UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
Bacharelado em Sistemas de Informação
Valdeci Jeferson Pereira Rocha

GRAPPHIA: implementação de mecânica de recompensas

Diamantina, MG
2019

Valdeci Jeferson Pereira Rocha

GRAPPHIA: implementação de mecânica de recompensas

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Sistemas de Informação do Departamento de Computação, como parte dos requisitos exigidos para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Profa. Dra. Luciana Pereira de Assis

Diamantina, MG
2019

Valdeci Jeferson Pereira Rocha
Grapphia/ Valdeci Jeferson Pereira Rocha. – Diamantina, MG, 2019-
77 p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Luciana Pereira de Assis

Trabalho de Conclusão de Curso
Universidade Federal dos Vales do Jequitinhonha e Mucuri, 2019.

1. Jogos Digitais Educacionais. 2. Recompensas. 3. Engajamento I. Profa. Dra. Luciana Pereira de Assis. II. Universidade Federal dos Vales do Jequitinhonha e Mucuri. III. Faculdade de Ciências Exatas. IV. GRAPPHIA:implementação de mecânica de recompensas visando aumento do engajamento dos usuários

Valdeci Jeferson Pereira Rocha

GRAPPHIA: implementação de mecânica de recompensas

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Sistemas de Informação do Departamento de Computação, como parte dos requisitos exigidos para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Profa. Dra. Luciana Pereira de Assis

Aprovado em: 11/07/2019.

Luciana Pereira de Assis

Profa. Dra. Luciana Pereira de Assis
Orientadora

Marcelo Ferreira Rego

Prof. Me. Marcelo Ferreira Rego
UFVJM - Universidade Federal dos Vales do Jequitinhonha e Mucuri

Adriana Nascimento Bodolay

Profa. Dra. Adriana Nascimento Bodolay
UFVJM - Universidade Federal dos Vales do Jequitinhonha e Mucuri

Diamantina, MG

Dedicado este trabalho a minha família, aos meus amigos e meus mentores, que me acompanharam e contribuíram em mais essa caminhada.

AGRADECIMENTOS

Primeiramente gostaria de agradecer aos meus pais, Zenilda e Valdeci, pelo amor, incentivo e apoio incondicional, que me possibilitaram a oportunidade de estudar fora, e seguir esse sonho. Devo agradecer inicialmente também a meu irmão Paulo, e cunhada Madalena pelos puxões de orelha e apoio, apesar da distância. Devo expressar a minha gratidão a outras duas pessoas que também seguiram comigo todo esse tempo, Thais e Martha, pelo apoio e incentivo, acreditando em mim mais do que eu mesmo em alguns momentos.

Agradeço a minha orientadora Profa. Dra. Luciana Pereira de Assis pela orientação, apoio, confiança e paciência durante a realização deste trabalho.

Agradeço imensamente aos meus amigos de Diamantina, especialmente meus parceiros de "República Sparta DTNA", pela paciência, companheirismo, irmandade e por sempre me ajudarem com o que era necessário, além do conhecimento e convívio que construímos. Agradeço a todos os nossos agregados da República Mexicanas e República Apollo XVII, entre outros, pelo acolhimento e pelos momentos vividos.

Agradeço a todos os meus professores que proporcionaram o conhecimento, e a evolução do caráter, no processo de formação profissional e também pessoal. E a todos que direta ou indiretamente, fizeram parte da minha formação, o meu muito obrigado!

“Understand the things I say
Don’t turn away from me
’Cause I spent half my life out there
You wouldn’t disagree ...
(THE CRANBERRIES, Ode to my family)

RESUMO

A utilização das tecnologias nos processos educacionais vem se tornando excelentes ferramentas para auxilio no processo de aprendizagem, principalmente na aprendizagem de crianças. Dentre as tecnologias mais utilizadas neste processo de ensino-aprendizagem, temos os jogos digitais, por tornarem o processo de aprendizagem da criança uma atividade lúdica e voluntária. Para serem considerados como educacionais, estes jogos necessitam de um bom embasamento pedagógico e didático, porém, na maioria das vezes, questões como criação de estratégias que visem engajar os jogadores com o jogo são negligenciadas. Engajar o jogador resume em motivá-lo a interagir mais com a aplicação, gerando mais tempo de interação entre jogo e o jogador, e maior aprendizado, isso no caso dos jogos digitais educacionais. O Grapphia é um destes jogos digitais educacionais, e que auxilia no ensino da Língua Portuguesa (LP), mais em especifico da ortografia. Atualmente, o Grapphia tem implementado poucas estratégias de engajamento dentro dele. O objetivo deste trabalho é implementar uma técnica de engajamento dentro do jogo, denominada *Achievements ou Badges*, que busque aumentar a motivação do jogador dentro do jogo.

Palavras-chave: Jogos Digitais Educacionais. Recompensas. Engajamento. Unity.

ABSTRACT

The use of technologies for educational processes has become useful tools in auxiliary processes of learning, especially for children. Among the most used technologies in this teaching-learning process, lies the digital games, therefore turning this learning method into a playful and voluntary activity for the children. To be held as educational, these games need a strong pedagogic and didactic base, and for most of the time, matters like strategic foundations that aim to engage the players with the game are neglected. To engage a player in the game sum up to interact with the application, creating more interacting time between the game and the player, thus leading for greater learning in these educational digital games. The Grapphia is among these educational digital games that assists in the learning of the Portuguese language (LP), in particular for the spelling. Currently, the Grapphia has been implementing a few engaging strategies within it. The point of this work is to implement an engaging technique for the game, so called "Achievements" or "Badges" that pursues to increase the player's motivation within the game.

Keywords: Educational Digital Games. Rewards. Engagement. Unity.

LISTA DE ILUSTRAÇÕES

LISTA DE CÓDIGOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SGBD	Sistemas de Gestão de Base de Dados
JSON	Notação de Objetos JavaScript
NoSql	Bancos de dados não relacionais
IDE	Ambiente Integral de Desenvolvimento
UFVJM	Universidade Federal dos Vales do Jequitinhonha e Mucuri

SUMÁRIO

1 INTRODUÇÃO

O presente trabalho propõe avultar o aplicativo Grapphia (ASSIS et al., 2017), criado através da ferramenta Unity (UNITY, 2017), introduzindo uma mecânica de recompensas derivada da técnica de engajamento *Achievements ou Badges*. Essa incrementação tem como objetivo a criação de uma estratégia que vise o aumento da motivação dos jogadores. Este capítulo tem por finalidade apresentar o problema situado no contexto, evidenciando a motivação e os objetivos propostos.

O Grapphia é um jogo digital educacional, desenvolvido para dispositivos móveis, que visa auxiliar o processo de aprendizagem da Língua Portuguesa, focado no "ensino da grafia de palavras que possuem letras ou dígrafos concorrentes, que representam o mesmo som"(ASSIS et al., 2017, p. 3), direcionado a crianças alfabetizadas de 8 a 10 anos de idade. O ensino ortográfico do jogo se da através da complementação das palavras, que são apresentadas dentro da aplicação, e o usuário faz a escolha entre as duas letras apresentadas na tela. O jogo segue a ideia de que a aprendizagem desse tipo de palavras depende do uso da memória, cada palavra deve ser aprendida de forma individual (ALVARENGA, 1995 apud ASSIS et al., 2017). Seguindo essa ideia Assis et al. (2017) cita que a utilização de um jogo digital é uma forma mais apropriada para apresentar os exercícios de memorização.

A utilização de jogos digitais no âmbito educacional ganham cada vez mais espaço, logo que "o jogo enquanto função pedagógica é um instrumento que permite à criança sentir emoções satisfatórias e consequentemente, vinculado à atividade de construção de conhecimentos, atingirá seu real papel na educação"(SILVA, 2006, p. 22). Quanto mais prazerosa e divertida a atividade, maior o tempo de interação entre jogo e jogador. Podemos fazer uma relação do tempo de jogo com o nível de aprendizado adquirido, especificamente para o Grapphia, sendo que quanto maior o numero de apresentações da palavra ao jogador, mais ele estará exercitando a memorização das mesmas. Analisando esses pontos, buscar por meios de aumentar o engajamento do jogador quanto a utilização do Grapphia torna-se indispensável. Assim sendo, a proposta deste trabalho é a implementação de uma técnica de engajamento, em especifico da técnica *Achievements ou Badges*.

A técnica de engajamento *Achievements ou Badges* consiste na utilização de recompensas, por meio de pequenos prêmios virtuais, que podem ser *bottons* ou insígnias, entregues aos jogadores após a realização de alguma tarefa ou obtenção de alguma conquista (SENA; COELHO, 2012). Este tipo de estratégia funciona como uma pequena receita de bolo, pois dá ao desenvolvedor um passo-a-passo para realizar sua implementação, que promete aumento da motivação dos usuários quanto a utilização do jogo ou aplicativo, buscando aumentar o tempo de interação e utilização dos mesmos.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo geral deste trabalho consiste na incrementação do aplicativo Grapphia, desenvolvido com uso do software Unity e outras ferramentas, implementando uma estratégia de engajamento por meio da mecânica de recompensas dentro do jogo.

1.1.2 Objetivos Específicos

- Estudar as ferramentas de desenvolvimento;
- Estudar as estratégias existentes de engajamento por meio de recompensas que possam ser incrementadas a aplicação;
- Buscar aumentar o engajamento dos usuários quanto o uso da aplicação.

1.2 Organização

Este trabalho segue dividido em 5 capítulos. O Capítulo 1, e atual capítulo, estabeleceu uma breve introdução sobre o projeto a ser desenvolvido, bem como os objetivos pretendidos.

No Capítulo 2 é apresentada uma revisão dos conceitos básicos para entendimento do trabalho, desde um breve definição sobre jogos em geral, até a utilização de jogos digitais no âmbito educacional. No referido capítulo também é apresentado o histórico do aplicativo Grapphia, demonstrando sua evolução desde o jogo digital Lança Palavras. Posteriormente, é apresentada uma revisão das estratégias de engajamento, bem como o detalhamento da adotada no presente trabalho. O capítulo também apresenta as ferramentas utilizadas durante o processo de implementação do trabalho, as mesmas utilizadas no desenvolvimento da aplicação como um todo.

O Capítulo 3 apresenta os passos realizados para implementação da estratégia de recompensas, mostrando as inserções e edições de cenários, criação de *scripts*, entre outros.

No Capítulo 4 são apresentados os testes feitos para mostrar as interações dos usuários, com a obtenção das moedas e compra/liberação dos novos livros, e posteriormente a análise dos mesmos.

O Capítulo 5 apresenta as conclusões e os trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo tem como objetivo apresentar conceitos básicos para o entendimento do trabalho. A seção 2.1 apresenta os conceitos sobre jogos digitais, e a seção 2.2 sobre os jogos digitais educacionais. A seção 2.3 apresenta a evolução do jogo digital educacional Grapphia. A seção 2.4 fala sobre motivação e processo de engajamento dentro dos jogos digitais. A seção 2.5 detalha o processo de recompensas dentro de jogos, seguido da subseção 2.5.1 que explica uma dessas técnicas de engajamento utilizadas nesse processo. A seção 2.6 lista alguns exemplos de jogos digitais educacionais, e seus processos de recompensas. Por último a seção 2.7 apresenta o software Unity, e demais ferramentas utilizadas no desenvolvimento deste trabalho.

2.1 Jogos Digitais

Na década de 60, com a início da grande expansão da tecnologia, temos o surgimento dos primeiros jogos digitais. Atualmente, o primeiro, e um dos principais meios de acesso ao mundo da tecnologia, tanto para os jovens e crianças, é o jogo digital. Geralmente, seu primeiro contato com os equipamentos eletrônicos acontece por meio de algum vídeo game (GROS, 2003 apud SAVI; ULBRICHT, 2008).

Um jogo digital ou jogo eletrônico é uma atividade lúdica, composta por ações e decisões que resultam numa condição final (SCHUYTEMA, 2008 apud LUCCHESE; RIBEIRO, 2009), que são limitadas por um conjunto de regras e por um universo, regidos por um programa de computador (LUCCHESE; RIBEIRO, 2009).

Segundo Silva et al. (2009), um jogo digital possui representações em três universos:

- imaginário: na representação imaginária, tudo é subjetivo, ocorrendo na mente do jogador;
- real: em sua representação real, refere-se aos elementos reais do jogo, os componentes físicos;
- virtual: na representação virtual é a sua representação em bits, normalmente apresentada como imagens e/ou sons.

Lucchese e Ribeiro (2009) dizem que os jogos digitais estão intimamente ligados aos computadores, como desktops, consoles, dispositivos móveis. A utilização dos recursos computacionais são a base de diferenciação entre os jogos digitais e os não-digitais.

Sobre a sua classificação, não há uma divisão em comum na literatura, coexistindo diversas classificações e critérios base. Normalmente, eles fazem o agrupamento seguindo a existência de características e critérios similares, como o objetivo do jogo, o contexto de inserção do jogador, a forma de condução do personagem em jogo, e interação com o ambiente (LUCCHESE; RIBEIRO, 2009).

