Student Eyes Away From Screens.** [S.l.], 2014. Disponível em: <https://www.bloomberg.com/news/articles/2014-05-22/this-ipad-app-tries-to-take-student-eyes-away-from-screens>. Acesso em: 22 nov. 2017.

CABRAL, Carlos. **Criando uma aplicação móvel com Ionic 2 e Angular 2 em dez passos.** [S.l.], 2016. Disponível em: <https://tableless.com.br/criando-uma-aplicacao-movel-com-ionic-2-e-angular-2-em-dez-passos/>. Acesso em: 22 nov. 2017.

CAMPOS, Dannyrooh F. **Por que adotar o Angular para desenvolvimento?** [S.l.], 2017. Disponível em: <http://www.eng.com.br/artigo.cfm?id=5143>. Acesso em: 21 nov. 2017.

CANALTECH. **Com o Osmo, a brincadeira vai além das fronteiras do real e do digital.** [S.l.], 2014. Disponível em: <https://canalte.ch/SGTG>. Acesso em: 22 nov. 2017.

CORSO, Felipe L. **EasyEdu:** editor web para jogos multitoque. 2017. 92 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

CRUNCHBASE. **Osmo.** [S.l.], [2017?]. Disponível em: <https://www.crunchbase.com/organization/osmo>. Acesso em: 22 nov. 2017.

FIREBASE. **AdMob.** [S.l.], [2017?]a. Disponível em: <https://firebase.google.com/docs/admob/>. Acesso em: 22 nov. 2017.

FIREBASE. **Comece a usar gratuitamente.** [S.l.], [2017?]b. Disponível em: <https://firebase.google.com/pricing/?hl=pt-br>. Acesso em: 22 nov. 2017.

FIREBASE. **Documentation.** [S.l.], [2017?]c. Disponível em: <https://firebase.google.com/docs/>. Acesso em: 22 nov. 2017.

FIREBASE. **Google Analytics para Firebase.** [S.l.], [2017?]d. Disponível em: <https://firebase.google.com/docs/analytics/>. Acesso em: 22 nov. 2017.

ILEX. **Osmo une mundo virtual com o real para divertir as crianças**. [S.l.], 2014. Disponível em: <https://blogdoiphone.com/2014/05/osmo-une-mundo-virtual-com-o-real-para-divertir-as-criancas/>. Acesso em: 22 nov. 2017.

IONIC. **About us.** [S.l.], [2017?]. Disponível em: <https://ionicframework.com/about>. Acesso em: 20 nov. 2017.

KLAUS, Allan. **O que é Single Page Application?** [S.l.], 2016. Disponível em: <http://blog.locaweb.com.br/artigos/desenvolvimento-artigos/o-que-e-single-page-application/>. Acesso em: 20 nov. 2017.

LANDIM, Wikerson. **A evolução dos games e da indústria de jogos digitais.** [S.l.], 2015. Disponível em: < https://m.tecmundo.com.br/opiniao/87965-evolucao-games-industria-jogos-digitais-opiniao.htm>. Acesso em: 20 nov. 2017.

LEITE, Maria I; UGGIONI, Juliana. **O papel dos jogos digitais no aprendizado.** [S.l.], 2013. Disponível em: <http://dc.clicrbs.com.br/sc/noticias/noticia/2013/10/o-papel-dos-jogos-digitais-no-aprendizado-4288192.html>. Acesso em: 23 nov. 2017.

LOPES, Sérgio. **Aplicações mobile híbridas com Cordova e PhoneGap.** 1. ed. São Paulo: Casa do Código, 2016.

LYNCH, Max. **Sunsetting Ionic Cloud Push and Auth.** [S.l.], 2017. Disponível em: <http://blog.ionicframework.com/sunsetting-ionic-cloud-push-and-auth>. Acesso em: 20 nov. 2017.

MATOS, Beatriz R. D; SILVA, João G. de B. e. **Estudo comparativo entre o desenvolvimento de aplicativos móveis utilizando plataformas nativas e multiplataforma.** 2016. 61 f. Trabalho de Conclusão de Curso (Engenharia de Software) - Faculdade UnB Gama, Universidade de Brasília, Brasília.

PAULA, Bruno H. de. **Jogos digitais como artefatos pedagógicos:** o desenvolvimento de jogos digitais como estratégia educacional. 2015. 243 f. Dissertação (Mestrado em Artes Visuais) - Curso de Pós-graduação em Artes Visuais, Universidade Estadual de Campinas, Campinas.

PAULA, Bruno H. de; VALENTE, José A. Jogos digitais e educação: uma possibilidade de mudança da abordagem pedagógica no ensino formal. **Revista Ibero-americana de Educação**, Buenos Aires, v. 70, n. 1, p.09-28, jan. 2016.

PEREZ, Sarah. **Former Googlers Launch Osmo, A Gaming Device That Combines Real-World Play With The iPad.** [S.l.], 2014. Disponível em: <https://techcrunch.com/2014/05/22/former-googlers-launch-osmo-a-gaming-device-that-combines-real-world-play-with-the-ipad/>. Acesso em: 10 nov. 2017.

PIRES, Eduardo. **AngularJS, Angular 2, 4 e etc – Passando a confusão a limpo.** [S.l.], 2017. Disponível em: <http://www.eduardopires.net.br/2017/07/angularjs-angular-2-e-4-passando-confusao-limpo/>. Acesso em: 23 nov. 2017.

PLAYOSMO. **Meet Osmo.** [S.l.], [2017?]a. Disponível em: <https://www.playosmo.com/en/>. Acesso em: 20 nov. 2017.

PLAYOSMO. **Tangram.** [S.l.], [2017?]b. Disponível em: <https://www.playosmo.com/en/tangram/>. Acesso em: 20 nov. 2017.

ROETTGERS, Janko. **Tangible Play, Maker of Osmo Augmented Reality Toys, Raises $24 Million From Sesame Workshop, Others.** [S.l.], 2016. Disponível em: <http://variety.com/2016/digital/news/osmo-25-million-funding-1201936425/>. Acesso em: 22 nov. 2017.

