

Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

AprEnDO: aplicativo para fixação de EDO's - um relato de experiência

Autor: Leonardo Arthur Degolim Oliveira
Orientadora: Prof^a Dr^a Tatiane da Silva Evangelista
Coorientadora: Prof^a Bruna Nayara Moreira Lima
Data de apresentação: 05/07/2019

Brasília, DF

2019



Leonardo Arthur Degolim Oliveira

AprEnDO: aplicativo para fixação de EDO's - um relato de experiência

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB
Faculdade UnB Gama – FGA

Orientadora Prof^a Dr^a Tatiane da Silva Evangelista
Coorientadora Prof^a Bruna Nayara Moreira Lima

Brasília, DF
2019

Leonardo Arthur Degolim Oliveira

AprEnDO: aplicativo para fixação de EDO's - um relato de experiência

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Prof^a Dr^a Tatiane da Silva Evangelista
Orientadora

Prof^a Bruna Nayara Moreira Lima
Coorientadora

Prof^o Dr. Ronni Geraldo Gomes de Amorim
Convidado 1

Prof^o Dr. Fábio Macêdo Mendes
Convidado 2

Brasília, DF
2019

Resumo

O objetivo deste trabalho era desenvolver um jogo de celular o AprEnDO, para analisar o apoio à aprendizagem de equações diferenciais ordinárias (EDO) de 1^a ordem. O jogo contém perguntas a respeito de classificação e resolução dessas equações. A metodologia de trabalho foi um relato de experiência de uma aplicação de jogo em uma classe de Cálculo 2 (C2) da Faculdade do Gama da UnB em que a professora orientadora ministrou o ensino. Em 1/2019 a turma começou a utilizar o jogo e foram aproximadamente 2 meses de coletas em um servidor de dados para relatar a experiência do AprEnDO em sala de aula. Ao final do trabalho, são relatados os resultados dos dados analisados e as pendências para trabalhos futuros.

Palavras-chave: Equação Diferencial. Aplicativo para celular. Jogo. Software.

Lista de ilustrações

Figura 1 – Prototipação das telas iniciais do jogo	18
Figura 2 – Prototipação das telas dos módulos	19
Figura 3 – Prototipação das telas dos módulos	20
Figura 4 – Processo geral do jogo aprEnDO	23
Figura 5 – Processo de inicialização do jogo aprEnDO	24
Figura 6 – Processo de jogo de classificação	25
Figura 7 – Processo de ganhar uma fase	25
Figura 8 – Processo de jogo de resolução	26
Figura 9 – Envio de estatísticas	26
Figura 10 – Zerar estatísticas após envio	27
Figura 11 – Diagrama de Classe do AprEnDO	27
Figura 12 – Tela inicial	35
Figura 13 – Modo de classificação parte 1	37
Figura 14 – Modo de classificação parte 2	38
Figura 15 – Modo de classificação parte 3	39
Figura 16 – Exemplo da fase de classificação de homogeneidade	40
Figura 17 – Exemplo da fase de classificação de tipo	41
Figura 18 – Feedback fornecido à uma tentativa inválida	42
Figura 19 – Tela de finalização do modo de classificação	43
Figura 20 – Modo de resolução parte 1	44
Figura 21 – Modo de resolução parte 2	45
Figura 22 – Jogo da memória com 1 carta selecionada	46
Figura 23 – Jogo da memória com alguns pares encontrados	47
Figura 24 – Vitória de jogo da memória	48
Figura 25 – Tela de envio de estatísticas	49
Figura 26 – Alerta de matrícula inválida	49
Figura 27 – Alerta de falta de internet	50
Figura 28 – Loader visível enquanto envia estatísticas	51
Figura 29 – Alerta de envio com sucesso	51
Figura 30 – Estatísticas colhidas de classificação	52
Figura 31 – Estatísticas colhidas de resolução	53

Lista de abreviaturas e siglas

ED	Equação Diferencial
EDO	Equação Diferencial Ordinária
PVI	Problemas de valor inicial
VI	Valor Inicial
APP	Aplicativo
OCDE	Organização para a Cooperação e Desenvolvimento Econômico
Cap.	Capítulo
C2	Cálculo 2
UnB	Universidade de Brasília
FGA	Faculdade do Gama
PISA	Programa Internacional de Avaliação de Estudantes
JSON	JavaScript Object Notation
RBIE	Revista Brasileira de Informática na Educação
RENOTE	Revista de Novas Tecnologias na Educação

Sumário

1	INTRODUÇÃO	7
2	REFERENCIAL TEÓRICO	10
2.1	Brasil com Matemática	10
2.2	Sucesso de jogos no ensino	11
2.3	Engenharia de software	13
2.3.1	Teste de software	14
2.3.2	Plataformas mobile	14
3	METODOLOGIA	15
4	ENGENHARIA DE SOFTWARE	16
4.1	Requisitos de Software	16
4.2	Prototipação	17
4.3	Banco de equações	20
4.4	AprEnDO	22
4.5	Servidor aprEnDO	32
4.6	Acesso ao código	34
4.7	Empacotamento	34
5	MANUAL DO APRENDO	35
5.1	Pacote 1	35
5.1.1	Módulo 1	35
5.1.2	Módulo 2	43
5.2	Pacote 2	48
6	ANÁLISE	52
7	CONCLUSÃO	56
	REFERÊNCIAS	57

1 Introdução

Sabe-se que os jogos propiciam diversão ao seres humanos e vê-se que a gamificação está sendo explorada em várias áreas diversas como marketing, saúde e educação. Segundo (NETO; BLANCO; SILVA, 2017) as tecnologias digitais em sala de aula podem ser instrumentos que auxiliam no processo de ensino e de aprendizagem. Quem sabe então se os jogos e os aplicativos gamificados não podem ser usados na educação para ajudar mais pessoas a aprender e se interessarem pelo assunto de Matemática ou qualquer outro. Será que usar jogos digitais para o ensino de Matemática pode ajudar os alunos a se interessarem e aprenderem mais? Se sim, como utilizar?

Existem relatos de trabalhos que usam jogos e gamificação na matemática. Mas a maioria do conteúdo de Matemática para o ensino superior encontrado era cálculo 1. No Brasil os jovens tem dificuldades com matemática. Segundo o relatório do (INEP, 2015a) o Brasil tem uma qualidade de ensino de matemática inferior a de muitos países. A Organização para a Cooperação e Desenvolvimento Econômico (OCDE) utiliza uma escala de classificação que vai de 1 a 6 para as habilidades de matemática e de acordo com o conjunto da população brasileira que participou da pesquisa no Programa Internacional de Avaliação de Estudantes (PISA), inferiu-se que 70.3% dos estudantes brasileiros estão abaixo do nível 2, o qual foi estabelecido como o mínimo para exercer a cidadania como cidadão pleno (INEP, 2015a). O estudo é realizado a cada três anos. Foi realizado no ano passado, porém até o momento o resultado ainda não foi publicado <<http://portal.inep.gov.br/web/guest/acoes-internacionais/pisa/resultados>>.

Além das informações relatadas acima outras três também contribuíram para formular a proposta deste trabalho. Um deles foi o estudo de (NETO; BLANCO; SILVA, 2017), que é uma revisão sistemática de literatura realizada nas bases de dados Scielo Library, Science Direct, ACM Library e IEEE Xplore Digital Library entre outras e nos periódicos RBIE e a RENOTE. O estudo procurou relatos em artigos da existência de ferramentas relacionadas com gamificação que abordem dificuldades de aprendizagem de Matemática e/ou Discalculia. De 2008 trabalhos selecionados, nenhum eram relacionando gamificação e dificuldades de matemática ao mesmo tempo. Com essas conclusões encontradas no estudo os autores (NETO; BLANCO; SILVA, 2017) concluíram que:

"identifica-se a necessidade de pesquisas sobre esta temática (gamificação com dificuldades de aprendizagem de Matemática), já que as dificuldades de aprendizagem na Matemática são frequentes em sala de aula, e a gamificação tem-se mostrado uma ferramenta promissora nos ambientes de ensino e aprendizagem em todos os níveis de ensino"(NETO; BLANCO; SILVA, 2017).