Battaiola (2000) propõe uma classificação em relação aos jogos digitais atuais, apresentando a distribuição dos jogos em oito grupos:

- Estratégia: consiste em jogos que a principal ferramenta é a análise critica e tomadas de decisões na busca de um objetivo, apresentando mais um desafio intelectual;
- Simuladores: são jogos que buscam a imersão do jogador em primeira pessoa, com representações do mundo real;
- Aventura: são jogos que fazem a mistura das ações embasadas no raciocínio e reflexo do jogador, a soluções de enigmas implícitos seguem a base de desafios, sem enfoque no tempo de resolução, apenas no resultado;
- Infantil: são jogos voltados ao público infantil, utilizando quebra-cabeças ou histórias simples, com enfoque na diversão da criança;
- Passatempo: são jogos simples que buscam desafiar o jogador com quebra-cabeças rápidos, sem um enredo bem elaborado, visando a busca da maior pontuação;
- RPG: são jogos que fazem uma representação dos RPG convencionais de mesa, com os mesmos objetivos.
- Esporte: são jogos que seguem a representação computacional dos esportes populares reais, como futebol, basquete, vôlei, entre outros;
- Educação/Treinamento: são jogos que podem possuir características das demais categorias, com diferenças no seu enfoque, que além de buscar a diversão também objetivam critérios didáticos e pedagógicos.

O Grapphia, por ter sido criado para o público infantil, busca ser um desses jogos digitais utilizados pelas crianças nos seus primeiros passos no mundo tecnológico, transformando as suas primeiras interações em momentos também de aprendizado. Segundo a classificação apresentada, conseguimos enquadrá-lo no grupo de jogos de Educação/Treinamento, graças ao seu embasamento e enfoque educacional.

2.2 Jogos Digitais Educacionais

A utilização e consumo de jogos digitais cresce cada vez mais na sociedade atual. Em senso realizado pelo Ministério da Cultura no ano de 2018 (SAKUDA et al., 2018), os jogos digitais devem obter um crescimento de 7% ao ano no mundo, girarando em torno de 16% para o mercado brasileiro, atingindo US$ 1,4 bilhão, no período de 2017 a 2021. O cenário Brasileiro se mostra cada vez mais promissor, e atualemnte reconhecido como o ponto central de jogos no mercado da América Latina, graças aos investimentos em eventos de eSports[1], e também ao grande número de consumidores existente (SAKUDA et al., 2018).

Segundo Tarouco et al. (2004) os jogos podem ser utilizados como ótimas ferramentas instrucionais, pois eles divertem, motivam, facilitam o aprendizado e ainda aumentam

[1] Esportes eletrônicos.

a capacidade de retenção de conteúdos passados, como um exercício das funções mentais e intelectuais. A utilização destas tecnologias no âmbito educacional se tornam mais presente, graças a essas características de desenvolvimento intelectual e engajamento.

Os jogos digitais educacionais tem a intenção de buscar a união entre entretenimento e o ensino ao mesmo tempo, transformando todo e qualquer momento de interação prazeroso e produtivo. Os jogos digitais educacionais são todos aqueles jogos digitais que podem ser utilizados para algum fim educacional ou estiverem pedagogicamente embasadas (TAROUCO et al., 2004).

Amate (2007) cita que os jogos digitais educacionais levam em conta principalmente os critérios didáticos e pedagógicos, demonstrando ser essa a principal diferença entre eles e os jogos que visam apenas a diversão. A necessidade do embasamento nas práticas pedagógicas adequadas ao usuário é clara para a consideração do jogo como educacional (SILVA et al., 2009).

2.3 Grapphia

O jogo digital educacional Grapphia, criado pelos discentes do curso de Sistemas de Informação da UFVJM, nasce dos autores V. C. Santos (2016) e Leal (2016), criado para auxiliar no processo de ensino-aprendizagem da ortografia, voltado para o público infantil. Os dois autores criaram em conjunto, o Laça Palavras, produto de trabalho de conclusão de curso dos autores, disponibilizado para dispositivos Android, utilizando o software de criação de jogos Unity. A figura 1 apresenta a tela inicial da aplicação.

Figura 1 – Tela inicial do Laça Palavras

Fonte: Leal (2016)

Leal (2016, p. 43) diz que "o jogo tem como foco tornar bem mais divertido e menos cansativo os treinos ortográficos das palavras, principalmente palavras que possuem o caso da concorrência". Essa concorrência se dá quando existem duas letras aptas a representar o mesmo som, no mesmo contexto. No Grapphia, é apresentado ao usuário diversas dessas palavras

mencionadas, para serem completadas corretamente de acordo com as opções disponibilizadas. Os autores dividiram as palavras em 4 grupos, como apresentado por Leal (2016):

> "O jogo foca em quatro grupos de palavras que se enquadram no caso da concorrência. Dentre elas, palavras que contém sílabas com “S” ou “Z”, mas que possui som de [z] (exemplo a palavra “casa”), palavras que contém sílabas com “SS” ou Ç, mas que possui som de [s] (exemplo a palavra “passo”), palavras que contém sílabas com “L” ou “U”, mas que possui som de [u] (exemplo a palavra “alça”) e palavras que contém sílabas com “G” ou “J”, mas que possui som [g] (exemplo a palavra “majestade”)"(LEAL, 2016, p.43).

A figura 2 mostra a tela de jogo apresentando uma palavra com caso de concorrência entre as letras "L"e "U".

Figura 2 – Tela de jogo do Laça Palavras

Fonte: V. C. Santos (2016)

Outro ponto trabalhado pelos autores foi o Ajuste Dinâmico de Dificuldade, ou "Balanceamento Dinâmico de Dificuldade"(V. C. SANTOS, 2016, p.40), fazendo com que alguns componentes disponíveis no jogo sirvam de avaliador de desempenho do usuário, e realizar ajustes de dificuldade a medida que o jogador interage com o jogo, tornando sua experiência mais divertida e consistente (KUANG, 2012 apud V. C. SANTOS, 2016).

Posteriormente, buscando aumentar os cenários de jogo, melhorias estéticas, e simplificação do código, foi criado o Grapphia, vindo de uma transformação do Laça Palavras, proposto pelos autores Assis et al. (2017) e M. J. G. Santos (2019). A aplicação, assim como o Laça Palavras, foi desenvolvida com o uso do software Unity. A tela inicial do Grapphia e mostrada na figura 3.

O Grapphia continua com o foco neste grupo de palavras concorrentes, também chamadas de "palavras irregulares"(ASSIS et al., 2017, p.2), que não são controladas por uma regra ortográfica, e existem letras concorrentes para sua representação.

Buscando uma solução para a dificuldade na aprendizagem dessa categoria de palavras, os autores Assis et al. (2017) se embasaram na metodologia de ensino de Alvarenga (1995), que diz que a utilização de exercícios para memorização desse grupo de palavras serve

Figura 3 – Tela inicial do Grapphia

Fonte: Assis et al. (2017)

de ajuda no seu aprendizado. A utilização de jogos digitais nesse processo de memorização se mostram como ferramentas mais atrativas e divertidas do que a utilização de dicionários para busca de apropriação desse conteúdo.

Na primeira versão, ainda como Laça Palavras, a apresentação das palavras era em apenas um cenário de jogo. Porém, no Grapphia, passou a ser dividido em grupos de palavras, cada grupo representado em um livro com determinada temática e em um jogo. A figura 4 apresenta a tela de escolha dos livros onde estão divididos os grupos de palavras, que são: *A Fazenda*, *Passeio no Zoo* e *Sol de Verão*.

Figura 4 – Tela da estante de livros do Grapphia

Fonte: Assis et al. (2017)

O livro *A Fazenda* apresenta as palavras do grupo "S"e "Z". A figura 5 ilustra a capa do livro, mostrando a tela de leitura da história. Ao selecionar o livro na estante e acessar a história, o usuário necessita realizar a sua leitura, bastando arrastar o dedo do canto inferior direito ao canto inferior esquerdo para passar as páginas, funcionando como uma interação de

troca de páginas. Após a leitura da história, o usuário pode pressionar o botão “Jogar” e assim passar para a escolha de personagem. A figura 6 apresenta a tela de jogo relacionada ao livro *A Fazenda*. Nela é apresentada a palavra na parte superior da tela, e suas opções de completar a palavra na parte inferior, podendo pressioná-las para realizar a interação, incrementando o número de acertos a cada resposta correta.

Figura 5 – Tela de leitura livro *A Fazenda*

Fonte: Assis et al. (2017)

Figura 6 – Tela de jogo livro *A Fazenda*

Fonte: M. J. G. Santos (2019)

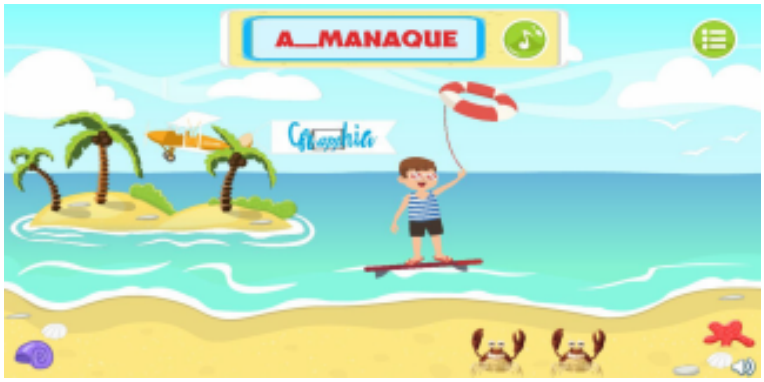
O livro *Sol de Verão* apresenta as palavras do grupo "L"e "U". A figura 7 ilustra a capa do livro, mostrando a tela de leitura da história. O procedimento para acessar sua história é o mesmo do livro *A Fazenda*, bastando apenas selecionar sua opção na estante de livros, realizar a leitura, e seguir para a tela de jogo. A figura 8 apresenta a tela de jogo relacionada ao livro *Sol de Verão*.

O livro *Passeio no Zoo* apresenta as palavras do grupo "G"e "J". A figura 9 ilustra a capa do livro, mostrando a tela de leitura da história. Este livro ainda se encontra em fase de construção.

Figura 7 – Tela de leitura livro *Sol de Verão*

Fonte: Elaborado pelo autor

Figura 8 – Tela de jogo livro *Sol de Verão*



Fonte: Elaborado pelo autor

Além da utilização do Unity, tanto no Laça Palavras e posteriormente Grapphia, foi escolhido o banco de dados relacional SQLite para ajudar na criação da aplicação. Esta ferramenta é *open source*[2] e *serverless*[3]. Por não possuir comunicação com uma rede externa, a extração de dados gerados de cada dispositivo deve ser manual, para depois ser analisada, mostrando ser um processo operoso e lento (ESTANISLAU, 2018).

Com isso foi proposto por Estanislau (2018) a integração do aplicativo com algum servidor. A melhor escolha seria de um banco de dados não relacional, devido a sua alta flexibilidade[4] e escalabilidade[5], em comparação a base de dados relacional (NAYAK; PORIY; POOJARY, 2013 apud ESTANISLAU, 2018). O autor realizou a escolha do Firebase Realtime Database para realizar a integração com a aplicação, que é "um banco de dados não relacional NoSQL, hospedado na nuvem"(ESTANISLAU, 2018). Os dados criados e armazenados são

[2] Código fonte disponibilizado com licença de código aberto. O direito autoral permite realizar estudos, modificações e distribuição do software de graça (CANALTECH, 2018)
[3] Não tem comunicação com qualquer servidor ou rede externa (SQLITE, 2017)
[4] Não tem restrições na estruturação da base de dados
[5] Capacidade de expansão de um sistema sem que haja perca de desempenho

Figura 9 – Tela de leitura livro *Passeio no Zoo*

Fonte: Elaborado pelo autor

sincronizados em tempo real com todos os clientes associados. Com isso foi possível permitir a extração dos dados de maneira mais fácil dos dispositivos, e facilitar a análise futura desses dados obtidos, por meio da base de dados única criada na nuvem.

2.4 Motivação e Engajamento

O pouco de incentivo que o Grapphia utiliza, dentro de jogo, é a apresentação do número de acertos do usuário e montagem de um quebra-cabeça. Como pode ser visto, o Grapphia não se destaca pela utilização de mecanismos de engajamento. Em nenhum trabalho ou levantamento foi buscado a descoberta e utilização de estratégias que fossem focadas no engajamento dos usuário.

É indiscutível atualmente o sucesso e a crescente utilização dos jogos digitais, ou games, no dia-a-dia das pessoas, das mais variadas faixas etárias, seja pela busca de entretenimento, entre outros enfoques de utilização. Segundo Sena e Coelho (2012, p.1), "grande parte deste sucesso e fascínio que os videogames criam só pode ser entendido com base em uma análise dos componentes que integram o sistema de motivação que faz o jogador se manter jogando".