SANTOS, Maria A. I. dos. **Utilização de Realidade Aumentada no Desenvolvimento de Software Educacional:** um exemplo em alguns conceitos na Astronomia. 2016. 108 f. Dissertação (Mestrado em Computação Aplicada) - Universidade Estadual de Feira de Santana, Feira de Santana.

SILVA, Diogo H. **Aplicativo Multiplataforma de Escrita Colaborativa em Tempo Real.** 2015. 84 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) - Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares Da Rocha, Marília.

VIANA, Daniel. **Firebase:** descubra no que esta plataforma pode te ajudar. [S.l.], 2017. Disponível em: <https://www.treinaweb.com.br/blog/firebase-descubra-no-que-esta-plataforma-pode-te-ajudar/>. Acesso em: 22 nov. 2017.

APÊNDICE A – Detalhamento dos casos de uso do editor

Neste apêndice são demonstrados nos Quadros 10 a 17, o detalhamento dos casos de uso utilizados pela aplicação desenvolvida.

Quadro 11 – Caso de Uso UC01

UC01 – Efetuar *login*	
Descrição	Permite o acesso a aplicação
Pré-condição	Ter conexão com a internet
Atores	Usuário
Cenário Principal	1. O sistema apresenta a tela de *login*. 2. O usuário informa e-mail e senha. 3. O usuário clica na ação de *login*. 4. O sistema consulta o usuário filtrando pelo e-mail e senha. 5. O sistema redireciona o usuário para a tela de jogos.
Fluxo de Alternativo 1	Após o passo 1 do fluxo principal, caso o usuário não tenha conta e pressiona o botão de cadastrar-se: 1. O sistema abre a tela de cadastro. 2. O usuário preenche todos os campos obrigatórios do formulário. 3. O usuário clica no botão de salvar. 4. O sistema persiste os dados no banco de dados. 5. O sistema redireciona o usuário para a tela de jogos.
Fluxo de Alternativo 2	Após o passo 4 do fluxo principal, caso o usuário tenha esquecido a senha: 1. O usuário clica no botão esqueceu sua senha. 2. O sistema abre uma tela para informar o e-mail. 3. O usuário informa o seu e-mail cadastrado. 4. O sistema envia uma notificação de redefinição de senha para o e-mail. 5. O sistema apresenta a mensagem de “enviado com sucesso” e volta para o cenário principal.
Fluxo de exceção 1	No passo 4 do fluxo principal, caso a conta de e-mail não exista ou a senha seja inválida: 1. O sistema apresentará a mensagem de erro para o usuário.
Pós Condição	O usuário tem acesso ao sistema de acordo com seu perfil.

Fonte: elaborado pelo autor.

Quadro 12 – Caso de Uso UC02

UC02 – Criar jogos	
Descrição	Permite a construção de novos jogos
Pré-condição	Usuário Autenticado. Ter conexão com a internet.
Atores	Usuário
Cenário Principal	1. O usuário entra no álbum. 2. O sistema apresenta as informações do álbum. 3. O sistema apresenta um botão para criação de novos jogos. 4. O usuário clica na ação para criação. 5. O sistema abre a tela de escolha de *templates*. 6. O usuário escolhe o *template* desejado. 7. O sistema redireciona para tela de cadastro do jogo referente ao *template* escolhido. 8. O usuário preenche o formulário com as informações da atividade. 9. O usuário clica no botão de `salvar`. 10. O sistema persiste os dados no banco. 11. O sistema retorna para o passo 2.
Pós Condição	Novos jogos.

Fonte: elaborado pelo autor.

Quadro 13 – Caso de Uso UC03

UC03 – Manter jogos	
Descrição	Permite a consulta, edição, remoção e movimentação do jogo
Pré-condição	Usuário Autenticado. Ter conexão com a internet.
Atores	Usuário
Cenário Principal	1. O usuário entra no álbum. 2. O sistema apresenta as informações do álbum. 3. O sistema busca os jogos referentes ao álbum no servidor. 4. O sistema lista os jogos para o usuário. 5. O sistema permite que o usuário edite, remova ou mova um jogo listado.
Fluxo de Alternativo 1	Após o passo 3 do fluxo principal, caso o usuário tenha escolhido alterar o jogo: 1. O sistema abre a tela de cadastro de jogos com os dados do jogo. 2. O usuário altera os campos do formulário desejado. 3. O usuário clica no botão de salvar. 4. O sistema persiste os dados no banco. 5. O sistema retorna ao fluxo principal.
Fluxo de Alternativo 2	Após o passo 3 do fluxo principal, caso o usuário tenha escolhido excluir o jogo: 1. O sistema abre uma tela de confirmação de exclusão. 2. O usuário confirma a exclusão. 3. O sistema apaga o registro do banco de dados. 4. O sistema retorna ao fluxo principal.
Fluxo de Alternativo 3	Após o passo 3 do fluxo principal, caso o usuário tenha escolhido mover o jogo: 1. O sistema abre uma tela com os álbuns do usuário. 2. O usuário escolhe para qual álbum deseja mover o jogo. 3. O sistema altera o registro do banco de dados. 4. O sistema retorna ao fluxo principal.
Fluxo de exceção 1	No passo 1 do Alternativo 2, caso o usuário não queira excluir o jogo: 1. O sistema retorna ao fluxo principal.
Pós Condição	Lista de jogos atualizado. Dados sincronizados entre servidor e dispositivos.

Fonte: elaborado pelo autor.

Quadro 14 – Caso de Uso UC04

UC04 – Jogar jogos	
Descrição	Permite que o usuário realize uma atividade.
Pré-condição	Usuário Autenticado. Ter jogos cadastrados.
Atores	Usuário
Cenário Principal	1. O usuário entra na tela de jogos. 2. O sistema lista todos os jogos do usuário, indicando o tipo de jogo. 3. O usuário escolhe uma atividade. 4. O sistema abre a tela do respectivo jogo. 5. O usuário realiza as ações de jogatina. 6. O sistema verifica se o usuário ganhou ou perdeu a partida. 7. O sistema apresenta a mensagem. 8. O sistema volta ao passo 2 quando o jogo termina.
Pós Condição	Atividade executada

Fonte: elaborado pelo autor.