No mesmo estudo de (NETO; BLANCO; SILVA, 2017) o autor cita (DICHEVA *et al.*, 2015) dizendo que "a falta de pesquisa na área é justificada por ser uma temática nova.", o que pode aumentar com o tempo caso as pessoas se interessem por essa 'temática nova' de utilizar gamificação na educação e para o ensino de matemática.

Outro fator foi o estudo de (SOUZA, 2016) o qual diz que Cálculo 2 (C2) é uma disciplina das que mais causa a evasão dos alunos do curso de Matemática noturno na UnB. Após saber da existência do estudo no site de monografias da unb houve a comparação da ementa de C2 no curso de Matemática noturno do Darcy Ribeiro com a ementa de C2 do tronco comum no curso de Engenharias da UnB no Gama e foi constatado que há a equivalência dos conteúdos ministrados das disciplinas. Ambas iniciam para os alunos o conteúdo de equações diferenciais. Assunto que por acaso é pouco encontrado na literatura de gamificação. O mais encontrado é voltado para área de limite, derivada e integral como mostrado o aplicativo em (SILVA *et al.*, 2016) e também presente na própria FGA. Verificou-se que há poucos jogos de Matemática para o conteúdo de equação diferencial. O único jogo de equações diferenciais encontrado é um de vídeo-game que aplica as equações diferenciais na movimentação dos personagens (princípio da dinâmica de Newton)(GIACINTI *et al.*, 2013). A maioria dos jogos de Matemática é para o ensino fundamental.

Existem também estudos mostrando como a tecnologia da suporte para jogos na hora do ensino e ajuda na fixação do conhecimento.

Tendo relatado as dificuldades no ensino de matemática e a possível ajuda de gamificação e jogos digitais, gerou-se a questão: Como dar suporte no ensino de EDO 1^a ordem apresentados em sala de aula de forma lúdica? A partir deste questionamento, levantou-se o objetivo: desenvolver um jogo para celular Android que dê suporte ao ensino de equações diferenciais ordinárias (EDO) de 1^a ordem. O jogo visa treinar os alunos a reconhecer, classificar e resolver equações diferenciais presentes no dia-a-dia e no ambiente da engenharia.

Decidiu-se fazer um jogo para celular com o intuito de inserir no ambiente dos alunos uma ferramenta a mais (o jogo) para ajudar os estudantes a aprenderem mais e se possível se divertindo, ou que pelo menos seja interessante. Deseja-se que seja um meio de treinamento e fixação do conhecimento para aprenderem.

A primeira parte da metodologia seguida foi a pesquisa bibliográfica para levantar o referencial teórico a respeito da contribuição efetiva de jogos e seu sucesso no ensino. Já a segunda parte da metodologia foi um relato de experiência do desenvolvimento do software e a aplicação em alunos de C2 na universidade do Gama, utilizando estatísticas levantadas do jogo chamado aprEnDO para avaliar o impacto do aplicativo educacional para celular como ferramenta de suporte didático no ensino principalmente em classificação de EDO

de 1º nível.

Utilizou-se o processo de prototipação de baixa fidelidade para desenhar as telas do jogo pois ela serve como auxílio para uma das etapas do processo de desenvolvimento de software que é a elicitação dos requisitos. Houve a consolidação dos requisitos funcionais que foram definidos e a modelagem de um diagrama de classe para explicar a aplicação AprEnDO que é um projeto do *react native*. São indicados os domínios utilizados para baixar as dependências que compõem o software e toda a parte da hospedagem dos códigos tanto da aplicação, como para gerar o arquivo de alimentação e os que definiram o servidor que guarda os dados das estatísticas enviadas pelos alunos. O projeto pode ser reutilizado por quem desejar e ajudas na contribuição são bem vindas. Com o jogo pronto será aplicado em um grupo da turma de C2 no período do primeiro semestre de 2019 para que possa ser gerado dados e estatísticas para concluir se o jogo trouxe alguma eficiência no aprendizado ou não.

O jogo contém três módulos. Dois deles para jogar com diferentes fases e dificuldades e o outro restante serve para envio de estatísticas dos dados dos jogadores para um servidor com um banco de dados. Foi desejado planejá-lo para ser 'fácil' de enviar os dados colhidos para análise. É chamada de fácil por ser considerada rápida do jogador utilizar. Exige-se apenas a matrícula do aluno (que será validada) e a confirmação de um clique para acordar compartilhar suas estatísticas de variações entre o tempo atual e o último envio.

Houve uma falta de planejamento no desenvolvimento de software devido aos atrasos de elicitação dos requisitos o que impossibilitou a completude de alguns e como o jogo havia data certa para início de aplicação impossibilitou o término do desenvolvimento de alguns requisitos.

O capítulo 2 abordará o referencial teórico, dando ênfase nos baixos índices de classificação do Brasil no conhecimento de matemática, apoiando a gamificação e jogos como uma prática que deixa as tarefas e atividades mais divertidas, revisando conceitos de engenharia de software para desenvolvimento de jogos. O capítulo 3 explica a metodologia do trabalho seguida. O capítulo 4 fala a respeito da engenharia de software aplicada no projeto, cita requisitos do jogo, artefatos de auxílio, ambiente de desenvolvimento, o banco de dados, como é o processo de empacotamento e o servidor que recebe dados de jogo. O capítulo 5 é o manual do AprEnDO que explica as fases do jogo, como espera-se que ele seja jogado e fotos do aplicativo. O Capítulo 6 mostra a análise dos dados colhidos. O capítulo 7 apresenta a conclusão do trabalho e o possíveis atividades futuras e o capítulo 7 mostra as referências do trabalho.

2 Referencial Teórico

O referencial teórico presente discorrerá a respeito de quatro temas para defender a ideia deste trabalho. O primeiro tema contextualiza o Brasil com a matemática, depois abordaremos o uso de jogos como um facilitador no ensino, seja para jogos de matemática ou não, sejam jogos eletrônicos ou não. O terceiro tema justificará o uso do mapa conceitual como uma ferramenta de validação de conhecimento adquirido e por fim trará conceitos da engenharia de software para um bom processo de desenvolvimento de software em jogos e/ou aplicações gamificadas.

2.1 Brasil com Matemática

No Brasil 70.3% dos alunos estão abaixo do nível de conhecimento em Matemática. Nível este que de acordo com a Organização para a Cooperação e Desenvolvimento Econômico (OCDE) foi estabelecido para medir a capacidade do alunos em exercer plenamente sua cidadania ([INEP, 2015a](#)). A qualidade do ensino de Matemática no Brasil é ruim de acordo com ([INEP, 2015b](#)). O estudo do INEP é realizado a cada três anos e é lançado no final do ano seguinte. Foi realizado pela última vez em 2015 quando o Brasil foi 13º colocado em um estudo com 14 países participantes da OCDE. Ficou na frente da República Dominicana e atrás de países como Coréia do Sul, Canadá, Portugal e Estados Unidos. De acordo com o ([ESTADÃO, 2016](#)) a posição do Brasil para a qualidade do ensino de Matemática e ciências é 133 entre 139 países participantes.

Um dos porquês desses índices baixos é que existe o desânimo em salas de aula, as vezes por parte dos professores e outras por parte dos alunos. Os professores precisam se reinventar para atrair a atenção dos alunos e melhorar a eficiência do aprendizado em sala de aula. Parte do desânimo dos alunos em sala de aula deve-se por achar a Matemática como algo chato, não entenderem o conteúdo e não terem uma base de conteúdo bem solidificado.