De acordo com Ghozland (2010) a motivação é quem determina se o jogador continuará a jogar após alguns minutos de interação, além do quanto ele irá gastar de tempo jogando e se chegará a terminar o jogo.

Alguns estudos apontam que a motivação é dividida em dois tipos: a intrínseca e a extrínseca (MATLIN, 2004 apud SENA; COELHO, 2012). Sena e Coelho (2012) explicam que a intrínseca remete-se a motivação para trabalhar com aquilo que é considerado interessante, desafiador ou empolgante, enquanto que a extrínseca, se trata da motivação de trabalhar em um determinado assunto visando a obtenção de uma recompensa. O mercado de games utilizam, em geral, a motivação intrínseca, mas também propõe soluções quanto ao outro tipo de motivação.

Existem atualmente no mundo dos games a utilização de várias técnicas de engaja-

mento, que "são técnicas de game design utilizadas para motivar e manter um jogador interessado no game"(SENA; COELHO, 2012, p.3), como se fossem pequenas receitas de bolo que podem ser utilizadas na criação de jogos ou aplicativos, surgindo dessa sua popularização.

> "As Técnicas de Engajamento influenciam no prolongamento do estado motivado de maneira direta, mantendo o jogador mais tempo no jogo. Contudo, normalmente as técnicas não são responsáveis pela motivação inicial, ou seja, o jogador não inicia um game por causa das técnicas de engajamento, mas pode aumentar seu tempo de jogo por causa delas"(SENA; COELHO, 2012, p.4).

Olhando para os jogos digitais educacionais, ao se utilizar dessas técnicas de engajamento em conjunto com o embasamento pedagógico e didático necessário para esse tipo de jogo, pode fazer com que a apresentação de conteúdo didático e nível de aprendizado se torne maior, seguindo o aumento do tempo de interação do jogador com o jogo. Devido a necessidade de aprimorar o Grapphia e das vantagens de utilização de métodos de engajamento, este trabalho se apresenta.

2.5 Recompensas

Existem três elementos base que devem compor a narrativa de um jogo e o próprio design do game: necessidades, desafios e recompensas (SENA; COELHO, 2012). Ghozland (2010) argumenta que o design do game deve construir o ciclo, gerando as necessidades, criação de uma sucessão de desafios e, por fim, gerar recompensação. Ao trabalhar essas três variáveis é possível gerenciar a motivação do jogador, entre outros aspectos.

> "Temos que oferecer desafios ao jogador para entretê-lo e testá-lo junto com recompensas que o motivariam a continuar jogando. Motivar o jogador também é entender suas necessidades com o propósito de cumpri-las. É, portanto, saber, mas também “controlar” o progresso de suas necessidades, a fim de aumentá-las e modificá-las, do começo até o final do jogo"(GHOZLAND, 2010, p.2, tradução nossa)[6].

As recompensas podem ter várias formas, e devem estar em correlação com o universo do jogo e expectativas do jogador. Ela está relacionada a um desafio, teste ou esforço, assim sendo, a recompensa necessita ser proporcional à dificuldade de sua obtenção. O Grapphia atualmente utiliza as informações de progresso (acertos) como forma principal de recompensa dentro do jogo. Outra estratégia utilizada é a montagem de um quebra-cabeça, que vai sendo completado a medida que os acertos vão ocorrendo. A proposta deste trabalho é implementar uma mecânica de recompensas mais correlata e interativa, buscando fechar o ciclo de funcionamento do jogo, e motivar os jogadores a dedicar mais tempo de interação com o jogo.

[6] We have to offer challenges to the player in order to entertain him and test him along with rewards that would motivate him to continue playing. [...] modify them from the beginning till the end of the game.

2.5.1 *Achievements* ou *Badges*

Neste trabalho a técnica de engajamento escolhida foi a *Achievements* ou *Badges*. Sena e Coelho (2012, p. 3) diz que essa estratégia funciona com a recompensação de "pequenos prêmios virtuais na forma de *bottons* ou insígnias, oferecidos aos jogadores depois de realizarem alguma tarefa ou obterem alguma conquista". A cada desafio apresentado, e resultado satisfatório obtido, o usuário recebe sua recompensa.

Na adaptação da técnica para o Graphhia, o usuário receberá moedas virtuais (*Rabbit Coins*) a cada acerto dentro dos modos de jogo. A necessidade de obtenção das moedas se dá pela existência de liberação das histórias em livros presentes no jogo. Todos esses processos vão compor a mecânica de recompensas.

2.6 Exemplos de recompensas dentro de jogos digitais educacionais

As subseções abaixo apresentam jogos digitais educacionais que auxiliam no processo de aprendizagem e alfabetização, dos conteúdos mais diversos, e não só ortográficos, visando a análise das mecânicas de recompensas utilizadas dentro desses jogos.

2.6.1 Descubra as diferenças

O Descubra as diferenças [7] é um jogo que consiste na comparação de duas imagens a procura de erros, ajudando seu filho a melhorar a concentração e raciocínio. Sua tela inicial está apresentada na figura 10.

Figura 10 – Tela inicial Descubra as diferenças

Fonte: *print screen* do jogo Descubra as diferenças

O jogo é introduzido com os vilões que roubam as chaves e um baú do personagem principal (Figura 11) e, com isso, ele precisa passar por todos os níveis, referente ao número de

[7] URL de Download Play Store do Descubra as diferenças: https://play.google.com/store/apps/details?id=com.sinyee.babybus.findCha&hl=pt_BR

vilões, que são 4, os derrotando e recuperando os itens roubados. A figura 12 mostra os níveis de jogo.

Figura 11 – Roubo das chaves

Fonte: *print screen* do jogo Descubra as diferenças

Figura 12 – Mapa com os níveis

Fonte: *print screen* do jogo Descubra as diferenças

Ao finalizar as tarefas de comparação das imagens em cada um dos 4 níveis, o jogador recebe algumas moedas, apresentada na figura 13, que podem ser utilizadas em compras de dicas.

Ao clicar na lupa, as dicas compradas marcam onde estão os erros, um por dica, na comparação das imagens, ajudando o jogador a concluir as tarefas e passar de nível. O custo de cada dica é mostrado ao centro da tela (Figura 14), dentro da silhueta do troféu. O ganho de moedas varia a cada etapa dentro dos níveis.

Figura 13 – Ganho de moedas

Fonte: *print screen* do jogo Descubra as diferenças

Figura 14 – Utilizando a ajuda

Fonte: *print screen* do jogo Descubra as diferenças

2.6.2 *Kids Numbers and Math Lite*

Kids Numbers and Math Lite[8] é um jogo educativo que tenta auxiliar na aprendizagem dos números e matemática básica (operações de soma e subtração). A versão grátis do jogo é a que está sendo usada de exemplo. Sua tela inicial é apresentada na figura 15, onde existem as opções de comprar a versão completa, ver outros jogos dos desenvolvedores e jogar.

Os seus níveis são representados com os quebra-cabeças de animais (Figura 12) que, na versão grátis, disponibiliza 3 destes. Cada nível possui 6 etapas de jogo diversas (Figura 17), que representam desafios de soma, subtração, verificação do maior número presente no cenário, entre outros. A figura 18 demonstra a etapa de jogo voltado ao ensino da adição, onde informa, dentre as sugestões mostradas abaixo, qual a resposta correta para a operação.

Ao finalizar alguma das etapas de jogo, uma parte do quebra-cabeça é entregue como

[8] URL de Download Play Store do *Kids Numbers and Math Lite*: https://play.google.com/store/apps/details?id=zok.android.numbers&hl=pt_BR

Figura 15 – Tela inicial *Kids Numbers and Math Lite*

Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

Figura 16 – Níveis de Jogo

Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

recompensa ao jogador (Figura 19), e a cada nível completado, o quebra cabeça é montado por completo (Figura 20).

2.7 Tecnologias

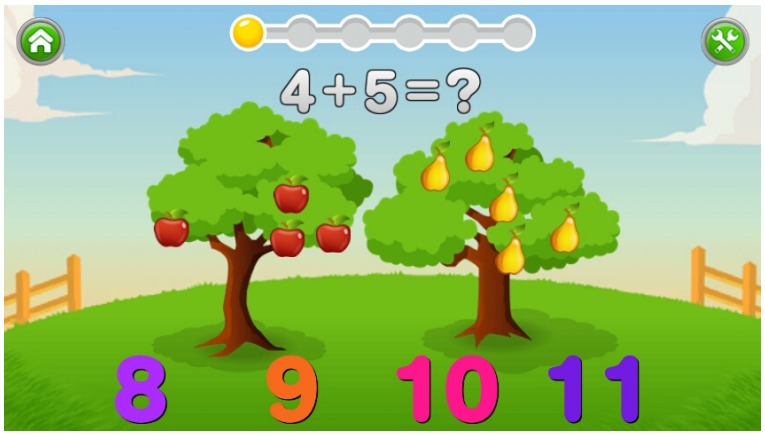
Esta seção tem como objetivo descrever as tecnologias e ferramentas utilizadas para que fosse possível a implementação da mecânica de recompensas dentro do Grapphia. Dentre elas, o Unity 3D, a linguagem C#, o Inkscape e o SQLite.

A subseção 2.7.1 apresenta a ferramenta computacional Unity 3D, a utilização de Scripts e da linguagem C#. A subseção 2.7.2 apresenta a ferramenta SQLite. Por último, a subseção 2.7.3 descreve a ferramenta Inkscape.

Figura 17 – Etapas de Jogo

Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

Figura 18 – Etapa de Adição



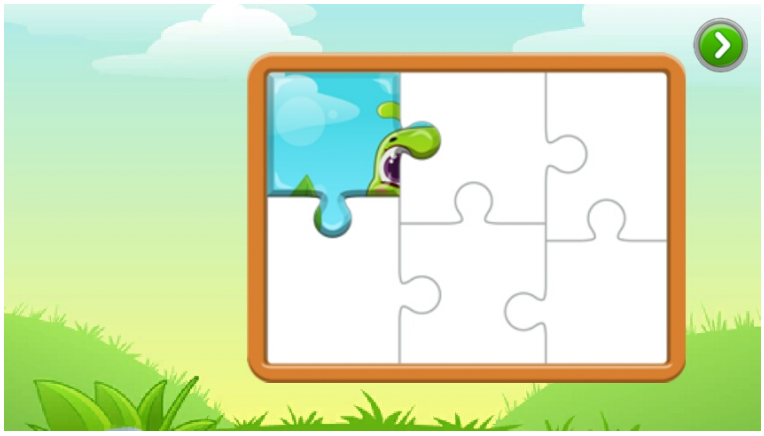
Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

2.7.1 Software Unity

O desenvolvimento de jogos digitais é um dos mercados da tecnologia que mais cresce atualmente. Os avanços tecnológicos nos últimos tempos, e a utilização dos jogos nas mais diversas áreas, não se restringindo só ao entretenimento, gera uma demanda por jogos cada vez mais complexos e de qualidade. A escolha de uma boa ferramenta para produção de jogos é de suma importância, por ajudar a trazer resultados de maneira fácil e ágil, seja para quem está começando no mundo de produção de jogos, até grandes empresas.

O software Unity (comumente conhecido como Unity3D), produzido e comercialmente distribuído pela empresa Unity Technologies, é uma *game engine* (motor de jogos) e um ambiente de desenvolvimento integrado (IDE) para criação de jogos, filmes e animações, entre outros usos. "Um motor de jogos é um software que fornece aos criadores de jogos o conjunto necessário de recursos para criação de jogos de maneira rápida e eficiente"(UNITY, 2018). Por ser um software extremamente poderoso e completo, o Unity é utilizado para desenvolver desde

Figura 19 – Ganhando parte do quebra-cabeça



Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

Figura 20 – Finalizando quebra-cabeça

Fonte: *print screen* do jogo *Kids Numbers and Math Lite*

jogos simples em 2D até jogos realmente complexos e realísticos em 3D, simuladores e realidade virtual.

Ele detém a popularidade entre desenvolvedores independentes e equipes pequenas nos últimos anos, pois existe uma quantidade enorme de tutoriais, cursos profissionalizantes e documentações, inclusive no próprio site do Unity.

O Unity utiliza das linguagens de *script*, C# e Javascript, com isso os jogos desenvolvidos na ferramenta podem ser gerados para as mais diversas plataformas presentes no mercado, dentre elas: dispositivos móveis (Android e iOS), consoles (Xbox, PS4, entre outros), desktops (Mac, Linux e Windows), e até mesmo para serem executados na Web. A ferramenta pode ser instalada nos Sistemas Operacionais Windows, Mac OS X e Linux.

No mercado de desenvolvimento existem outros *game engines* bastante utilizadas também, tais como: Unreal, CryEngine, Cocos2D, Cube, MonoGame, Construct, dentre outras. A versão do software Unity utilizado neste trabalho e a 2018.2.7.