Quadro 15 – Caso de Uso UC05

UC05 – Compartilhar jogos	
Descrição	Permite a consulta, edição, remoção e movimentação do jogo
Pré-condição	Usuário Autenticado. Ter conexão com a internet. Ter jogos compartilhados
Atores	Usuário
Cenário Principal	1. O usuário entra no álbum. 2. O sistema apresenta as informações do álbum. 3. O sistema busca os jogos referentes ao álbum no servidor. 4. O sistema lista os jogos para o usuário com o botão de mover. 5. O usuário clica em mover. 6. O sistema apresenta uma tela com todos os usuários cadastrados. 7. O usuário escolher com quem deseja compartilhar o jogo. 8. O sistema altera o banco de dados. 9. O sistema retorna ao passo 2.
Fluxo de Alternativo 1	No passo 1 do fluxo principal, caso o usuário queira importar o jogo: 1. O usuário abre a tela de jogos. 2. O sistema disponibiliza um botão de compartilhamento. 3. O usuário entra na tela de compartilhamento. 4. O sistema exibe os jogos que foram compartilhadas com o usuário. 5. O usuário escolhe aceitar ou recusar. 6. O usuário aceita o jogo. 7. O sistema abre uma tela com os álbuns do usuário. 8. O usuário escolhe para qual álbum deseja mover o jogo. 9. O sistema altera o registro do banco de dados. 10. O sistema retorna ao passo 1.
Fluxo de exceção 1	No passo 6 do Alternativo 1, caso o usuário recusa o jogo: 1. O sistema exclui o jogo do banco de dados. 2. O sistema retorna ao passo 1.
Pós Condição	Atividades compartilhadas. Dados sincronizados entre usuários.

Fonte: elaborado pelo autor.

Quadro 16 – Caso de Uso UC06

UC06 – Incluir conteúdo multimídia	
Descrição	Permite que o usuário inclua imagem ou som
Pré-condição	Usuário Autenticado.
Atores	Usuário
Cenário Principal	1. O usuário clica em adicionar conteúdo. 2. O sistema apresenta as opções de câmera e galeria. 3. O usuário escolhe uma opção. 4. O sistema abre a galeria. 5. O usuário escolhe o arquivo desejado. 6. O sistema carrega e apresenta o arquivo. 7. O usuário salvo o conteúdo. 8. O sistema realiza o upload do arquivo para o armazenamento.
Fluxo de Alternativo 1	No passo 3 do fluxo principal, caso o usuário escolha a câmera: 1. O sistema abre a câmera padrão do dispositivo 2. O usuário tira a foto e confirma. 3. O sistema retorna ao passo 6 do cenário principal.
Pós Condição	Carregamento do conteúdo multimídia.

Fonte: elaborado pelo autor.

Quadro 17 – Caso de Uso UC07

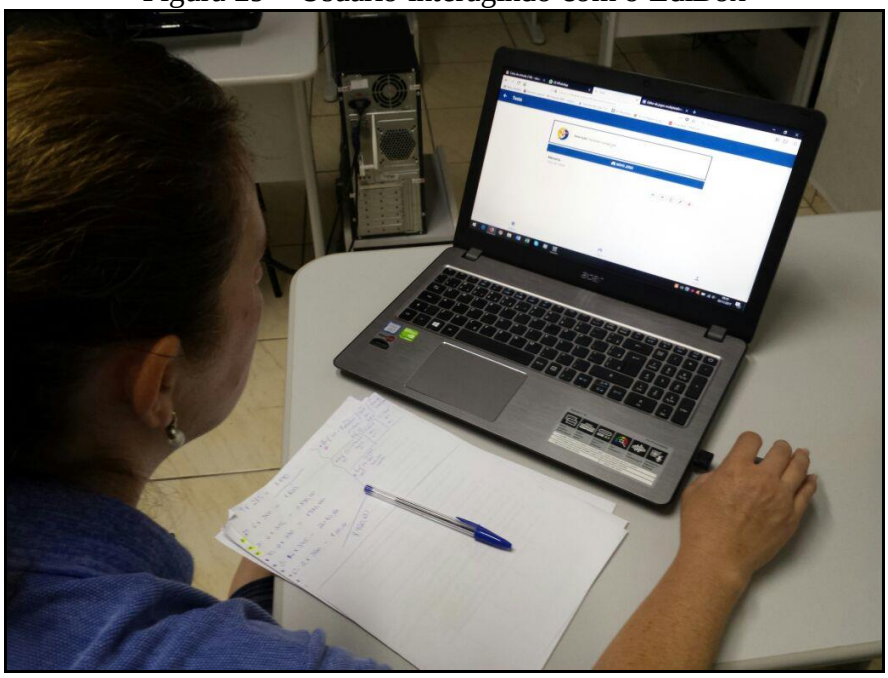
UC07 – Manter álbuns	
Descrição	Permite a consulte, edite, remova e crie novos álbuns
Pré-condição	Usuário Autenticado. Ter conexão com a internet.
Atores	Usuário
Cenário Principal	1. O usuário entra no editor. 2. O sistema apresenta o botão para adicionar álbum. 3. O usuário clica no botão. 4. O sistema abre uma tela de cadastro de álbum. 5. O usuário preenche o formulário. 6. O usuário pode adicionar uma imagem. 7. O usuário clica em salvar. 8. O sistema persiste os dados no banco. 9. O sistema retorna ao passo 1.
Fluxo de Alternativo 1	Após o passo 1 do fluxo principal, caso o usuário já tenha um álbum: 1. O sistema apresenta a lista de álbum. 2. O sistema disponibiliza para cada álbum o botão de editar e excluir ou abrir. 3. O usuário escolhe uma opção. 4. O usuário escolhe editar. 5. O sistema abre uma tela de cadastro de álbum com os campos preenchidos. 6. O usuário altera as informações do formulário. 7. O usuário clica em salvar. 8. O sistema altera as informações no banco de dados. 9. O sistema redireciona o usuário para o passo 1 do cenário principal.
Fluxo de Alternativo 2	Após o passo 2 do fluxo alternativo 1, caso o usuário tenha escolhido excluir o álbum: 1. O sistema abre uma tela de confirmação de exclusão. 2. O usuário confirma a exclusão. 3. O sistema apaga o registro do álbum do banco de dados. 4. O sistema apaga o registro dos jogos referentes ao álbum no banco. 5. O sistema retorna ao cenário principal.
Fluxo de Alternativo 3	Após o passo 2 do fluxo alternativo 1, caso o usuário tenha escolhido abrir o álbum: 1. O usuário efetua a operação desejada a partir do UC03.
Fluxo de Alternativo 4	Após o passo 2 do fluxo alternativo 1, caso o usuário tenha escolhido adicionar imagem ao álbum: 1. O usuário efetua a operação desejada a partir do UC06.
Fluxo de exceção 1	No passo 1 do Alternativo 2, caso o usuário não queira excluir o jogo: 1. O sistema retorna ao fluxo principal.
Pós Condição	Lista de jogos atualizado. Dados sincronizados entre servidor e dispositivos.