Outro problema é que existem poucos estudos relacionando gamificação com Matemática ([NETO; BLANCO; SILVA, 2017](#)), principalmente quando se fala de Matemática no ensino superior. Quando se encontra Matemática para nível superior com gamificação os estudos são focados para o conteúdo de cálculo 1 (limite, derivada e integral). Nada foi encontrado relacionado ao contexto de gamificação com equações diferenciais. Nenhum jogo de equações diferenciais (ED) foi encontrado.

O estudo ([NETO; BLANCO; SILVA, 2017](#)) fez um levantamento bibliográfico sobre gamificação com Matemática e dificuldades no ensino de Matemática e não encontrou

nenhum estudo na área de gamificação com dificuldades de aprendizado em Matemática.

Uma das maneiras de ajudar os alunos a se interessarem mais em sala de aula e atrair a atenção dos mesmos é utilizar o lúdico, ou seja, aprender brincando. Para isso o uso de computadores ou tecnologias da informação como o celular é útil para melhorar o engajamento nas tarefas, principalmente com exercícios e aplicações para a prática das matérias ensinadas em sala de aula ([DUPAUL; STONER, 2007](#)).

2.2 Sucesso de jogos no ensino

Este tópico visa apoiar jogos e gamificação como uma estratégia boa para aprendizado dos jogadores e também como boa ferramenta para ser utilizada como aprendizagem. Para isso serão mostrados citações de autores que reforçam isso que foi dito, assim também como exemplos de casos reais da utilização de jogos em ambientes educacionais.

Jogo é prática que ajuda na concretização do conhecimento, além de tornar o ambiente mais prazeroso ([COELHO, 2010](#)).

"O caráter lúdico, bem como a possibilidade de atuação crítica, proporciona ao aluno uma participação efetiva no processo de ensino aprendizagem, se tornando um momento ímpar de crescimento pessoal e coletivo."([COELHO, 2010](#)). O que significa que contribui para o aluno se tornar um ser ativo e pensante, capacitando-o a exercer seu papel como cidadão.

"Os jogos despertam o interesse dos jovens trazendo diversos benefícios aliados à educação[...]"([SILVA et al., 2016](#))

"Na literatura, encontram-se vários trabalhos que demonstram profissionais de educação utilizando os jogos como ferramenta de auxílio ao aprendizado"([SOUZA; FRANÇA, 2016](#), p. 3)

"Como a finalidade na sala de aula é estimular ideias dos alunos, ensinar apenas com aulas expositivas tradicionais pode dificultar o aprendizado."([SOUZA; FRANÇA, 2016](#), p. 4)

"Uma das propostas metodológicas para ensino de engenharia de software e suas disciplinas, são os jogos educacionais. Sabe-se que os jogos educacionais, segundo Nunes e Parreira (2015), têm sido intensamente utilizados por profissionais da área de educação como auxílio para a construção do conhecimento. Em sua pesquisa, Fukusawa et al. (2015) apontam alguns dos benefícios que os jogos educacionais podem trazer ao processo de ensino e aprendizagem como, por exemplo, a motivação e o aprendizado por descoberta. Portanto, os jogos podem proporcionar a vivência em experiências de aprendizagem concretas ([MONSALVE; WERNECK; CESAR, 2010](#) apud [SOUZA; FRANÇA, 2016](#), p. 4)."

Silva et al. (2015) comentam que uma abordagem alternativa às aulas tradicionais, devido a elas serem mais teóricas e expositivas, é a utilização dos jogos, pois esta abordagem preza por uma teoria de motivação humana como ponto de partida (SOUZA; FRANÇA, 2016).

"Uma das propostas de melhoria de aprendizado em sala de aula são os jogos educacionais" (SOUZA; FRANÇA, 2016, p. 4)

Foram pesquisados nos Anais da base WEI e selecionados os artigos que tivessem trabalhos com jogos que auxiliassem no aprendizado de ensino superior das disciplinas de Engenharia de Software, referencialmente os que validassem com alunos. (SOUZA; FRANÇA, 2016)

Jogos e gamificação diferem-se, porém ambos já vem sendo usados em ambientes de ensino. Existem vários exemplos de casos de sucesso, como por exemplo "O bicho papão da matemática virou um gatinho", é um jogo de matemática para alunos do 1º e 2º ano fundamental. O jogo ajuda a fazer divisões. Segundo a notícia no link <<http://portal.mec.gov.br/component/content/article?id=72701>>, 300 alunos se beneficiaram deste projeto. Um dos símbolos que ficou marcado era de uma aluna que tinha reprovado, tirava notas baixas, não interagia muito com os outros alunos e acabou se envolvendo, aprendendo o jogo de tal maneira que passou a tirar 10, ir resolver no quadro e se sentir capaz. É relatado também que não melhorou só em matemática, como em outras matérias. Segundo o professor, além da menina citada, muitos outros que não sabiam divisão aprenderam também.

Outros exemplos de jogos sendo utilizados em contextos educacionais: Este estudo (SOUZA; FRANÇA, 2016) fez um levantamento de jogos para o uso específico na engenharia de software. São listados 20 jogos no apêndice, com cada um focado em uma disciplina do curso.

Mais um estudo (SILVA *et al.*, 2016), que é uma proposta de aplicativo gamificado para ensino de cálculo onde é proposto um jogo para o ensino de matemática com o conteúdo voltado para os temas de conjunto, limite, derivada e integral, não é voltado para o tema de EDO 1^a ordem.

Então até aqui é possível ver que existem muitos jogos no ensino. Porém jogos para matemática no ensino superior não foram encontrados muitos trabalhos, e afunilando um pouco mais para jogos de equação diferencial não foi encontrado nenhum.

Atratividade de jogos está relacionada a mecanismos psicológicos e sociais (SOUZA; FRANÇA, 2016)

Para o jogo ser bem aceito e cumprir com a sua meta, ele deve dar uma boa base de conhecimento e motivação (SOUZA; FRANÇA, 2016).

O computador e tecnologias como celular, além de serem ferramentas de auxílio, são também motivadoras para os estudantes ([SANTOS, 2017](#)).

Este tópico visa reafirmar o potencial dos jogos tecnológicos para ser usado como ferramenta de atração dos alunos para os colégios e universidades e as citações neste tópico visam reforçar que jogos podem ser utilizados para fazer os estudantes gostarem e se atreverem mais no contexto da matemática. Espera-se que os alunos busquem e tenham a vontade do conhecimento por si próprio para que se tornem mais independentes. Também conclui-se que existem poucos estudos de jogos na área de C2, específico para ED, apesar de terem estudos na área de matemática, estes destinam-se a cálculo 1 e matérias do ensino fundamental.

2.3 Engenharia de software

Hoje em dia softwares estão presente em todos os lugares, tudo que a gente 'toca'. Softwares não se desgastam (físicamente) como hardware, mas estão sujeitos a modificações durante o seu ciclo de vida ([FILHO, 2015](#)). As modificações as vezes podem causar efeitos acidentais e/ou não esperados. Um software precisa de modificações conforme o tempo passa e para isso acontecer com mais tranquilidade exige a necessidade de uma documentação. Quanto melhor a documentação, mais fácil para entender. Para manter um bom projeto de software é necessário ter ou criar uma cultura de engenharia de software para adotar as melhores práticas. As quais que compreendem os pilares de custo, tempo de desenvolvimento e qualidade de software ([FILHO, 2015](#)).

No glossário de terminologia de Engenharia de Software da IEEE Std 610.12-1990, define-se Engenharia de Software como a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção de um software; isto é aplicação de engenharia de software. Engenharia de software também pode ser o estudo das abordagens ([IEEE, 1990](#)).

Como softwares precisam de manutenção corretiva e/ou evolutiva, também está sujeito a inserção de defeitos decorrentes do desenvolvimento. Estes defeitos podem ser vistos e consertados antes da entrega ([FILHO, 2015](#)) ou ser descoberto pelo usuário que está utilizando e não espera se deparar com o erro. Por isso além de uma documentação, software também precisa de testes, estes que quando bem feitos asseguram a qualidade e confiabilidade do produto de software.