2.7.1.1 *Scripts*

Os *scripts* são "roteiros", ou listas de comandos, que são seguidos por sistemas computacionais e automatizam processos. No software Unity, a utilização dos *scripts* é um ingrediente essencial na criação de todos os jogos. A cada interação do jogador, mudança de cenários, controle de animações, entre outras tarefas, existe a necessidade de um *script* para controlar esses processos e ações.

Para movimentar um simples objeto, seja ele uma bola ou um carro, um *script* é necessário para controlar o comportamento físico do objeto, fazendo com que ele se movimente sozinho, ou quando acontece alguma interação do jogador com o objeto.

O Unity possui uma classe base da qual todos os *script* derivam, chamada MonoBehaviour. Ao utilizar a linguagem C#, é necessária derivação desta classe. Todos os *scripts* devem implementar o método *Start()*, chamado sempre no início da execução do *script*, que define configurações iniciais básicas.

2.7.1.2 C#

Existem no mercado atualmente diversas linguagens de programação que podem ser utilizadas na construção de aplicações, como por exemplo, C, C++, Java, Objective-C, C, Python, Fortran, Perl, dentre outras, cada uma com seus recursos, foco de utilização, e plataforma para desenvolvimento. A linguagem de programação C# foi utilizada no desenvolvimento deste trabalho, com a criação e modificação de scripts.

O C# é uma linguagem de programação criada pela Microsoft, orientada a objetos, e que integra a plataforma .NET, com foco no desenvolvimento de Serviços WEB XML, conhecidos também como Web Services. O princípio da criação e utilização dos Web Services "consiste em permitir que as aplicações, sejam elas da Web ou Desktop, ou ainda middleware, se comuniquem e troquem dados de forma simples e transparente, independente do sistema operacional ou da linguagem de programação"(LIMA; REIS, 2002, p. 3).

Segundo o próprio site da Microsoft:

> "C# é uma linguagem elegante, orientada a objeto e fortemente tipada, que permite que os desenvolvedores criem uma variedade de aplicativos robustos e seguros executados no .NET Framework. Você pode usar C# para criar aplicativos de cliente do Windows, serviços Web XML, componentes distribuídos, aplicativos cliente-servidor, aplicativos de banco de dados e muito, muito mais"(MICROSOFT, 2015).

A plataforma .NET é composta de várias ferramentas de apoio ao desenvolvimento de aplicações, composta por linguagens de programação, compiladores, modelo de objetos e outros, com a linguagem C# presente dentre as que compõe a plataforma. O C# permite que uma mesma aplicação possa ser executada em diversos dispositivos de hardware, sejam eles PCs, *handhelds* ou qualquer outro dispositivo móvel (LIMA; REIS, 2002, p. 3-4). A documentação

completa da linguagem esta disponível no site da Microsoft[9].

2.7.2 SQLite

No desenvolvimento do jogo digital Laça Palavras, e posteriormente Grapphia, foi utilizado o sistema de gerenciamento de banco de dados (SGBD) relacional SQLite. Sua utilização surgiu da necessidade de armazenamento dos dados relacionados ao usuário, banco de palavras, entre outras informações, possibilitando o salvamento do estado de jogo dos usuários, permitindo ao jogador parar o jogo e continuar em outro momento de onde parou na sua última utilização, e também possibilitando o funcionamento de algumas funcionalidades internas da aplicação.

Segundo próprio site do SQLite (2016), o SQLite é uma biblioteca que implementa um banco de dados relacional SQL embutido à aplicação. Ele não necessita de nenhuma configuração inicial, para sua utilização no Android "apenas é necessário especificar a instrução SQL para gerar o banco de dados e ele é criado automaticamente" (COMACHIO, 2011, p. 25).

O SQLite é um software *open source*, multiplataforma, e possibilita o acesso a base de dados sem execução de um processo SGBD (Sistemas de Gestão de Base de Dados) em paralelo, bastante recomendável para softwares destinados a dispositivos com hardware mais modesto, como o caso de dispositivos móveis. Com isso, devido a escolha de desenvolvimento do jogo digital educacional para dispositivos móveis Android, o SQLite foi selecionado.

2.7.3 Inkscape

A ferramenta Inkscape é um editor de gráficos vetoriais, que possibilita a edição e criação de imagens vetoriais, com código-fonte aberto, semelhante a Adobe Illustrator, Corel Draw, Freehand, entre outros. Álém de ser uma alternativa livre, também é gratuita. Utiliza como extensão nativa o Scalable Vector Graphics (SVG), que é um padrão aberto baseado no XML do consórcio W3C (Consórcio World Wide Web). O W3C é um consórcio onde várias organizações filiais, times e comunidade, se unem buscando a criação de padrões para a Web.

Neste trabalho foi utilizado o Inkscape para criação dos gráficos dos cenários e todos os outros gráficos utilizados.

[9] Documentacão da linguagem C# disponível em: https://docs.microsoft.com/pt-br/dotnet/csharp/

3 METODOLOGIA

O presente capítulo objetiva abordar todos os passos necessários para que a implementação da mecânica de recompensas dentro do jogo fosse possível.

A seção 3.1 apresenta a atual estrutura das bases de dados do Grapphia e as modificações realizadas, com a subseção 3.1.1 apresentando a base de dados SQLite, a subseção 3.1.2 a base de dados remota do Firebase, e a subseção 3.1.3 detalhando as modificações.

A seção 3.2 descreve os *scripts* criados e, por ultimo, a seção 3.3, junto às suas subseções, apresentam as modificações e adições de cenários necessários.

3.1 Estrutura do Banco de Dados

Inicialmente a aplicação do Grapphia utilizava unicamente, como SGBD, o SQLite, uma base de dados relacional local, utilizado para armazenar informações relevantes ao funcionamento da aplicação como um todo. No decorrer da evolução e desenvolvimento da aplicação, foi adicionado o Firebase Realtime Databse, que é uma base de dados não-relacional hospedada em nuvem. Ele facilitou a obtenção dos dados gerados pelos dispositivos com o uso da aplicação, facilitando futuros estudos sobre os dados recuperados. Na realização do trabalho nenhuma alteração foi feita na base de dados do Firebase.

3.1.1 Base de Dados local com o SQLite

A base de dados do SQLite consistia de 3 tabelas: *user* (Tabela 1), *palavraOpcao* (Tabela 2), e *palavraAcertoUser* (Tabela 3). Com a integração dos serviços do Firebase Realtime Database, uma tabela denominada *keyPhone* (Tabela 4) foi inserida na base de dados SQLite para auxiliar seu funcionamento, realizada por Estanislau (2018, p.58).

Para que fosse feita a utilização do SQLite no software Unity, a inclusão da biblioteca *SQLite4Unity3d* foi necessária, e toda a integração do SGBD foi feita pelos autores Leal (2016) e V. C. Santos (2016). A figura 21 mostra o código que estabelece a conexão entre o SQLite e o aplicativo Grapphia, presente no *script telaInicial.cs*

Tabela 1 – Tabela user

user	
Atributo	Tipo
ID (PK)	integer
Name	varchar
Score	integer
Erros	integer
Nivel	integer
scoreDitado	integer
erroDitado	integer

Tabela 2 – Tabela palavraOpcao

palavraOpcao	
Atributo	Tipo
ID (PK)	integer
palavra_completa	varchar
palavra	varchar
opcao1	varchar
opcao2	varchar
nivel	integer
nome_audio_menino	varchar
nome_audio_menina	varchar
nome_audio_ditado	varchar

Tabela 3 – Tabela palavraAcertoUser

palavraAcertoUser	
Atributo	Tipo
ID (PK)	integer
idUser	varchar
idPalavra	bool
acerto	varchar
nivelPalvra	varchar

Tabela 4 – Tabela keyPhone

keyPhone	
Atributo	Tipo
keyFirebase	varchar

3.1.2 Base de Dados remota com o Firebase

Na base de dados remota do Firebase Realtime Database os dados são armazenados em arquivos no formato JSON (Notação de Objetos JavaScript), que é um formato de armazenamento em texto totalmente independente de linguagem, que possui um formato ideal para o armazenamento e troca de dados (JSON, 1999). São salvas na base de dados do Firebase as informações pessoais dos usuários (Figura 22), as palavras gerais de jogo (Figura 23) e número de acertos juntamente com as palavras respondidas corretamente pelo usuário (Figura 24). Toda a integração com o Realtime Database do Firebase foi realizada pelo autor Estanislau (2018).

3.1.3 Inserção e alteração de tabelas na base de dados SQLite

O primeiro passo para tornar possível a implementação da mecânica de recompensas dentro do jogo foi a criação de duas novas tabelas para compor a base de dados local SQLite, adicionando o trecho de código apresentado na figura 25, criando as tabelas: *LivrosLoja* (Tabela 5) e *LivroUser* (Tabela 6).

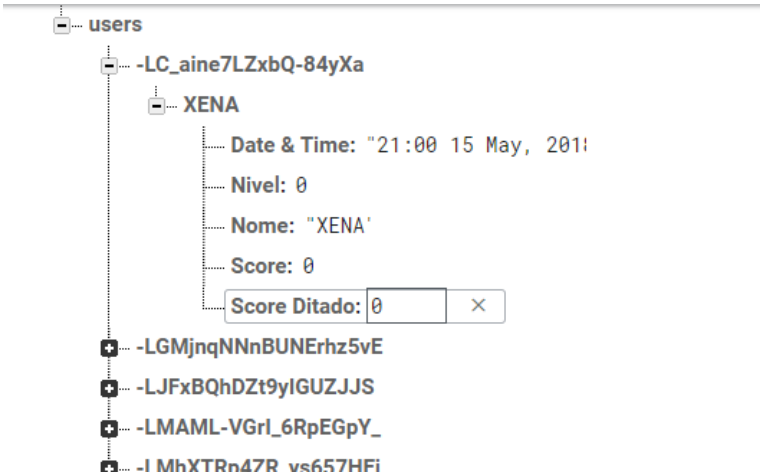
Figura 21 – Conexão com a Base de Dados SQLite e criação das tabelas

```
var filepath = string.Format("{0}/{1}", Application.persistentDataPath, "grapphia");
if (!File.Exists(filepath))
{
    var loadDb = new WWW("jar:file://" + Application.dataPath + "!/assets/" + "grapphia");
    while (!loadDb.isDone) { }
    File.WriteAllBytes(filepath, loadDb.bytes);
}
var dbPath = filepath;
SQLiteConnection _connection = new SQLiteConnection(dbPath, SQLiteOpenFlags.ReadWrite | SQLiteOpenFlags.Create);

_connection.CreateTable<user>();
_connection.CreateTable<palavraOpcao>();
_connection.CreateTable<palavraAcertoUser>();
_connection.CreateTable<keyPhone>();
```

Fonte: Elaborado pelo autor

Figura 22 – Usuários inseridos na base de dados



Fonte: Elaborado pelo autor

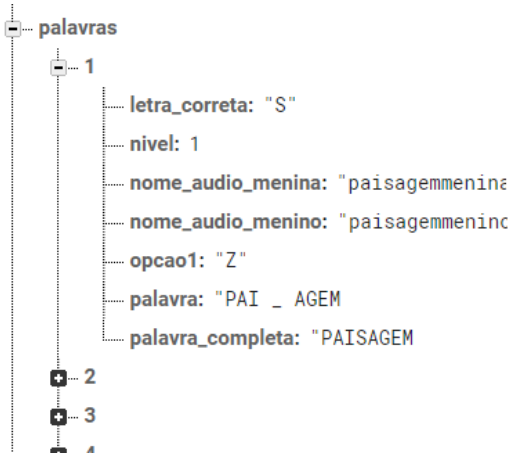
Tabela 5 – Tabela LivrosLoja

LivrosLoja	
Atributo	Tipo
idLivro	integer
titulo	varchar

A tabela *LivrosLoja* foi adicionada com o intuito de armazenar uma identificação para os livros, facilitando a busca das suas informações em tempo de execução. Ao carregar a tela inicial do jogo pela primeira vez, as informações dos livros são salvas na tabela, feita pelo método *salvar_livros_no_banco()*, apresentado na figura 26, e presente no *script storeBooksLogica.cs*. Já a tabela *LivroUser* armazena a relação de livros liberados por usuário, identificando quais livros o mesmo pode realizar a leitura sem necessidade de compra.

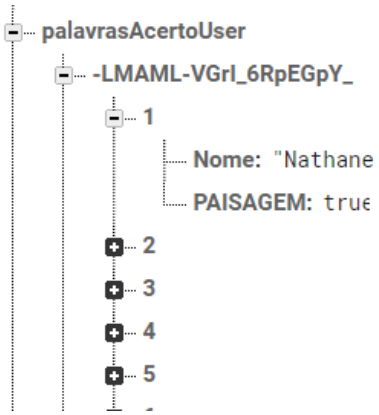
Outro passo foi a alteração da tabela *user* (Tabela 7), adicionando mais dois atributos:

Figura 23 – Palavras inseridos na base de dados



Fonte: Elaborado pelo autor

Figura 24 – Acertos inseridos na base de dados



Fonte: Elaborado pelo autor

Tabela 6 – Tabela LivroUser

LivroUser	
Atributo	Tipo
idUser	integer
idLivro	integer

Coins e *BookPrice*.