Fonte: elaborado pelo autor.

APÊNDICE B – Experimento do EdiBox

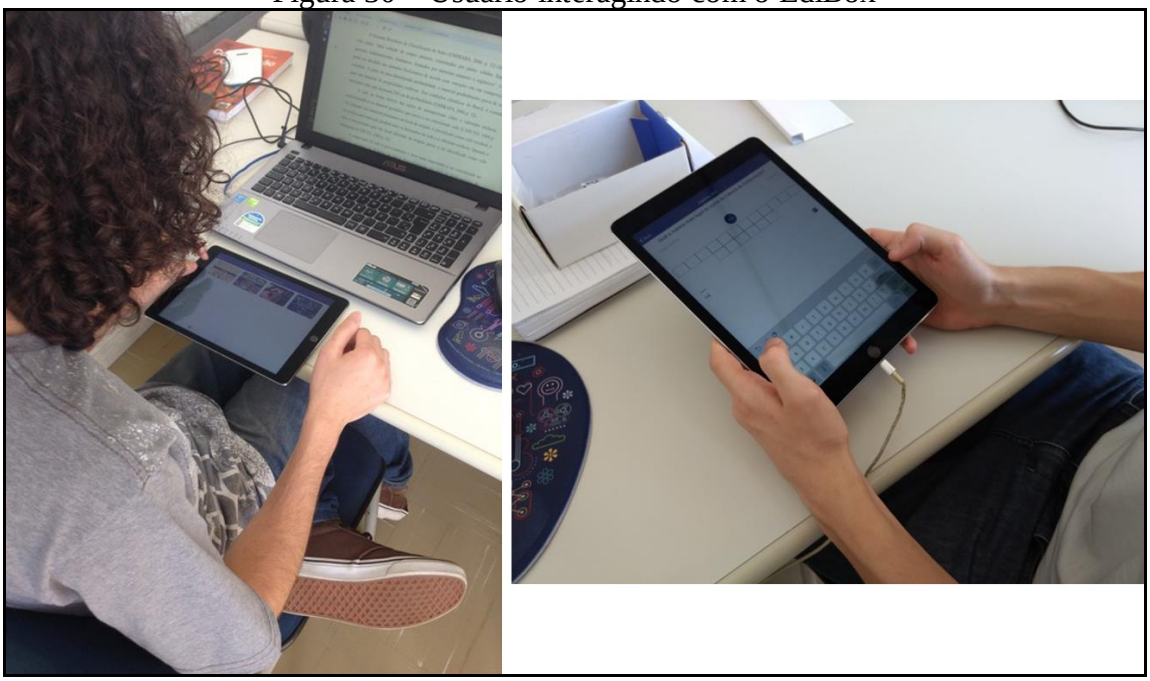
Neste apêndice são apresentadas fotos do teste realizado no dia 29 de novembro de 2017. É possível observar na Figura 29 e Figura 30 o editor sendo utilizado em múltiplos dispositivos pelos usuários especialistas.

Figura 29 – Usuário interagindo com o EdiBox



Fonte: elaborado pelo autor.

Figura 30 – Usuário interagindo com o EdiBox



Fonte: elaborado pelo autor.

APÊNDICE C – Questionário do experimento

Neste apêndice são apresentados o questionário e o roteiro de testes que o usuário seguiu para o testar o EdiBox: Editor de jogos multiplataforma. As Figura 31 e Figura 32 mostram as perguntas de perfil do usuário. As Figura 33 a 37 representam o roteiro com as instruções para o usuário realizar algumas atividades no editor, testando desta forma as funcionalidades. A Figura 41 exibe as perguntas sobre usabilidade do editor.

Figura 31 – Perguntas sobre o perfil do usuário

Editor de jogos multiplataforma - EdiBox

O questionário é parte integrante do Trabalho de Conclusão de Curso intitulado "EdiBox: Editor de jogos multiplataforma" realizado na Universidade Regional de Blumenau pelo acadêmico Marcos Douglas Hoppe 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