Engenharia de software está presente e tem descrições de melhores práticas em todas as fases desde a concepção, elaboração, construção e transição de um projeto. Seja em um projeto com metologia tradicional ou ágil, um projeto passa por essas fases. A engenharia de software "tem como objetivo apoiar o desenvolvimento profissional de software,

cobrindo todos os aspectos da produção de um software."(MONSALVE; WERNECK; CESAR, 2010 apud SOUZA; FRANÇA, 2016)

[(SOUZA; FRANÇA, 2016)] (??) concordam que a engenharia de software é uma área muito jovem e sofre contínuas mudanças nos seus fundamentos tecnológicos concretizadas nos métodos e ferramentas de suporte, portanto necessita de métodos de ensino lúdicos e dinâmicos que possam contribuir na aprendizagem do estudante.

Eng de Software percorre o levantamento de requisitos de um jogo, o planejamento e desenvolvimento das funcionalidades, testes para garantir que está tudo funcionando como o ocorrido e o empacotamento e a entrega para a finalização. Após cumprir este ciclo, para um software continuar 'vivo' é necessário que com o tempo ele receba manutenção e melhorias e também se deseja, evoluções. Além de agregar valor para o cliente ou desenvolver o acordado, também é necessário gerenciar sua infra-estrutura, definir onde hospedar as aplicações, os custos decorrentes de utilizar serviços ou hardwares de terceiros, realizar as configurações para a padronização e fazer o controle de mudanças e gestão da qualidade.

2.3.1 Teste de software

Teste de software é uma atividade importante do desenvolvimento de software pois está relacionado à qualidade de software, este pode ajudar a verificar o cumprimento dos requisitos.

(VALLE, 2017, p. 17) diz que um teste tem basicamente 4 fases, o planejamento, projeto, a execução e a avaliação do resultado dos testes. Já (PRESSMAN, 2011), (DELAMARO; MALDONADO; JINO, 2016) dizem que os testes devem acontecer ao longo do processo de desenvolvimento do software, pois é o momento onde as funcionalidades estão frescas no pensamento do desenvolvedor e este deve garantir com testes que ocorra o funcionamento esperando das funções.

2.3.2 Plataformas mobile

Existem diversas maneiras de se construir APP para mobiles. Algumas apenas para celulares Android, outras para sistemas iOS e outras para ambas plataformas. Existem estratégias de desenvolvimento onde o código gerado já é nativo da própria plataforma alvo e outras onde o código é transformado para a plataforma nativa. Existe um projeto chamado kivy, onde é escrito código Python e o kivy converte o código para gerar aplicações para Android e iOS.

Outra estratégia é o react native, onde o código é escrito utilizando HTML, CSS e JavaScript com o react e parte desse código é convertido em nativo para rodar com maior eficiência nos celulares.

3 Metodologia

A metodologia utilizada foi a de relato de experiência. Foi levantado um referencial teórico a respeito do auxílio da contribuição efetiva de jogos e seu sucesso no ensino e os conceitos da Engenharia de Software.

O relato de experiência da aplicação do jogo aprEnDO em alunos de C2 na universidade do Gama, levantou estatísticas do jogo para avaliar o impacto do aplicativo como ferramenta de suporte ao aprendizado em EDO de 1º nível.

O conteúdo da matéria ministrado aos alunos durou cerca de 1 mês. Após o período de ensino da matéria, a turma utilizou a aplicação em torno de 2 meses para o reforço do conteúdo e neste tempo foram enviando suas estatísticas para um servidor central. A análise de estatísticas enviadas do jogo foi quantitativa e os dados foram consolidados em tabelas para análise.

Serão relatados bugs, erros e falhas encontrados pelos alunos durante a fase de jogo. Também serão relatados os *crashes* do servidor de dados, caso ocorram.

4 Engenharia de Software

4.1 Requisitos de Software

Os requisitos de software foram levantados e utilizados na hora do desenvolvimento para alcançar os objetivos elencados para o jogo.

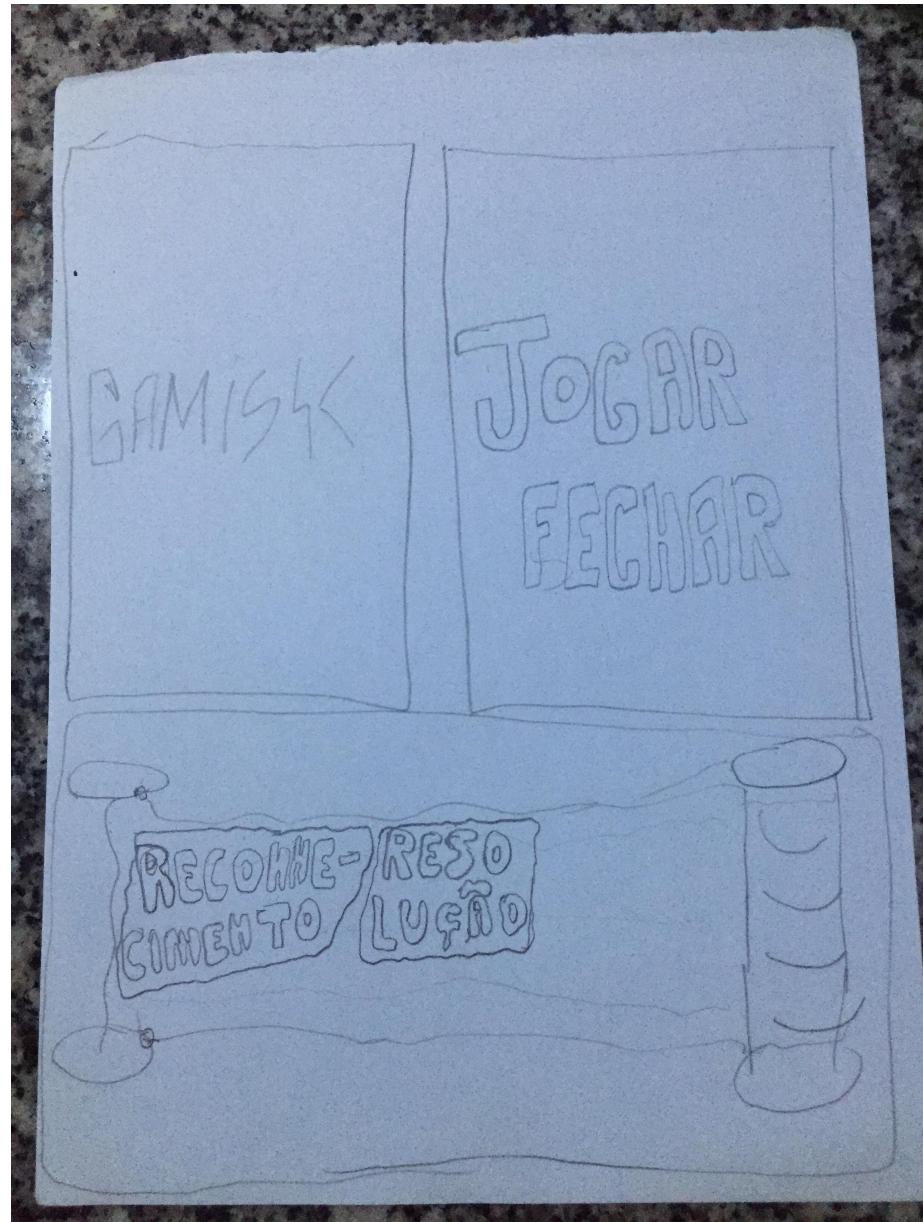
- Ter um módulo para classificação de EDO
 - Ter fase de classificação de ordem
 - Ter fase de classificação de tipo
 - Ter fase de classificação de linearidade
 - Ter fase de classificação de homogeneidade
 - Ter fase de classificação de separável
 - Ter fase de classificação de exata
 - Escolher 20 questões aleatórias do banco de equações
- Ter um módulo para resolução de EDO de 1^a ordem
 - Ter fase de resolução de EDO 1^a ordem homogênea
 - Ter fase de resolução de EDO 1^a ordem não homogênea
 - Ter fase de resolução de EDO 1^a ordem exata
 - Ter fase de resolução de EDO 1^a ordem não exata
 - Escolher 10 pares de equações aleatórias do banco de equações
- Estar disponível para download no google play
- Sinalizar ao usuário fases concluídas
- Permitir envio de bugs e erros
 - Notificar se o erro é por falta de internet
 - Notificar se o erro é por problema do servidor
- Permitir envio de sugestões e feedback
 - Notificar se o erro é por falta de internet
 - Notificar se o erro é por problema do servidor