O atributo *Coins* inserido na tabela *user* possui o número de moedas que o usuário detém dentro do jogo, e *BookPrice* referencia o preço que é necessário pagar em moedas para comprar algum livro.

Os livros não possuem um valor fixo na tabela que armazena suas informações (*LivrosLoja*), para cada um, seu valor depende do número de livros que o usuário já liberou, sendo incrementado a cada nova compra. Por isso a escolha de colocar o atributo na tabela *User* como referência, com isso o valor do livro sempre será o do usuário corrente.

Figura 25 – Criando as novas tabelas na base de dados SQLite

```
_connection.CreateTable<LivrosLoja>();
_connection.CreateTable<LivroUser>();
```

Fonte: Elaborado pelo autor

Figura 26 – Inserindo informações dos livros na base de dados SQLite

```
public void salvar_livros_no_banco()
{
    livrosLojaDados = new LivrosLoja[total_livros_loja];

    for (int i = 0; i < total_livros_loja; ++i)
        livrosLojaDados[i] = new LivrosLoja();

    livrosLojaDados[0].idLivro = 1;
    livrosLojaDados[0].titulo = "A Fazenda";
    livrosLojaDados[1].idLivro = 2;
    livrosLojaDados[1].titulo = "O Zoológico";
    livrosLojaDados[2].idLivro = 3;
    livrosLojaDados[2].titulo = "Sol de Verão";
    livrosLojaDados[3].idLivro = 4;
    livrosLojaDados[3].titulo = "Teste Comp.";
    livrosLojaDados[4].idLivro = 5;
    livrosLojaDados[4].titulo = "Teste Saldo";

    for (int i = 0; i < total_livros_loja; ++i)
    {
        _connection.Insert (livrosLojaDados[i]);
    }
}
```

Fonte: Elaborado pelo autor

3.2 *Scripts*

Para a implementação da lógica de funcionamento da mecânica de recompensas foi necessário adicionar um novo *script*, o *storeBooksLogica.cs*, que é apresentado no Apêndice A. Ele detêm toda a estrutura das novas tabelas criadas na base de dados do SQLite, e a maioria dos métodos citados no decorrer deste trabalho. Também existiu a necessidade de alteração de alguns *script* dentre os existentes.

3.3 Cenários

No Unity os cenários são as interfaces de interação do usuário com o jogo. Neste trabalho houve alteração de 2 cenários entre os já existentes e a adição de 6 novos. Os modificados foram apenas: *telaEstante* e *telaJogo1*. Os cenários criados foram:

Tabela 7 – Tabela user alterada

user	
Atributo	Tipo
ID (PK)	integer
Name	varchar
Score	integer
Erros	integer
Nivel	integer
scoreDitado	integer
erroDitado	integer
Coins	integer
BookPrice	integer

- *telaStoreBooks*;
- *telaLeituraLivro1*;
- *telaLeituraLivro2*;
- *telaLeituraLivro3*;
- *telaLeituraLivro4*;
- *telaLeituraLivro5*.

3.3.1 Alteração do cenário *telaEstante*

No cenário *telaEstante*, apresentado na figura 27, foi inserido na parte superior, demarcada pelo retângulo de coloração preta, o número de moedas que o usuário possui.

No retângulo mais ao centro foi inserido um botão, representado por um *post-it* com um carrinho de compras contendo livros que, ao ser pressionado, carrega o cenário *telaStoreBooks*. Vinculado ao botão central temos o método *loadSceneStoreBooks()* (Figura 28), localizado no *script storeBooksLogica()*, que verifica a existência de informações sobre os livros liberados para o usuário corrente na tabela da base de dados local *LivrosUser*. Caso o resultado da busca seja menor do que 1, o método *identifica_livros_liberados_no_banco()* (Figura 29) é chamado, também presente no *script*. Este método insere na tabela *LivrosUser* a relação contendo o ID do usuário e ID do livro, identificando os livros que são liberados para leitura do usuário desde o seu primeiro uso, que consistem em três: *A Fazenda*, *O Zoológico* e *Sol de Verão*.

Os livros liberados inicialmente são os mesmos encontrados no cenário *telaEstante* (Figura 27), que remetem também aos modos de jogo implementados.

Figura 27 – Cenário *telaEstante* após a modificação

Fonte: Elaborado pelo autor

Figura 28 – Carregamento da Biblioteca (*telaStoreBooks*) de compra e leitura de Livros

```
public void loadSceneStoreBooks(){
    var totalRelacaoesInseridas = _connection.Table<LivroUser>().Where(x => x.idUser == dadosJogo.Instance.currentUser.Id);
    Debug.Log ("\nTotal de Relações user livros no BD: " + totalRelacaoesInseridas.Count());
    Debug.Log ("\nUser id BD: " + dadosJogo.Instance.currentUser.Id);
    if(totalRelacaoesInseridas.Count()<1){
        BancoLivros.Instance.identifica_livros_liberados_no_banco();
    };
    SceneManager.LoadScene ("telaStoreBooks");
}
```

Fonte: Elaborado pelo autor

3.3.2 Alteração do cenário *telaJogo1*

No cenário de Jogo 1 (*telaJogo1*), demonstrado na figura 30, foi adicionado na parte superior, demarcada pelo retângulo de coloração preta, o número de moedas que o usuário possui. A cada acerto dentro de jogo é incrementado o seu número de moedas, definido o valor de 10 moedas por incremento, como demostra o trecho de código na figura 31, presente no *script gameController.cs*.

3.3.3 Criação do cenário *telaStoreBooks*

O cenário *telaStoreBooks* (Figura 32) foi adicionado para servir de interface de interação totalmente focada na mecânica de recompensas, e consiste em uma biblioteca da qual é possível selecionar os livros para leitura ou compra (liberação).

Na figura 33, em seu canto superior, é apresentado o número de moedas que o usuário detém, ao lado de um *post-it* com uma interrogação que, ao ser clicado, mostra um quadro de instruções.

O cenário também é composto por uma estante com alguns livros que, ao serem pressionados, executam o método *pressionaLivroEstanteStoreBooks(int codigoLivro)*. Este método está no *script*, *storeBooksLogica.cs*, apresentado pelo código 1. O método apresenta um

Figura 29 – Identifica livros liberados para o usuário na base de dados

```
public void identifica_livros_liberados_no_banco()
{
    livrosUserRelacao = new LivroUser[total_livros_estante];

    for (int i = 0; i < total_livros_estante; ++i)
        livrosUserRelacao[i] = new LivroUser();

    livrosUserRelacao[0].idLivro = 1;
    livrosUserRelacao[0].idUser = dadosJogo.Instance.currentUser.Id;
    livrosUserRelacao[1].idLivro = 2;
    livrosUserRelacao[1].idUser = dadosJogo.Instance.currentUser.Id;
    livrosUserRelacao[2].idLivro = 3;
    livrosUserRelacao[2].idUser = dadosJogo.Instance.currentUser.Id;

    for (int i = 0; i < total_livros_estante; ++i)
    {
        _connection.Insert (livrosUserRelacao[i]);
    }
}
```

Fonte: Elaborado pelo autor

Figura 30 – Cenário *telaJogo1* após a modificação

Fonte: Elaborado pelo autor

Figura 31 – Incremento das moedas

```
++dadosJogo.Instance.currentUser.Score;
dadosJogo.Instance.currentUser.Coins+=10;
```

Fonte: Elaborado pelo autor

quadro (quadroLoja) que informa ao usuário se o livro está liberado ou não. O método recebe como parâmetro de entrada o código do livro selecionado pelo usuário, que será utilizado para recuperar os dados do livro na tabela *LivrosLoja*. Esta tabela é uma das novidades inseridas na base de dados SQLite durante a execução dos mecanismos de recompensa.

A partir das informações recuperadas na base de dados, verifica-se a existência da relação do livro pressionado e o usuário corrente na tabela *LivroUser*, que identifica se o livro

Figura 32 – Cenário *telaStoreBooks*

Fonte: Elaborado pelo autor

Figura 33 – Quadro de instruções *telaStoreBooks*

Fonte: Elaborado pelo autor

está liberado ou não para o usuário em questão, ativando e desativando os elementos que compõe o quadro. Se o livro estiver liberado é mostrado o botão de leitura, caso contrário é mostrado o valor do livro e um botão ser pressionado e realizar a sua compra.

Código 1 – Método *pressionaLivroEstanteStoreBooks(int codigoLivro)*

```
1 public void pressionaLivroEstanteStoreBooks(int codigoLivro)
2 {
3 __toStringPrecoLivros();
4 __if (true) {
5 ____quadroLoja.SetActive (true);
6 ____var buscaTituloLivroBanco = _connection.Table<LivrosLoja>().Where(x
      => x.idLivro == codigoLivro);
7 ____LivrosLoja tituloLivroBanco = new LivrosLoja();
8 ____foreach (var t in buscaTituloLivroBanco)
9 ____{
10 ______tituloLivroBanco.titulo=t.titulo;
11 ____}
12 ____tituloLivro.text =""+ tituloLivroBanco.titulo;
13 ____var verificaExistenciaDeRelacao = _connection.Table<LivroUser>().
      Where(x => x.idUser == dadosJogo.Instance.currentUser.Id && x.idLivro
       == codigoLivro);
14 ____if(verificaExistenciaDeRelacao.Count()<1){
15 ______desabilitaLeituraDosLivros();
16 ______if(dadosJogo.Instance.currentUser.Coins<dadosJogo.Instance.
      currentUser.BookPrice){
17 ________desabilitaCompraDosLivros();
18 ________btn_comprar_bloqueado.SetActive(true);
19 ______}else{
20 ________habilitaCompraDoLivro(codigoLivro);
21 ________btn_comprar_bloqueado.SetActive(false);
22 ______}
23 ______valorModedasQuadroLoja.SetActive(true);
24 ______desenhoMoedasQuadroLoja.SetActive(true);
25 ____}else{
26 ______habilitaLeituraDoLivro(codigoLivro);
27 ______desabilitaCompraDosLivros();
28 ______btn_comprar_bloqueado.SetActive(false);
29 ______valorModedasQuadroLoja.SetActive(false);
30 ______desenhoMoedasQuadroLoja.SetActive(false);
31 ____}
32 ____if(codigoLivro==1){
33 ______capaLivro1.SetActive(true);
34 ______capaLivro2.SetActive(false);
35 ______capaLivro3.SetActive(false);
36 ______capaLivro4.SetActive(false);
37 ______capaLivro5.SetActive(false);
38 ____}else if(codigoLivro==2){
39 ______capaLivro2.SetActive(true);
40 ______capaLivro1.SetActive(false);
41 ______capaLivro3.SetActive(false);
```

```
42 ····capaLivro4.SetActive(false);
43 ····capaLivro5.SetActive(false);
44 ··}else if(codigoLivro==3){
45 ····capaLivro3.SetActive(true);
46 ····capaLivro1.SetActive(false);
47 ····capaLivro2.SetActive(false);
48 ····capaLivro4.SetActive(false);
49 ····capaLivro5.SetActive(false);
50 ··}else if(codigoLivro==4){
51 ····capaLivro4.SetActive(true);
52 ····capaLivro1.SetActive(false);
53 ····capaLivro2.SetActive(false);
54 ····capaLivro3.SetActive(false);
55 ····capaLivro5.SetActive(false);
56 ··}else{
57 ····capaLivro5.SetActive(true);
58 ····capaLivro1.SetActive(false);
59 ····capaLivro2.SetActive(false);
60 ····capaLivro3.SetActive(false);
61 ····capaLivro4.SetActive(false);
62 ··}
63 }
64 }
```

A figura 34 apresenta o *quadroLoja*, que é o quadro de informações relacionados ao livro quando pressionado e, no caso exemplificado, o livro está liberado para a leitura. Nele temos a apresentação dos elementos de título e capa do livro, um botão de voltar e o botão de ler. Ao pressionar o botão *LER*, o método *btnClick_Ler(int codigoLivro)* (Figura 35) é chamado, presente no *script storeBooksLogica.cs*, e o cenário de leitura do livro é carregado por ele. Este método utiliza o código do livro recebido como parâmetro para definir o cenário de leitura, utilizando o método público *SceneManager.LoadScene()*, que recebe o nome do cenário como parâmetro para carregá-lo.