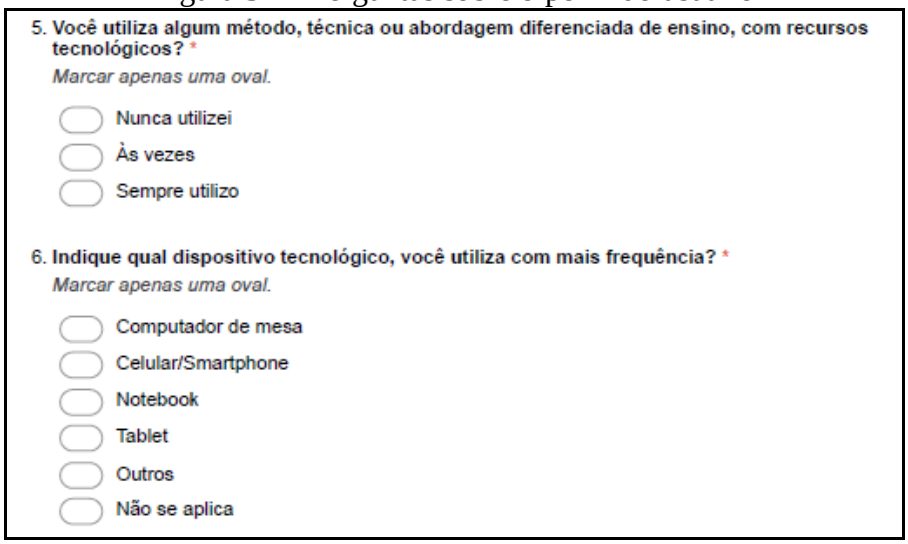
2. **Idade:** *
   Marcar apenas uma oval.
   - ◯ Tenho menos de 5 anos
   - ◯ Tenho entre 6 a 10 anos
   - ◯ Tenho entre 11 a 15 anos
   - ◯ Tenho entre 16 a 20 anos
   - ◯ Tenho entre 21 a 25 anos
   - ◯ Tenho entre 26 a 30 anos
   - ◯ Tenho mais de 30 anos

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
   - ◯ Pós-graduado

4. **Você possui algum dispositivo tecnológico, como celular, tablet ou computador?** *
   Marcar apenas uma oval.
   - ◯ Sim
   - ◯ Não

Fonte: elaborado pelo autor.

Figura 32 – Perguntas sobre o perfil do usuário



5. **Você utiliza algum método, técnica ou abordagem diferenciada de ensino, com recursos tecnológicos?** *

Marcar apenas uma oval.

- ◯ Nunca utilizei
- ◯ Às vezes
- ◯ Sempre utilizo

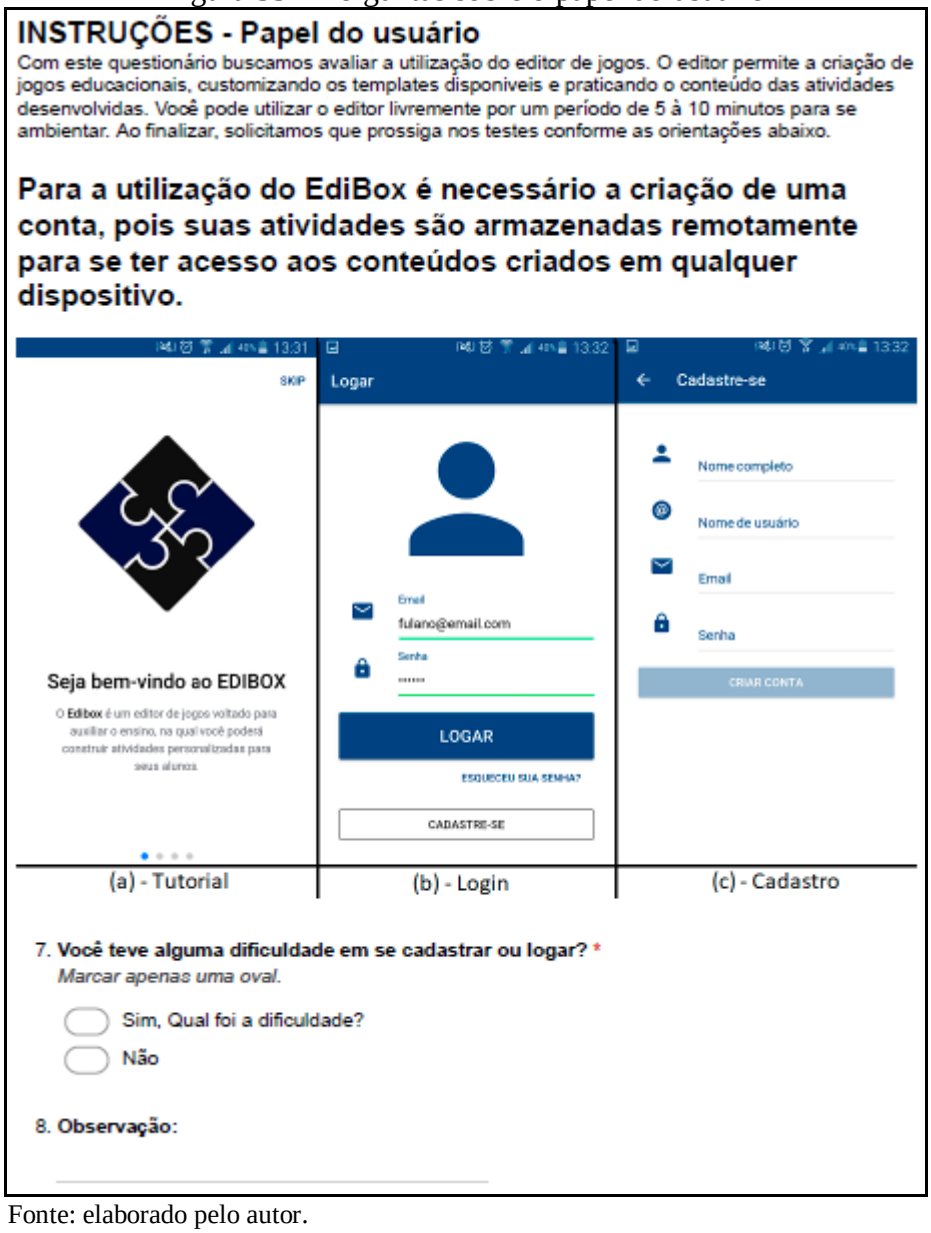
6. **Indique qual dispositivo tecnológico, você utiliza com mais frequência?** *

Marcar apenas uma oval.

- ◯ Computador de mesa
- ◯ Celular/Smartphone
- ◯ Notebook
- ◯ Tablet
- ◯ Outros
- ◯ Não se aplica

Fonte: elaborado pelo autor.