- Ter um módulo de envio de estatísticas
 - Identificar o aluno pela matrícula
 - Enviar estatísticas de classificação
 - * armazenar fases concluídas
 - * armazenar quantidade de vezes que cada fase foi concluída
 - * armazenar 20 questões presentes na tentativa
 - * armazenar tempo total gasto em cada conclusão
 - Enviar estatísticas de resolução
 - * armazenar fases concluídas
 - * armazenar quantidade de vezes que cada fase foi concluída
 - * armazenar 10 questões presentes na tentativa
 - * armazenar tempo total gasto em cada conclusão
 - Sinalizar ao usuário quando a estatística for enviada
 - Sinalizar ao usuário quando a estatística não for enviada
 - * Notificar se o erro é por falta de internet
 - * Notificar se o erro é por problema do servidor

4.2 Prototipação

A prototipação não seria utilizada, até deparar-se com o problema de não saber como seria a imagem final das telas, então optou-se por utilizar a prototipagem de baixa fidelidade, onde o importante seria ter uma noção geral de como seriam as telas. Desenhou-se então no papel um exemplo de como imaginava-se que seria a tela final do jogo. O primeiro desenho foi o seguinte

Figura 1 – Prototipação das telas iniciais do jogo



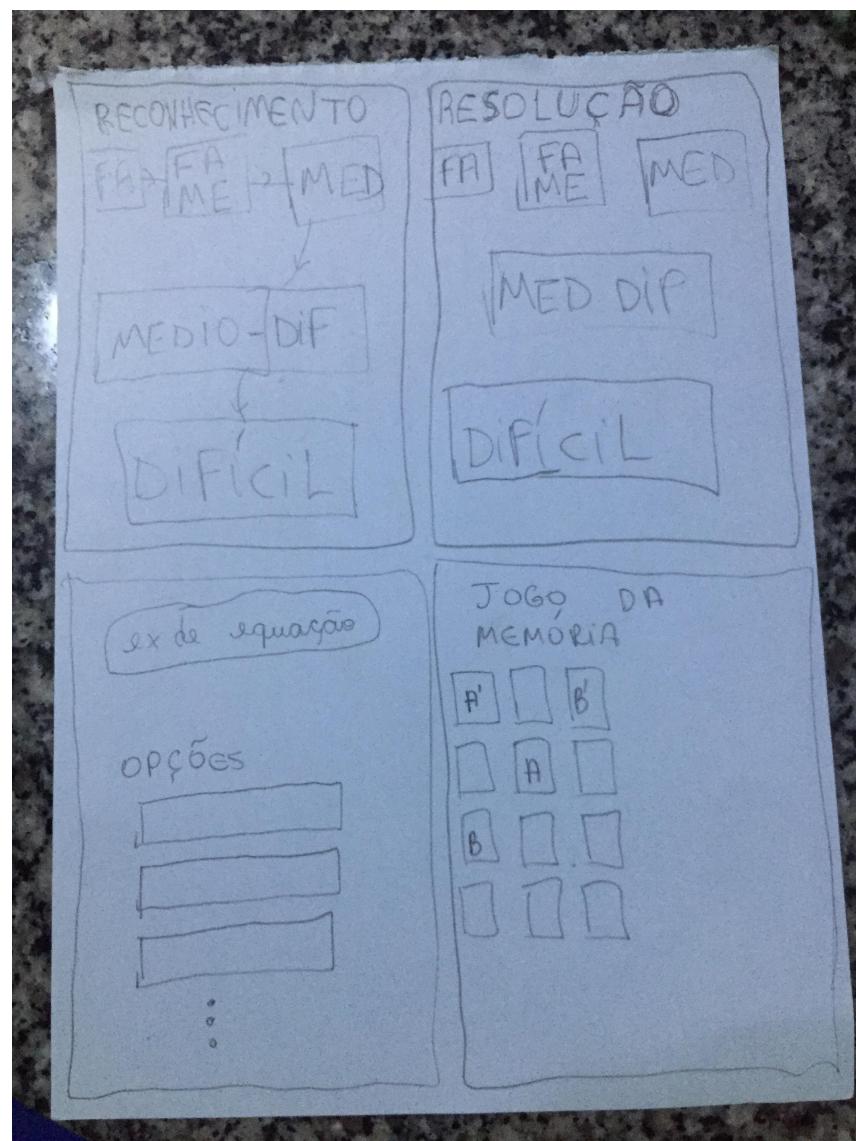
Fonte: do próprio autor

Na figura 1 no primeiro retângulo em cima e à esquerda é possível ver escrito "Gamisk", que seria o primeiro nome do jogo e a primeira tela a ser apresentada ao clicar no ícone do APP. Ao lado do retângulo Gamisk à direita é possível ver escrito "Jogar" e "Fechar" que seria a segunda tela do jogo. Porém percebeu-se que não haveria necessidade das duas telas e então decidiu-se pular direto para a que apresenta "RECONHECIMENTO" E "RESOLUÇÃO". A necessidade da tela gamisk seria em caso do jogo demorar para carregar recursos, o que não acontece. A necessidade da segunda tela era para ter o botão fechar, para o jogador poder sair do jogo, porém os celulares android apresentam o botão "Home" para sair da aplicação, então viu-se desnecessário criar a funcionalidade de sair do jogo. Então ao abrir o APP o usuário já será redirecionado para escolher o módulo de

jogo que deseja jogar.

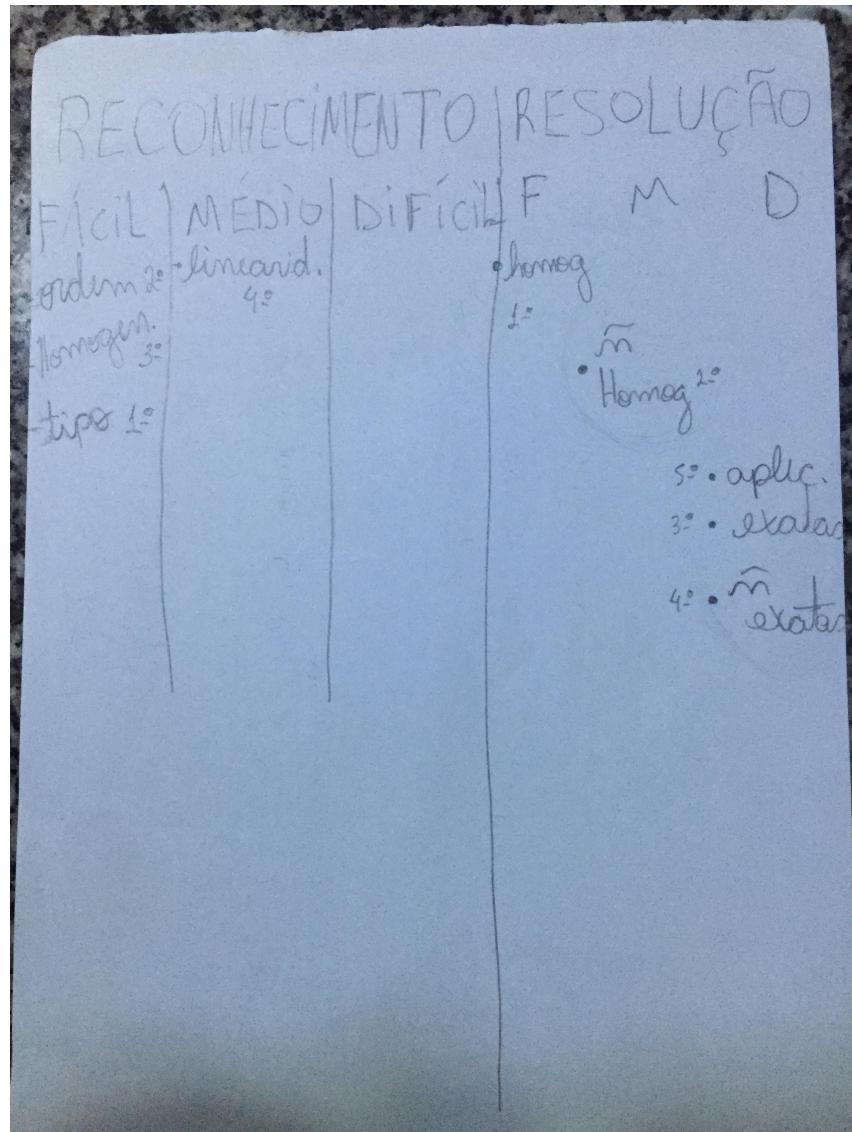
A figura 2 retrata como seria o jogo enquanto pensava-se que existiria os níveis de dificuldades fácil, médio e difícil. Depois houve o mapeamento dos níveis de dificuldade para o indicado na figura 3. A mudança foi então que ao invés das fases terem nome 'fácil', 'médio' e 'difícil', elas passaram a ter o nome de 'ordem', 'homogeneidade', 'tipo' e 'linearidade' e foram ordenadas de acordo com o que acreditava-se ser mais fácil para classificar até o mais difícil.