Já a figura 36 apresenta o *quadroLoja* na situação contrária, quando o livro não está liberado para a leitura. Nele temos os mesmos elementos base do quadro para a leitura, com apenas um adicional, que é o custo em moedas para realizar a compra do livro, e o botão de compra. O botão de compra não é apresentado caso o número de moedas do usuário (*dadosJogo.Instance.currentUser.Coins*) for menor que o valor de compra do livro (*dadosJogo.Instance.currentUser.BookPrice*), demostrado no código 1, apresentado em seu lugar o botão *SEM SALDO*, que não pode ser pressionado, mostrado na figura 37.

Ao pressionar o botão *COMPRAR*, o método *btnClick_Comprar (int codigoLivro)* é chamado (Figura 38), presente no *script storeBooksLogica.cs*, que carrega o método que faz a liberação do livro e realiza algumas alterações nos elementos mostrados no quadro. O método *libera_livro_no_banco_para_o_usuario(int codigoLivro)* faz essa liberação. Ele insere na tabela *LivroUser* a relação de ID do usuário e ID do livro, subtrai o número de moedas gastas na compra

Figura 34 – Quadro montado para leitura

Fonte: Elaborado pelo autor

Figura 35 – Método *btnClick_Ler(int codigoLivro)*

```
public void btnClick_Ler(int codigoLivro)
{
    if(codigoLivro == 1)
        SceneManager.LoadScene ("telaLeituraLivro1");
    else if(codigoLivro == 2)
        SceneManager.LoadScene ("telaLeituraLivro2");
    else if(codigoLivro == 3)
        SceneManager.LoadScene ("telaLeituraLivro3");
    else if(codigoLivro == 4)
        SceneManager.LoadScene ("telaLeituraLivro4");
    else
        SceneManager.LoadScene ("telaLeituraLivro5");
}
```

Fonte: Elaborado pelo autor

do livro do usuário (*Coins*), e aumenta o valor de compra dos livros em 200 (*BookPrice*). A cada compra, os livros ficam 200 moedas mais caro. O método é apresentado na figura 39, também presente no *script*.

3.3.4 Criação dos cenários de leitura

Como mostrado na subseção anterior, ao ser clicado o botão de leitura do quadro de informações dos livros, é carregada a cena de leitura referente ao livro selecionado. Foram adicionados 5 destes cenários de leitura, um para cada livro. Dentre estes livros 3 são os livros base dos níveis de jogo, e os outros dois, adicionados para realizar os testes de funcionamento da mecânica de recompensas, e futuramente dar espaço a novos livros, podendo ser incrementados mais cenários. A figura 40 mostra a cena de leitura do livro *A Fazenda*, onde temos a capa do livro e um botão para retornar ao cenário referente a biblioteca de livros (*telaStoreBooks*). A

Figura 36 – Quadro montado para a compra

Fonte: Elaborado pelo autor

Figura 37 – Quadro montado para a compra com saldo insuficiente

Fonte: Elaborado pelo autor

animação do livro funciona segundo a interação real de uma pessoa realizando a leitura de um livro, e ao pressionar o canto inferior direito é possível realizar a passagem de páginas até o final da história. Todos os cenários de leitura seguem essa base de funcionamento e organização.

Figura 38 – Método *btnClick_Comprar (int codigoLivro)*

```
public void btnClick_Comprar (int codigoLivro)
{
    BancoLivros.Instance.libera_livro_no_banco_para_o_usuario(codigoLivro);
    desabilitaCompraDosLivros();
    habilitaLeituraDoLivro(codigoLivro);
    btn_comprar_bloqueado.SetActive(false);
    valorModedasQuadroLoja.SetActive(false);
    desenhoMoedasQuadroLoja.SetActive(false);
}
```

Fonte: Elaborado pelo autor

Figura 39 – Método *libera_livro_no_banco_para_o_usuario(int codigoLivro)*

```
public void libera_livro_no_banco_para_o_usuario(int codigoLivro){
    dadosJogo.Instance.currentUser.Coins = (dadosJogo.Instance.currentUser.Coins) - (dadosJogo.Instance.currentUser.BookPrice);
    dadosJogo.Instance.currentUser.BookPrice += 200;
    _connection.Update(dadosJogo.Instance.currentUser);
    relacaoLivroUser = new LivroUser();
    relacaoLivroUser.idLivro = codigoLivro;
    relacaoLivroUser.idUser =  dadosJogo.Instance.currentUser.Id;
    _connection.Insert (relacaoLivroUser);
}
```

Fonte: Elaborado pelo autor

Figura 40 – Cenário *telaLeituraLivro1*

Fonte: Elaborado pelo autor

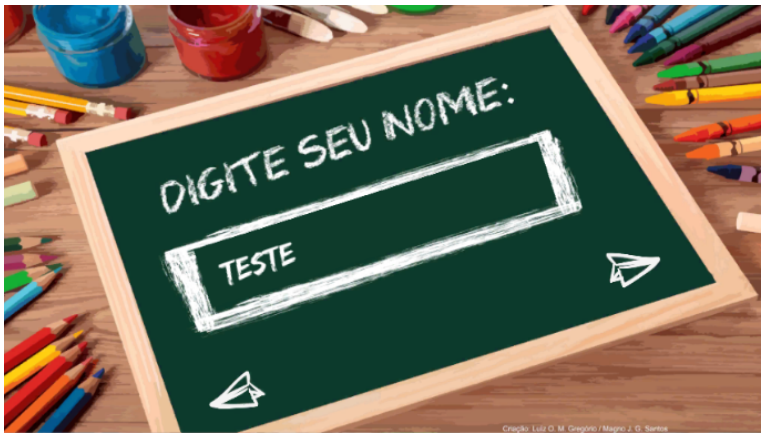
4 TESTES E RESULTADOS

O presente capítulo tem como objetivo mostrar o funcionamento da mecânica de recompensas através de testes de execução, demostrando as inserções necessárias na base de dados, e resposta dos cenários. Na seção 4.1 é detalhada a criação dos usuários e das tabelas na base de dados. As seções 4.2 e 4.3 mostram as informações e momento de inserção nas tabelas *LivrosLoja* e *LivroUser*, respectivamente. A inserção da relação de liberação de livros para o usuário, e modificação do número de moedas e valor de compra do livro, são apresentadas na seção 4.4. Por ultimo, o processo de ganho de moedas, com as alterações também na base de dados, está presente na seção 4.5.

4.1 Criação das tabelas *LivrosLoja* e *LivroUser*

Ao executar a aplicação pela primeira vez, no mesmo momento onde é criado o usuário, apresentado na figura 41, são criadas a base de dados e as suas tabelas, apresentadas na figura 42. Dentre as tabelas temos *LivrosLoja* e *LivroUser*, inseridas exclusivamente para auxiliar no funcionamento da mecânica de recompensas.

Figura 41 – Tela de criação de usuário

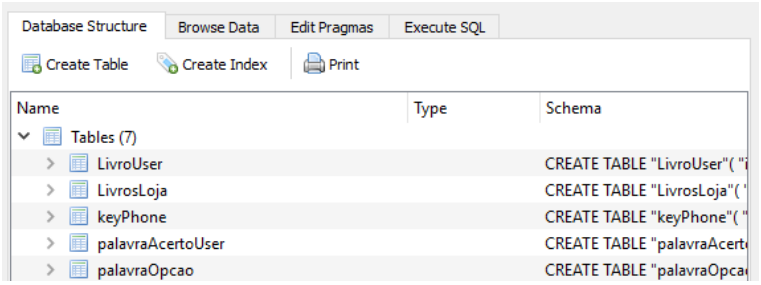


Fonte: Elaborado pelo autor

4.2 Inserção das informações dos livros a base de dados SQLite

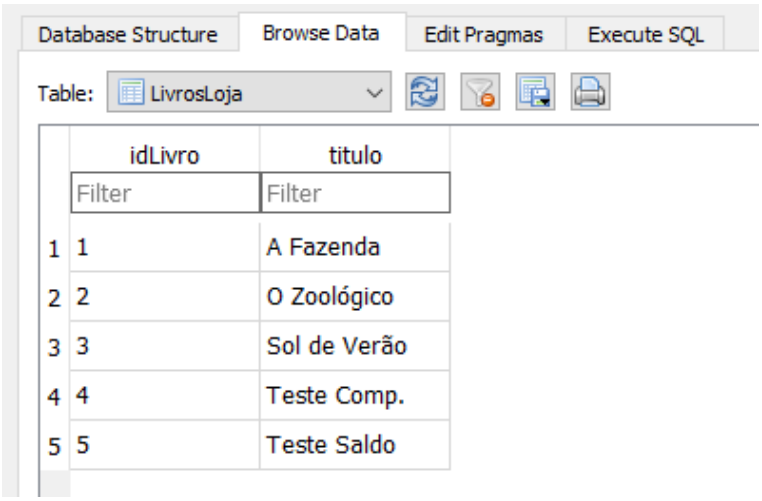
Após a criação do usuário, também quando é executada a aplicação pela primeira vez, as informações dos livros são inseridas a base de dados, como mostra a figura 43. Na tabela foi inserido o código de identificação do livro e seu título, que servem de referência ao carregar suas telas de leitura, quadro de informações e liberação.

Figura 42 – Tabelas da Base de Dados SQlite



Fonte: Elaborado pelo autor

Figura 43 – Dados da tabela *LivrosLoja*



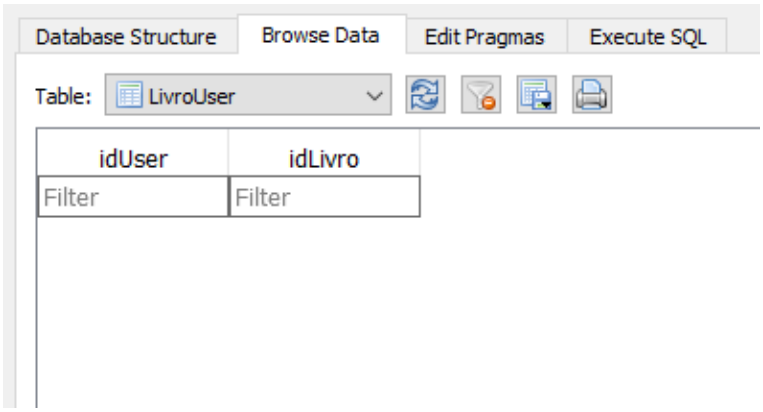
Fonte: Elaborado pelo autor

4.3 Liberação dos livros iniciais

Quando criado o usuário, o jogador é transferido para a tela de seleção de nível de jogo. Neste ponto ainda não temos nenhuma informação presente na tabela *LivroUser* (Figura 44). A tabela é preenchida apenas quando é carregada a tela que possibilita a leitura e compra dos livros, apresentada na figura 45.

Inicialmente apenas três livros são liberados: *A Fazenda*, *O Zoológico* e *Sol de Verão*, referentes também aos níveis de jogo existentes. A figura 46 mostra a tabela *LivroUser* já com as informações destes livros liberados inicialmente. O *idUser* possui o *Id* do usuário corrente (1), e o *idLivro* possui os códigos dos livros liberados. O código de livro 1 faz referência ao livro a *A Fazenda*, 2 ao *O Zoológico* e 3 ao *Sol de Verão*. A liberação destes três livros acontece na primeira vez que o usuário é levado a tela da loja (Figura 45).

Figura 44 – Tabela *LivroUser* vazia



Fonte: Elaborado pelo autor

Figura 45 – Tela da loja de livros

Fonte: Elaborado pelo autor

4.4 Compra de Livro

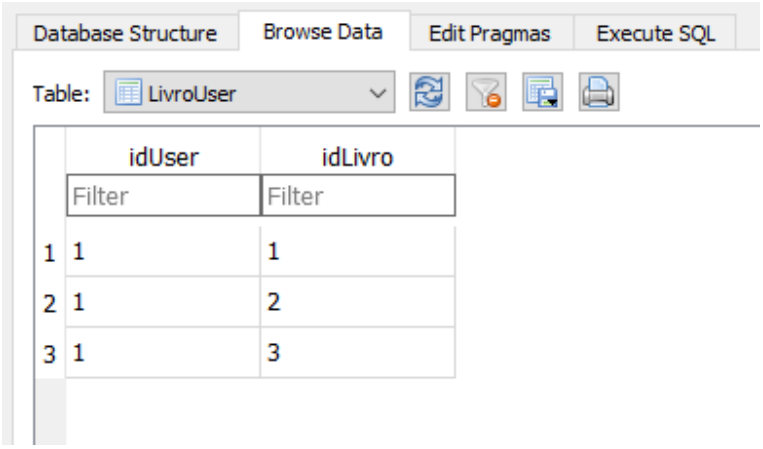
Na tabela *LivrosLoja* (Figura 43), além dos livros iniciais, foram inseridos mais dois para testes. Um auxiliando na demostração do funcionamento da compra e liberação do livro, e outro para testar o bloqueio da compra na falta de saldo.

O livro de teste da compra possui o código de identificação igual a *4*, e título *Teste Comp*. Já o livro de bloqueio tem código *5*, e título *Teste Saldo*.