Figura 33 – Perguntas sobre o papel do usuário



INSTRUÇÕES - Papel do usuário

Com este questionário buscamos avaliar a utilização do editor de jogos. O editor permite a criação de jogos educacionais, customizando os templates disponíveis e praticando o conteúdo das atividades desenvolvidas. 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.

Para a utilização do EdiBox é necessário a criação de uma conta, pois suas atividades são armazenadas remotamente para se ter acesso aos conteúdos criados em qualquer dispositivo.

(a) - Tutorial (b) - Login (c) - Cadastro

7. **Você teve alguma dificuldade em se cadastrar ou logar?** *

Marcar apenas uma oval.

- ◯ Sim, Qual foi a dificuldade?
- ◯ Não

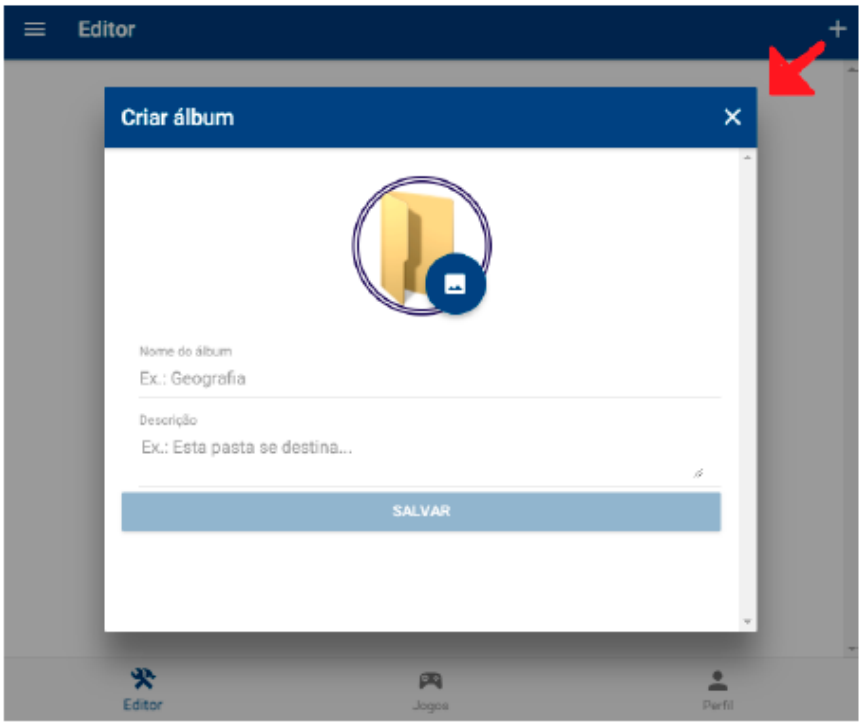
8. **Observação:**

Fonte: elaborado pelo autor.

Figura 34 – Perguntas sobre o papel do usuário

Feito isso você foi direcionada para a pagina de jogos, que no momento não contém nenhum, diante disso vamos para a tela do editor, que pode ser acessada pelo menu lateral ou pela barra inferior, na pagina precisamos criar um álbum.

No editor clique no botão "+" localizado no canto superior direito, irá aparecer outra página para criação de um álbum, preencha os campos e adicione um imagem para o álbum, depois clique em salvar.

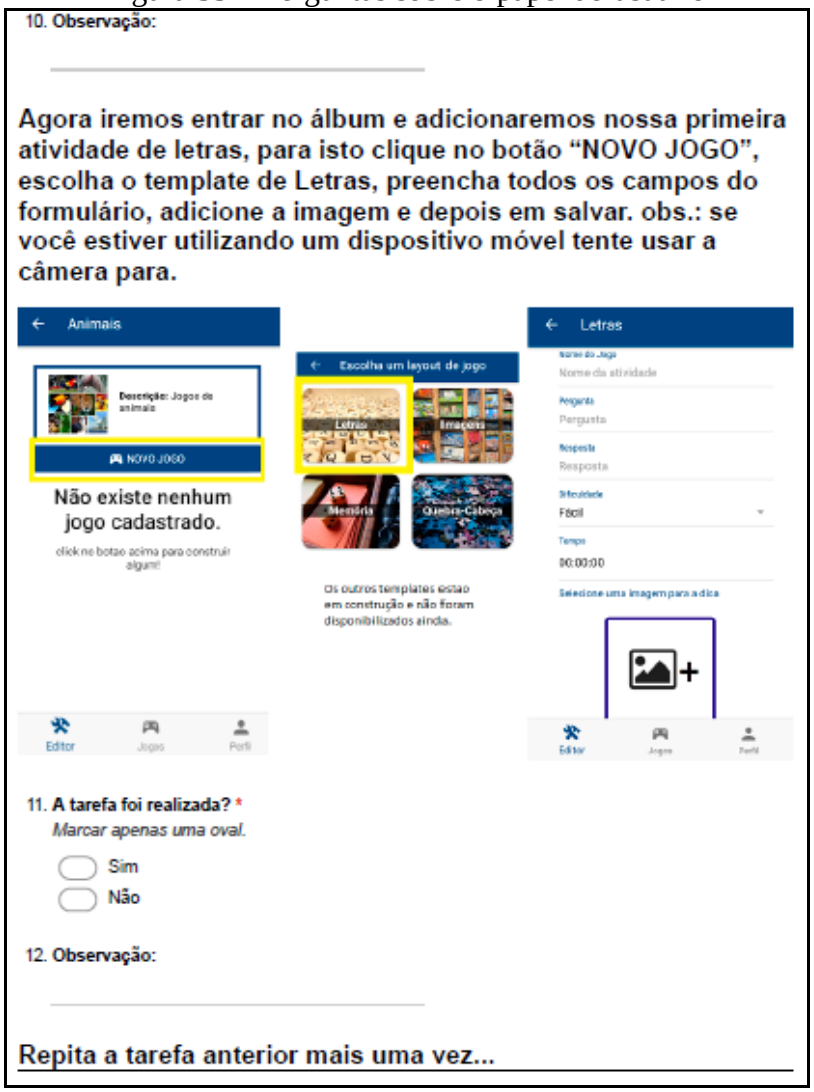


9. **A tarefa foi realizada?** *
Marcar apenas uma oval.