Figura 2 – Prototipação das telas dos módulos



Fonte: do próprio autor

Figura 3 – Prototipação das telas dos módulos



Fonte: do próprio autor

4.3 Banco de equações

Antes de existir o modelo definitivo dos arquivos de seeds, foi modelado a primeira versão do modelo relacional. Este modelo tinha 5 entidades, cada uma com seus atributos. As entidades modeladas foram:

- EQUAÇÃO_DIFERENCIAL com os atributos linearidade, separável, homogênea, exata e o id como chave primária.
- ORDEM com os atributos primeira, segunda, terceira, ordem superior
- DIFICULDADE com os atributos fácil, facilmedio, medio, mediodifícil e difícil
- TIPO com os atributos ordinária e parcial

- PERGUNTA com os atributos img_src, largura e comprimento
- RESPOSTA com os atributos img_src, largura e comprimento

Houve a reflexão de criar uma entidade comum para pergunta e resposta como por exemplo IMAGEM, onde PERGUNTA e RESPOSTA herdariam as propriedades. Após outra reflexão, optou-se por não utilizar o modelo do banco porque foi julgado que o problema não era tão complexo e os dados poderiam estar guardados em pastas organizadas ao invés de um banco.

O WolframAlpha será utilizado para fazer requisições de EDO's para serem utilizadas nas fases do jogo. Com uma chave de teste gratuita serão baixados os metadados em formato JSON através de uma API. A API baixada do wolfran na linguagem javascript foi baixada no endereço <<https://products.wolframalpha.com/api/libraries/javascript/>>. A chave gratuita permite 2000 requisições em um mês, com o código de série: 3GGQAT-98EG4KV6VL. Foi desenvolvido um script alimentado por um arquivo de *seeds* que realiza as requisições para a API do Wolfram Alpha para ler os metadados de cada EDO's e guardar/baixar os dados necessários. Os metadados são no formato *JSON*. Eles fornecem a url para a imagem .gif das equações perguntas e respostas (quando disponível), estas são utilizadas no jogo, então é necessário mapeá-las em um arquivo index.js para adicionar na pasta resource do jogo no *react native* para fazer com que a internet não seja um requisito para jogar, porém para enviar dados ao servidor será necessário o acesso à internet e quando não disponível, tem como requisito avisar o jogador para tentar novamente mais tarde e espera-se então que o procedimento de zerar as estatísticas não seja acionado.

Na pasta banco existe um arquivo chamado **seeds.txt** que é o arquivo com as equações diferenciais para alimentar o banco de dados da aplicação aprEnDO. O script **equações.js** lê o arquivo de seeds equação por equação, faz a requisição para o Wolfram Alpha utilizando os códigos da pasta wolfran_api e requisita todos os pod disponíveis (pod são os arrays de informação disponibilizados). Após ter os pods são filtradas as informações desejadas e salvas em arquivos de informações localizadas na pasta **info**. Os pods apresentam as *urls* das imagens de equações de perguntas e respostas, quando existe resposta. Equações sem solução só podem ser utilizadas no primeiro módulo do jogo, o de classificação e não são incluídas no módulo de resolução. As imagens de perguntas baixadas são salvas na pasta **pergunta** e as imagens de respostas salvas na pasta **resposta**.

Com todas as informações desejadas de cada equação é possível utilizar o arquivo **estatisticas.js** que lê todos os arquivos de informação para contabilizar as informações da quantidade de equações diferenciais, quais tem resposta e quais não, quais são homogêneas, exatas, separáveis, linear, não linear, ordem1, ordem2, ordem3, ordem de 4 para cima são consideradas ordem superior, além de fazer a contagem total,também são indicados o número da equação. Essas informações são escritas num arquivo de controle para

que possa ser lido pelo aplicativo aprEnDO e fazer a seleção das equações correta para renderizar, a depender do nível que a pessoa está jogando. O nome do arquivo de controle é **DADOS_GERAIS.json**.

4.4 AprEnDO

A linguagem de programação utilizada é o nodejs com o framework react native para gerar aplicação em código nativo android. Para baixar os pacotes e fazer o controle dos mesmos está sendo utilizado o nvm e o yarn.

O ambiente de desenvolvimento usa um emulador para simular o jogo.

A pasta do jogo aprendo que encontra-se dentro da pasta desenvolvimento tem o projeto react-native com node.js e android, o package.json com as dependências, o index.js que é o ponto de entrada, os códigos fonte e os recursos. Também tem um arquivo makefile que por padrão abre o emulador de celular android, com permissão de superusuário inicia o servidor do react para desenvolvimento, cria no emulador a mais recente build do aplicativo e inicia o terminal para logs do celular emulador.

Os arquivos da pasta e uma breve explicação: O arquivo index.js faz a importação dos módulos que são também as telas do jogo localizadas na pasta screen do código fonte em src. Também importa o módulo de navegação entra telas que precisa ser configurado para estar ciente das screens que antecedem a atual, sem a referência ao voltar a tela o jogo fecha. Uma configuração default que foi setada era para desaparecer a barra superior de navegação das telas, porém percebeu-se que em alguns celulares a barra aparecia e um pouco desconfigurada, já em outros modelos comportava como o esperado e estava invisível.

A pasta android tem o código gerado pelo react-native o qual é um projeto padrão com os arquivos necessário para andoid, código fonte em .java, o arquivo manifest, gradle e arquivos de configuração. Algumas dependências do jogo em react native foram adicionadas manualmente no código nativo do android.