No cenário referente a loja de livros (Figura 45), ao selecionar o livro de capa azul, apresenta-se o quadro de informações deste livro em específico. No exemplo apresentado na figura 47, o livro não está liberado para leitura, podendo o usuário optar pela compra, uma vez que, ele tem saldo suficiente para efetuá-la. Assim, ao acionar o botão comprar, o quadro é alterado e o botão "Ler"é disponibilizado, como mostra a figura 48.

Ao efetuar a compra de um livro, a tabela *LivroUser* irá receber uma nova tupla indicando o código do usuário corrente (código 1) e o código do livro adquirido (código 4). A figura 49 apresenta a base de dados já com a atualização da nova tupla.

Figura 46 – Dados da tabela *LivroUser*



Fonte: Elaborado pelo autor

Figura 47 – Quadro de informações para teste da compra

Fonte: Elaborado pelo autor

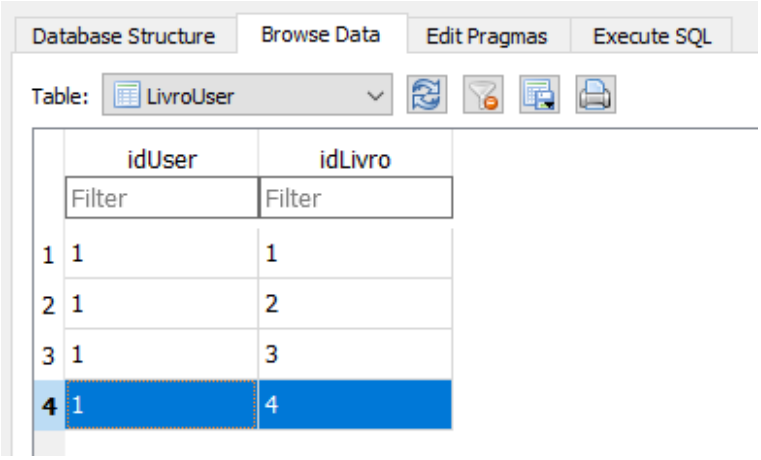
Após a compra, o número de moedas do usuário foi decrementando, passando a ser 0 (*Coins*), e o o valor da compra de livro acrescido de 200 moedas, passando a assumir o valor 300 (*BookPrice*), como mostra a figura 50.

Voltando a tela da loja, ao clicar no livro superior a direita, com capa de coloração verde, é apresentado quadro de informações montado para a compra. Apenas uma modificação é apresentada, que é a substituição do botão de compra pelo botão sem saldo, logo que o número de moedas do usuário (0) é insuficiente para sua liberação, como mostrado na figura 51.

Figura 48 – Quadro de informações após teste da compra

Fonte: Elaborado pelo autor

Figura 49 – Tabela *LivroUser* após a compra



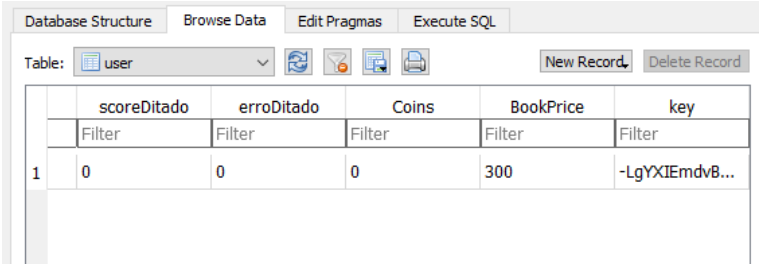
Fonte: Elaborado pelo autor

4.5 Ganho de moedas

Durante o jogo, sempre que o usuário realizar a seleção correta da letra que completa a palavra (Figura 52), é apresentado um quadro em amarelo, com a indicação de acerto, então uma parte do quebra-cabeça que compõe uma casa é adicionado, e o número de moedas é acrescido em 10 (dez). A figura 53 apresenta a tela de jogo do nível 1 neste momento de acerto.

No fim do teste realizado o número de moedas foi alterado para 120, cantando com um total de 12 acertos. O quantitativo de moedas é então armazenado na base de dados, mas especificamente na tabela *User*, conforme mostra a figura 54.

Figura 50 – Tabela *user* após a compra



Fonte: Elaborado pelo autor

Figura 51 – Quadro para teste do saldo

Fonte: Elaborado pelo autor

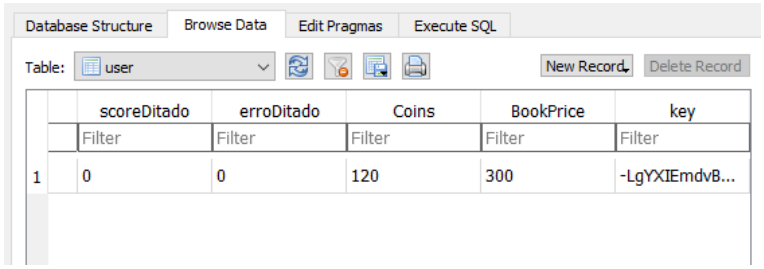
Figura 52 – Tela de jogo do nível 1 com saldo zero

Fonte: Elaborado pelo autor

Figura 53 – Tela de jogo do nível 1 após jogar

Fonte: Elaborado pelo autor

Figura 54 – Tabela *user* ao fim de jogo



Fonte: Elaborado pelo autor

5 CONCLUSÕES E TRABALHOS FUTUROS

A incrementação de atividades lúdicas no processo de aprendizagem torna-se cada vez mais presente no cotidiano, seja dentro ou fora do âmbito educacional. O Grapphia proporciona ajuda a crianças na aprendizagem correta da ortografia, referindo diretamente ao grupo de palavras irregulares, que possuam letras ou dígrafos concorrentes, e apresentem o mesmo som, em um mesmo contexto, e que não existam regularidades de escrita. Assim, tornar este processo de aprendizado mais divertido e engajador está diretamente relacionado com o tempo de jogo e atenção que o usuário pode dar a essa atividade, tornando a inclusão de uma mecânica de recompensa um requisito necessário para aprimorar o aplicativo em questão. Este mecanismo pode proporcionar maior tempo de aprendizagem e iteração entre jogo e o jogador.

Considerando o objetivo proposto, conclui-se que a implementação da mecânica de recompensas dentro do jogo Grapphia foi realizada com sucesso.

Como trabalho futuro, o jogo Grapphia pode ser incrementado com novos livros a serem comprados dentro do jogo, substituindo inicialmente os de teste. Além disso, os livros na estante que definem os grupos de palavras tratadas no aplicativo, podem ser liberados após a compra, utilizando as moedas implementadas neste trabalho. Outro ponto seria a realização de um estudo sobre o tempo de jogo dos usuários utilizando as versões com e sem a mecânica de recompensas implementada, ou até mesmo para balancear os valores de recompensas e custo dos livros.

BIBLIOGRAFIA

ALVARENGA, D. Análise de variações ortográficas. **Revista Presença Pedagógica**, 1995.

AMATE, Flávio Cezar. **Desenvolvimento de Jogos Computadorizados para Auxiliar a Aquisição da Base Alfabética de Crianças**. 2007. Tese (Doutorado) – Universidade de São Paulo - Escola de Engenharia de São Carlos.

ASSIS, Luciana et al. Grapphia: Aplicativo para Dispositivos Móveis para Auxiliar no Ensino da Ortografia. **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**, v. 6, n. 1, 2017.

BATTAIOLA, André Luiz. Jogos por Computador – Histórico, Relevância Tecnológica e Mercadológica, Tendências e Técnicas de Implementação, 2000.

CANALTECH. **O que é open source?** 2018. Disponível em: <https://canaltech.com.br/produtos/O-que-e-open-source/>. Acesso em:

COMACHIO, VANDERSON. **FUNCIONAMENTO DE BANCO DE DADOS EM ANDROID: UM ESTUDO EXPERIMENTAL UTILIZANDO SQLITE**. Trabalho de Diplomação, 2011.

ESTANISLAU, Pedro Henrique Cerqueira. **GRAPPHIA**: INTEGRANDO APLICATIVO COM O REALTIME DATABASE DA PLATAFORMA FIREBASE. Trabalho de Conclusão de Curso, 2018.

GHOZLAND, David. Designing for Motivation. **Gamasutra**, 2010. Disponível em: <https://www.gamasutra.com/view/feature/129852/designing_for_motivation.php?page=5>.

GROS, Begoña. The impact of digital games in education. **First Monday**, v. 8, n. 7, 2003. DOI: http://www.firstmonday.org/issues/issue8_7/xyzgros/index.html.

JSON. **Introdução ao JSON**. 1999. Disponível em: <https://www.json.org/json-pt.html>.

KUANG, Alex. **Dynamic Difficulty Adjustment**. 2012. Tese (Doutorado) – Worcester Polytechnic Institute.

LEAL, Ademir Santos. **Aplicação do método de aprendizagem por reforço Q-Learning na adaptatividade dinâmica de dificuldade de um jogo digital ortográfico**. Trabalho de Conclusão de Curso, 2016.

LIMA, E.; REIS, E. C# e .net – guia do desenvolvedor. Editora Campus Ltda, Rio de Janeiro, 2002.

LUCCHESE, Fabiano; RIBEIRO, Bruno. Conceituação de Jogos Digitais, 2009.

MATLIN, M. W. **Psicologia cognitiva**. LTC, 2004. ISBN 9788521613923. Disponível em: <https://books.google.com.br/books?id=FzGvAwAACAAJ>.

MICROSOFT. **Introdução à linguagem C# e ao .NET Framework**. 2015. Disponível em: <https://docs.microsoft.com/pt-br/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework>. Acesso em:

NAYAK, A.; PORIY, A.; POOJARY, D. Type of nosql databases and its comparison with relational databases. **International Journal of Applied Information Systems**, v. 5, n. 4, 2013.

SAKUDA, L. O. et al. Análise do Mercado Brasileiro de Jogos Digitais. In: SAKUDA, L. O.; FORTIM, I.; I., (Org) (Ed.). **2 Censo da Industria Brasileira de Jogos Digitais**. Brasília: Ministério da Cultura, 2018.

SANTOS, Magno Juliano Gonçalves. **DESENVOLVIMENTO DE JOGO EDUCACIONAL PARA DISPOSITIVOS MÓVEIS PARA AUXILIAR O PROCESSO DE ENSINO/APRENDIZAGEM DA ORTOGRAFIA UTILIZANDO A FERRAMENTA *UNITY***. Trabalho de Conclusão de Curso, 2019.

SANTOS, Vinícius Cordeiro. **Aplicação do algoritmo SARSA no balanceamento dinâmico de dificuldade de um jogo digital ortográfico**. Trabalho de Conclusão de Curso, 2016.

SAVI, Rafael; ULBRICHT, Vania Ribas. JOGOS DIGITAIS EDUCACIONAIS: BENEFÍCIOS E DESAFIOS. **CINTED-UFRGS**, 2008. DOI: https://seer.ufrgs.br/renote/article/view/14405/8310.

SCHUYTEMA, P. Design de games: uma abordagem prática. **São Paulo: Cengage Learning**, 2008.

SENA, Alexandre; COELHO, Dennis Kerr. Motivação dos Jogadores de Videogame – Uma breve visão sobre as Técnicas de Engajamento. **SBGames**, 2012.

SILVA, CLÁUDIA LUCIENE DE MELO. **LÚDICO E APRENDIZAGENS: CONCILIANDO JOGOS TRADICIONAIS E DIGITAIS**. 2006. Diss. (Mestrado) – Universidade Federal da Paraíba.

SILVA, Maycon Prado Rocha et al. **Jogos Digitais: definições, classificações e avaliação**. Tópicos em Engenharia de Computação, 2009.

SQLITE. **SQLIte is Serveless**. 2017. Disponível em: <https://www.sqlite.org/serverless.html>. Acesso em:

SQLITE. **About SQLite**. 2016. Disponível em: <https://www.sqlite.org/about.html>. Acesso em:

TAROUCO, Liane Margarida Rockenbach et al. Jogos educacionais. **CINTED-UFRGS**, 2004. DOI: http://www.cinted.ufrgs.br/ciclo3/af/30-jogoseducacionais.pdf.

UNITY. https://unity.com/, 2017.