- Sim
- Não

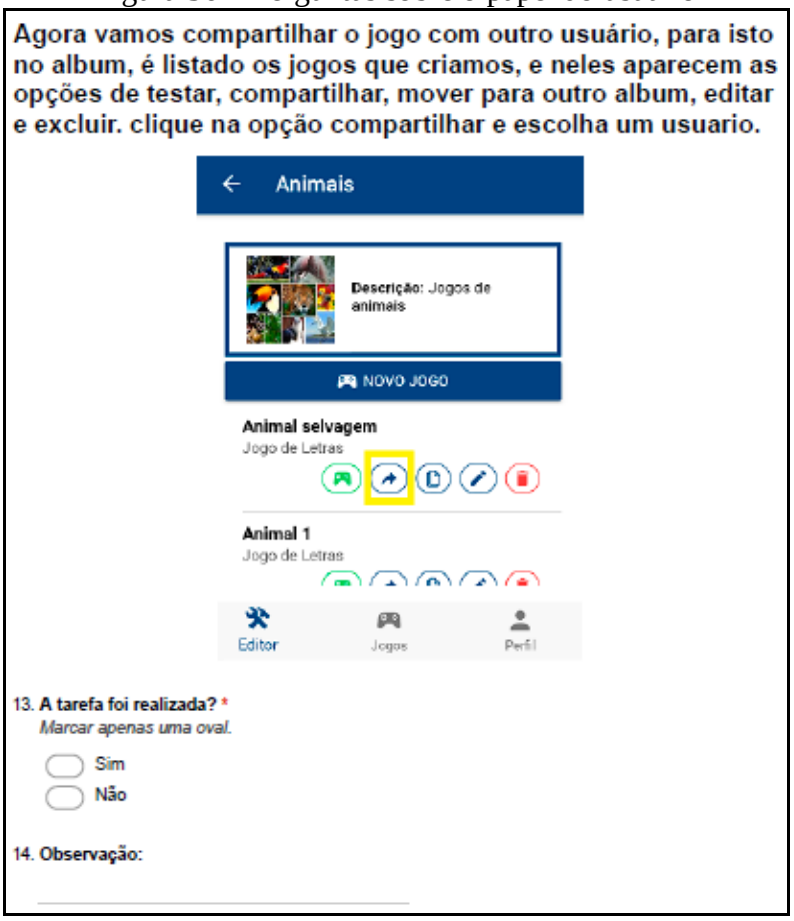
Fonte: elaborado pelo autor.

Figura 35 – Perguntas sobre o papel do usuário



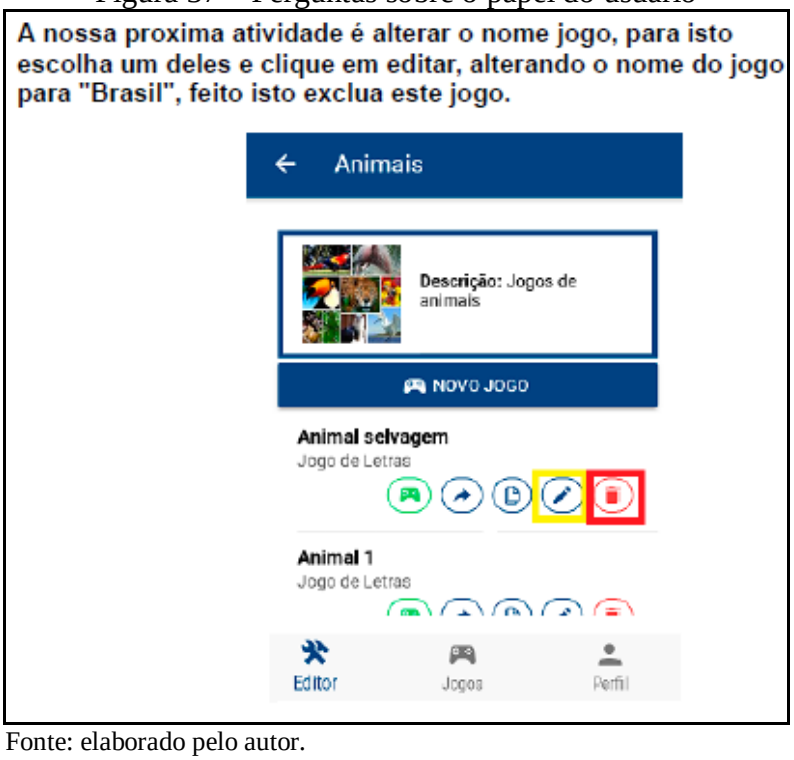
Fonte: elaborado pelo autor.

Figura 36 – Perguntas sobre o papel do usuário



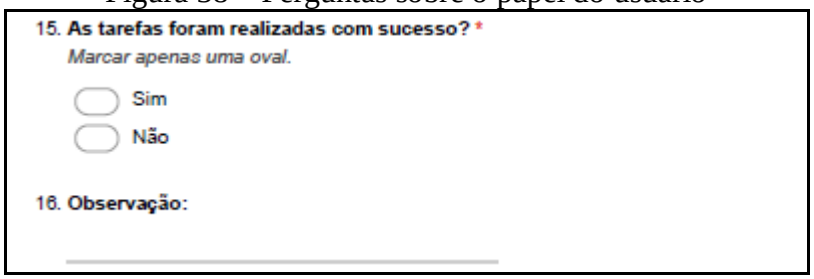
Fonte: elaborado pelo autor.

Figura 37 – Perguntas sobre o papel do usuário



Fonte: elaborado pelo autor.