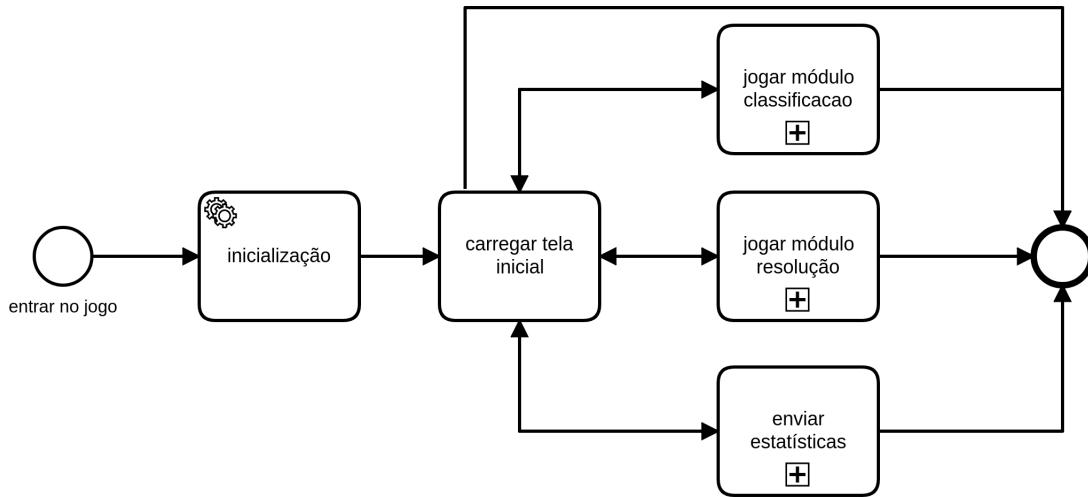
A pasta node_modules apresenta os vários módulos do node instalados em background

A pasta resource tem as imagens de todas as equações de perguntas usada no módulo de classificação, as respostas usadas no módulo de resolução e seus arquivos index.js que mapeia o número das equações com a importação das mesmas dentro do jogo, a importação só se da porquê todas as imagens já possuem o termo “require(‘numero da equação’)” que é pré-processada antes da escolha da equação aleatória a ser importada dentro da fase. Outro arquivo importante que tem nessa pasta é o **DADOS_GERAIS.json** que possui os metadados das equações, quanto à todas classificações, tipo, ordem, homogeneidade e

etc.

Foram desenhados processos para entender o fluxo do jogo. O primeiro processo é uma visão geral.

Figura 4 – Processo geral do jogo aprEnDO

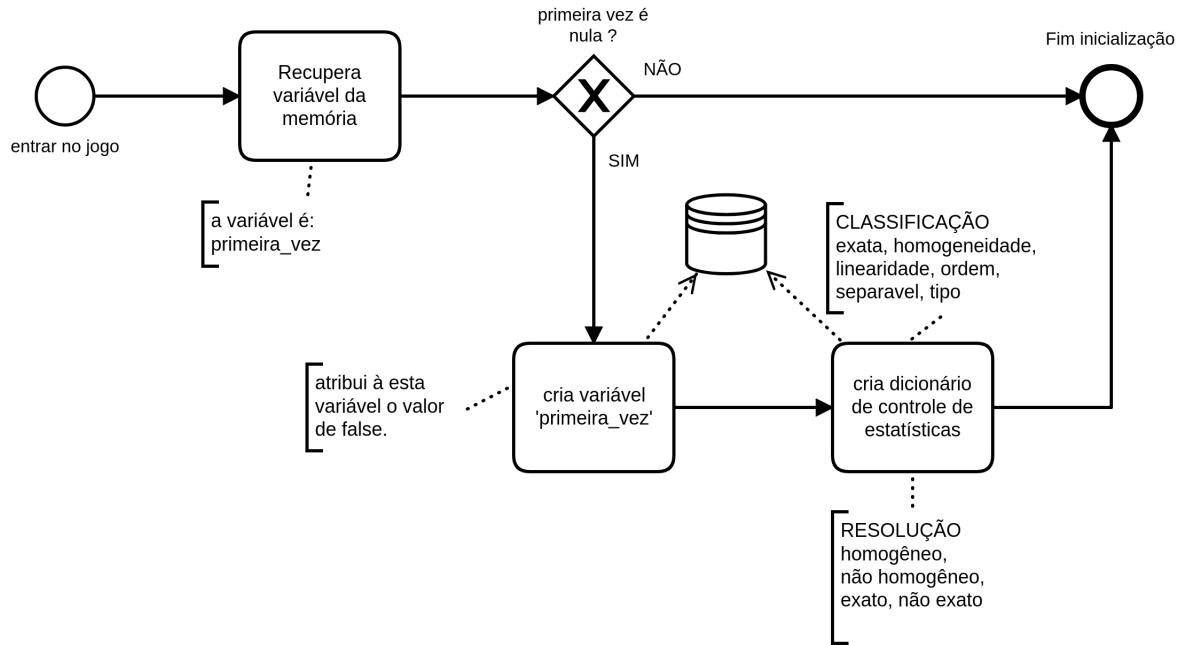


Fonte: do próprio autor

Na figura 4 mostra-se que ao entrar no APP a primeira atividade é um script do sistema, assim que este termina é carregada a tela inicial que permite ao usuário jogar o módulo de classificação, jogar o módulo de resolução, enviar estatísticas ou simplesmente sair do jogo sem fazer nada.

A primeira macro-atividade é a inicialização, indicada na figura 5.

Figura 5 – Processo de inicialização do jogo aprEnDO

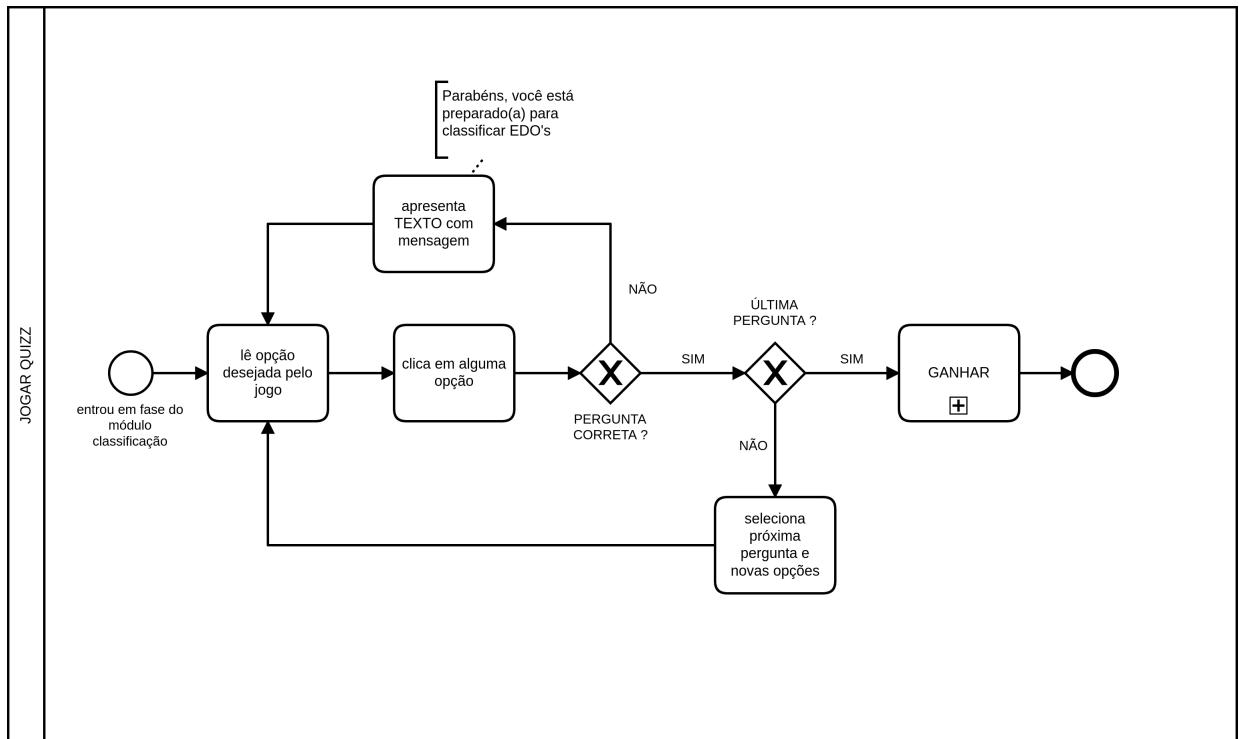


Fonte: do próprio autor

O processo 5 recupera a variável 'primeira_vez' da memória, caso a variável exista e tenha o valor de 'falso' significa que o usuário já jogou outras vezes, então o processo termina. Caso seja realmente a primeira vez do usuário é necessário criar a variável 'primeira_vez' para sinalizar que não é mais a primeira vez. Também é criado a estrutura de estatísticas para classificação e resolução e persistida no banco de dados do celular.