UNITY. **Game engines—how do they work?** 2018. Disponível em: <https://unity3d.com/what-is-a-game-engine>. Acesso em:

A SCRIPT STOREBOOKSLOGICA.CS COMPLETO

```
1 using UnityEngine;
2 using UnityEngine.UI;
3 using System.Collections;
4 using System.Collections.Generic;
5 using System.IO;
6 using System;
7 using SQLite4Unity3d;
8 using UnityEngine.SceneManagement;

10 // Classe Singleton onde armazena os Livros da Loja!
11 public class BancoLivros
12 {
13     private static BancoLivros instance;
14     public LivrosLoja[] livrosLojaDados;
15 ··public LivroUser[] livrosUserRelacao;
16 ··public LivroUser relacaoLivroUser;
17 ··public SQLiteConnection _connection;
18 ··private int total_livros_loja = 5;
19 ··private int total_livros_estante = 3;
20 ··public int total_livros_geral;



23     private BancoLivros()
24     {

26     }

28 ··public static BancoLivros Instance
29     {
30 ····get
31         {
32             if (instance == null)
33             {
34 ··········instance = new BancoLivros();
35             }
36             return instance;
37         }

39     }
40     public void salvar_livros_no_banco()
41     {
42 ····livrosLojaDados = new LivrosLoja[total_livros_loja];

44 ····for (int i = 0; i < total_livros_loja; ++i)
45 ····    livrosLojaDados[i] = new LivrosLoja();
```

```
47 ____livrosLojaDados[0].idLivro = 1;
48 ____livrosLojaDados[0].titulo = "A Fazenda";
49 ____livrosLojaDados[1].idLivro = 2;
50 ____livrosLojaDados[1].titulo = "O Zool gico";
51 ____livrosLojaDados[2].idLivro = 3;
52 ____livrosLojaDados[2].titulo = "Sol de Ver o";
53 ____livrosLojaDados[3].idLivro = 4;
54 ____livrosLojaDados[3].titulo = "Teste Comp.";
55 ____livrosLojaDados[4].idLivro = 5;
56 ____livrosLojaDados[4].titulo = "Teste Saldo";

58 ____for (int i = 0; i < total_livros_loja; ++i)
59 ____{
60 ________connection.Insert (livrosLojaDados[i]);
61         }
62     }
63     public void identifica_livros_liberados_no_banco()
64     {
65 ____livrosUserRelacao = new LivroUser[total_livros_estante];

67 ____for (int i = 0; i < total_livros_estante; ++i)
68 ____    livrosUserRelacao[i] = new LivroUser();

70 ____livrosUserRelacao[0].idLivro = 1;
71 ____livrosUserRelacao[0].idUser = dadosJogo.Instance.currentUser.Id;
72 ____livrosUserRelacao[1].idLivro = 2;
73 ____livrosUserRelacao[1].idUser = dadosJogo.Instance.currentUser.Id;
74 ____livrosUserRelacao[2].idLivro = 3;
75 ____livrosUserRelacao[2].idUser = dadosJogo.Instance.currentUser.Id;

77 ____for (int i = 0; i < total_livros_estante; ++i)
78 ____{
79 ________connection.Insert (livrosUserRelacao[i]);
80         }
81     }
82 __public void libera_livro_no_banco_para_o_usuario(int codigoLivro){
83 ____dadosJogo.Instance.currentUser.Coins = dadosJogo.Instance.
      currentUser.Coins - dadosJogo.Instance.currentUser.BookPrice;
84 ____dadosJogo.Instance.currentUser.BookPrice += 200;
85 ____relacaoLivroUser = new LivroUser();
86 ____relacaoLivroUser.idLivro = codigoLivro;
87 ____relacaoLivroUser.idUser =  dadosJogo.Instance.currentUser.Id;
88 ____connection.Insert (relacaoLivroUser);
89 __}
90 }
```

```
92 public class LivrosLoja
93 {
94     [PrimaryKey]
95     public int idLivro { get; set; }
96     [Unique]
97     public string titulo { get; set; }
98 }

100 public class LivroUser
101 {
102 ___public int idUser { get; set; }
103     public int idLivro { get; set; }
104 }

106 public class storeBooksLogica : MonoBehaviour {

108 ___public SQLiteConnection _connection;
109 ___private int total_livros=5;
110 ___public GameObject quadroLoja;
111 ___public GameObject btn_comprar_livro_4;
112 ___public GameObject btn_comprar_livro_5;
113 ___public GameObject btn_comprar_bloqueado;
114 ___public GameObject btn_ler_livro_1;
115 ___public GameObject btn_ler_livro_2;
116 ___public GameObject btn_ler_livro_3;
117 ___public GameObject btn_ler_livro_4;
118 ___public GameObject btn_ler_livro_5;
119 ___public GameObject btn_voltar;
120 ___public Text tituloLivro;
121 ___public GameObject valorModedasQuadroLoja;
122 ___public GameObject desenhoMoedasQuadroLoja;
123 ___public GameObject capaLivro1;
124 ___public GameObject capaLivro2;
125 ___public GameObject capaLivro3;
126 ___public GameObject capaLivro4;
127 ___public GameObject capaLivro5;

129 ___public Text coinsUser;
130     public Text booksPrice;

132     void Start () {
133 ______var filepath = string.Format("{0}/{1}", Application.
      persistentDataPath, "grapphia");
134 ______if (!File.Exists(filepath))
135 ______{
136 _________var loadDb = new WWW("jar:file://" + Application.dataPath + "!/
      assets/" + "grapphia");
```

```
137 ____while (!loadDb.isDone) { }
138 ____File.WriteAllBytes(filepath, loadDb.bytes);
139 __}
140 __var dbPath = filepath;
141 __connection = new SQLiteConnection(dbPath, SQLiteOpenFlags.ReadWrite
        | SQLiteOpenFlags.Create);
142 __Debug.Log("Final PATH: " + dbPath);
143 __//Criando tabela usu rio e tabela op   o!
144 __connection.CreateTable<user>();
145 __connection.CreateTable<palavraOpcao>();
146 __connection.CreateTable<palavraAcertoUser>();
147 __connection.CreateTable<LivrosLoja>();
148         connection.CreateTable<LivroUser>();
149 __toStringMoedas();
150 }

152 void Update () {
153 __toStringMoedas();
154 }

156 public void loadSceneStoreBooks(){
157 __var totalRelacaoesInseridas = connection.Table<LivroUser>().Where(x
        => x.idUser == dadosJogo.Instance.currentUser.Id);
158 __Debug.Log ("\nTotal de Rela  es user livros no BD: " +
      totalRelacaoesInseridas.Count());
159 __Debug.Log ("\nUser id BD: " + dadosJogo.Instance.currentUser.Id);
160 __if(totalRelacaoesInseridas.Count()<1){
161 ____BancoLivros.Instance.identifica_livros_liberados_no_banco();
162 __};
163 __SceneManager.LoadScene ("telaStoreBooks");
164 }

166 public void pressionaLivroEstanteStoreBooks(int codigoLivro)
167 {
168 __toStringPrecoLivros();
169 __if (true) {
170 ____quadroLoja.SetActive (true);
171 ____var buscaTituloLivroBanco = connection.Table<LivrosLoja>().Where(
      x => x.idLivro == codigoLivro);
172 ____LivrosLoja tituloLivroBanco = new LivrosLoja();
173 ____foreach (var t in buscaTituloLivroBanco)
174 ____{
175 ______tituloLivroBanco.titulo=t.titulo;
176 ____}
177 ____tituloLivro.text =""+ tituloLivroBanco.titulo;
178 ____var verificaExistenciaDeRelacao = connection.Table<LivroUser>().
      Where(x => x.idUser == dadosJogo.Instance.currentUser.Id && x.idLivro
```

```
        == codigoLivro);
179 ____if(verificaExistenciaDeRelacao.Count()<1){
180 ____desabilitaLeituraDosLivros();
181 ____if(dadosJogo.Instance.currentUser.Coins<dadosJogo.Instance.
      currentUser.BookPrice){
182 ____desabilitaCompraDosLivros();
183 ____btn_comprar_bloqueado.SetActive(true);
184 ____}else{
185 ____habilitaCompraDoLivro(codigoLivro);
186 ____btn_comprar_bloqueado.SetActive(false);
187 ____}
188 ____valorModedasQuadroLoja.SetActive(true);
189 ____desenhoMoedasQuadroLoja.SetActive(true);
190 ____}else{
191 ____habilitaLeituraDoLivro(codigoLivro);
192 ____desabilitaCompraDosLivros();
193 ____btn_comprar_bloqueado.SetActive(false);
194 ____valorModedasQuadroLoja.SetActive(false);
195 ____desenhoMoedasQuadroLoja.SetActive(false);
196 ____}
197 ____if(codigoLivro==1){
198 ____capaLivro1.SetActive(true);
199 ____capaLivro2.SetActive(false);
200 ____capaLivro3.SetActive(false);
201 ____capaLivro4.SetActive(false);
202 ____capaLivro5.SetActive(false);
203 ____}else if(codigoLivro==2){
204 ____capaLivro2.SetActive(true);
205 ____capaLivro1.SetActive(false);
206 ____capaLivro3.SetActive(false);
207 ____capaLivro4.SetActive(false);
208 ____capaLivro5.SetActive(false);
209 ____}else if(codigoLivro==3){
210 ____capaLivro3.SetActive(true);
211 ____capaLivro1.SetActive(false);
212 ____capaLivro2.SetActive(false);
213 ____capaLivro4.SetActive(false);
214 ____capaLivro5.SetActive(false);
215 ____}else if(codigoLivro==4){
216 ____capaLivro4.SetActive(true);
217 ____capaLivro1.SetActive(false);
218 ____capaLivro2.SetActive(false);
219 ____capaLivro3.SetActive(false);
220 ____capaLivro5.SetActive(false);
221 ____}else{
222 ____capaLivro5.SetActive(true);
223 ____capaLivro1.SetActive(false);
```

```
224 ·····capaLivro2.SetActive(false);
225 ·····capaLivro3.SetActive(false);
226 ·····capaLivro4.SetActive(false);
227 ···}
228 ··}
229 ·}

231 ·public void btnClick_Comprar (int codigoLivro)
232 ·{
233 ···BancoLivros.Instance.libera_livro_no_banco_para_o_usuario(
        codigoLivro);
234 ···desabilitaCompraDosLivros();
235 ···habilitaLeituraDoLivro(codigoLivro);
236 ···btn_comprar_bloqueado.SetActive(false);
237 ···valorModedasQuadroLoja.SetActive(false);
238 ···desenhoMoedasQuadroLoja.SetActive(false);
239 ·}
240 ·public void btnClick_Ler(int codigoLivro)
241 ·{
242 ···if(codigoLivro == 1)
243 ·····SceneManager.LoadScene ("telaLeituraLivro1");
244 ···else if(codigoLivro == 2)
245 ·····SceneManager.LoadScene ("telaLeituraLivro2");
246 ···else if(codigoLivro == 3)
247 ·····SceneManager.LoadScene ("telaLeituraLivro3");
248 ···else if(codigoLivro == 4)
249 ·····SceneManager.LoadScene ("telaLeituraLivro4");
250 ···else
251 ·····SceneManager.LoadScene ("telaLeituraLivro5");
252 ·}

254 ·public void btnClick_Voltar ()
255 ·{
256 ···if (true) {
257 ·····quadroLoja.SetActive (false);
258 ···}
259 ·}

261 ·private void habilitaLeituraDoLivro(int codigoLivro){
262 ···if(codigoLivro==1)
263 ·····btn_ler_livro_1.SetActive (true);
264 ···else if(codigoLivro == 2)
265 ·····btn_ler_livro_2.SetActive (true);
266 ···else if(codigoLivro == 3)
267 ·····btn_ler_livro_3.SetActive (true);
268 ···else if(codigoLivro == 4)
269 ·····btn_ler_livro_4.SetActive (true);
```

```
270 ____else
271 ________btn_ler_livro_5.SetActive (true);
272 ____}

274 ____private void desabilitaLeituraDosLivros(){
275 ____btn_ler_livro_1.SetActive (false);
276 ____btn_ler_livro_2.SetActive (false);
277 ____btn_ler_livro_3.SetActive (false);
278 ____btn_ler_livro_4.SetActive (false);
279 ____btn_ler_livro_5.SetActive (false);
280 ____}

282 ____private void habilitaCompraDoLivro(int codigoLivro){
283 ____if(codigoLivro == 4)
284 ________btn_comprar_livro_4.SetActive(true);
285 ____else if(codigoLivro == 5)
286 ________btn_comprar_livro_5.SetActive(true);
287 ____}

289 ____private void desabilitaCompraDosLivros(){
290 ____btn_comprar_livro_4.SetActive(false);
291 ____btn_comprar_livro_5.SetActive(false);
292 ____}

294 ____//Fun  o que n o deixa ultrapassar o tamanho do texto das moedas e
       n o deixe de aparecer
295 ____public void toStringMoedas(){
296 ____if(dadosJogo.Instance.currentUser.Coins>=10000)
297 ________coinsUser.text = "+9999";//Caso Valor Ultrapasse os 10000 e n o
       caiba na tela
298 ____else
299 ________coinsUser.text = " "+ dadosJogo.Instance.currentUser.Coins;//
       Mostro na estante o valor
300 ____}

302 ____public void toStringPrecoLivros(){
303 ____booksPrice.text = " "+ dadosJogo.Instance.currentUser.BookPrice;//
       Mostro na estante o valor do livro
304 ____}
305 }
```