Figura 38 – Perguntas sobre o papel do usuário

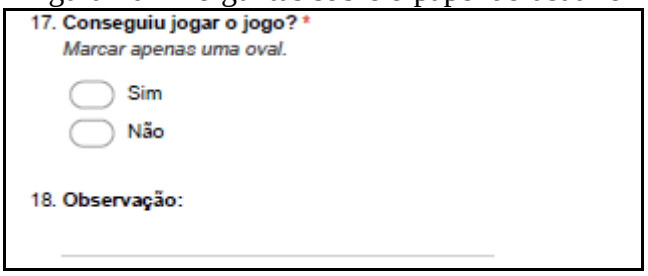


Fonte: elaborado pelo autor.

Figura 39 – Perguntas sobre o papel do usuário

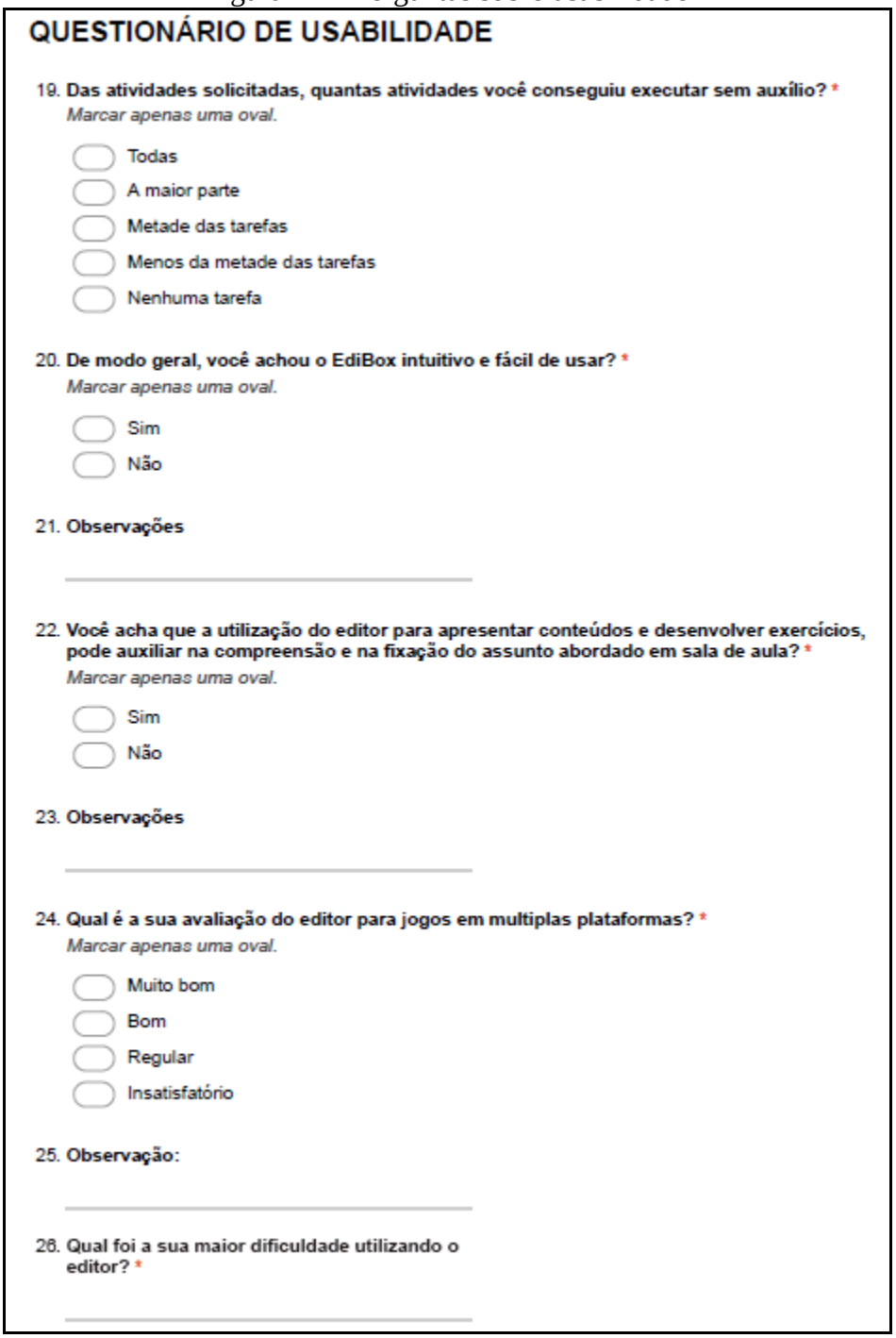
Fonte: elaborado pelo autor.

Figura 40 – Perguntas sobre o papel do usuário



Fonte: elaborado pelo autor.

Figura 41 – Perguntas sobre usabilidade



QUESTIONÁRIO DE USABILIDADE

19. **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

20. **De modo geral, você achou o EdiBox intuitivo e fácil de usar?** *
Marcar apenas uma oval.

- Sim
- Não

21. **Observações**

22. **Você acha que a utilização do editor para apresentar conteúdos e desenvolver exercícios, pode auxiliar na compreensão e na fixação do assunto abordado em sala de aula?** *
Marcar apenas uma oval.

- Sim
- Não

23. **Observações**

24. **Qual é a sua avaliação do editor para jogos em multiplas plataformas?** *
Marcar apenas uma oval.

- Muito bom
- Bom
- Regular
- Insatisfatório

25. **Observação:**

26. **Qual foi a sua maior dificuldade utilizando o editor?** *

Fonte: elaborado pelo autor.

Figura 42 – Resposta da pergunta discursiva

Nao teve dificuldades.
Tive dificulda de colocar a imagem no jogo..
nenhuma
Como entrar com as letras no jogo
na colocação da imagem
definir o tempo e demorei a entender como jogar
Os jogos criados não aparecem na tela "Jogos". Só é possível acessá-los pela tela "Editor"
Não tive dificuldades.
Falta de leitura
Não ter as dicas acompanhado na hora de criar os jogos.
Encontrar a opção de compartilhar jogo

Fonte: elaborado pelo autor.