A segunda macroatividade é o processo de jogar o módulo de classificação, nele o usuário precisa escolher alguma fase de classificação. A figura 6 mostra quais são as possíveis ações a serem tomadas pelo jogador.

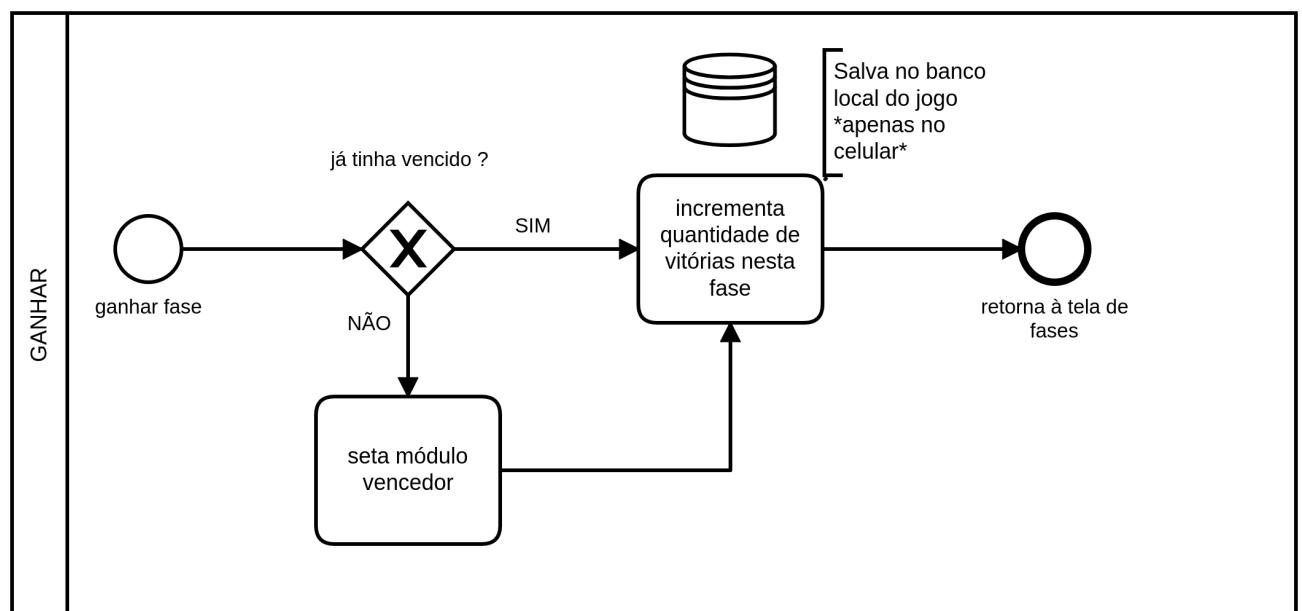
Figura 6 – Processo de jogo de classificação



Fonte: do próprio autor

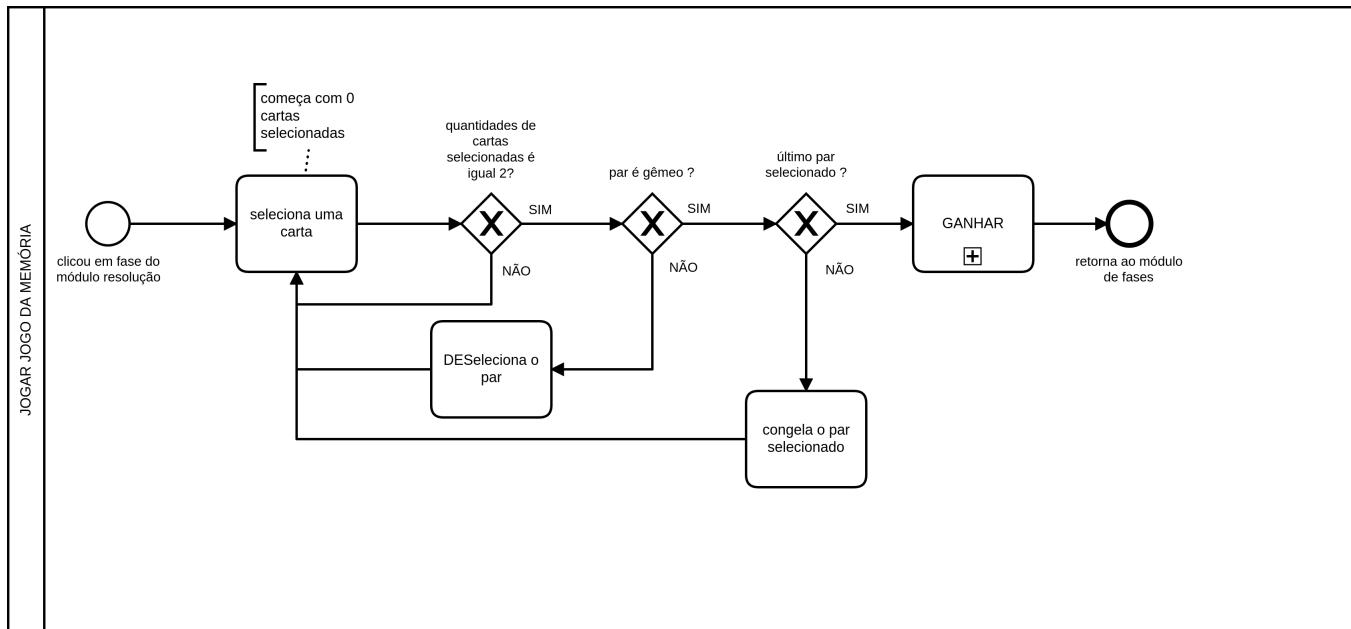
Ámbos os módulos 6 e 8 podem ativar o processo da figura 7 que consiste em armazenar as estatísticas de mais uma vitória nos dados do celular para que possam ser enviados ao servidor de coleta.

Figura 7 – Processo de ganhar uma fase



Fonte: do próprio autor

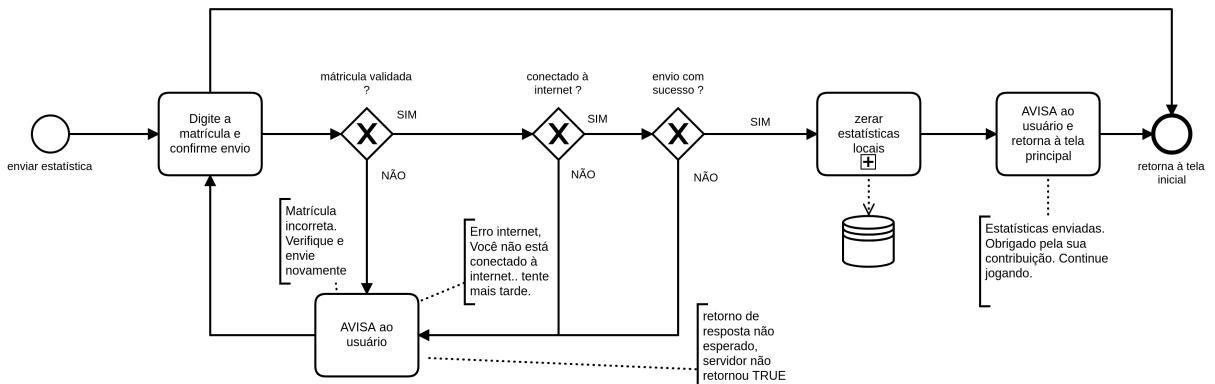
Figura 8 – Processo de jogo de resolução



Fonte: do próprio autor

A quarta macro-atividade envolve o envio de estatísticas de jogo para um servidor de análise e coleta. O processo é detalhado na figura 25.

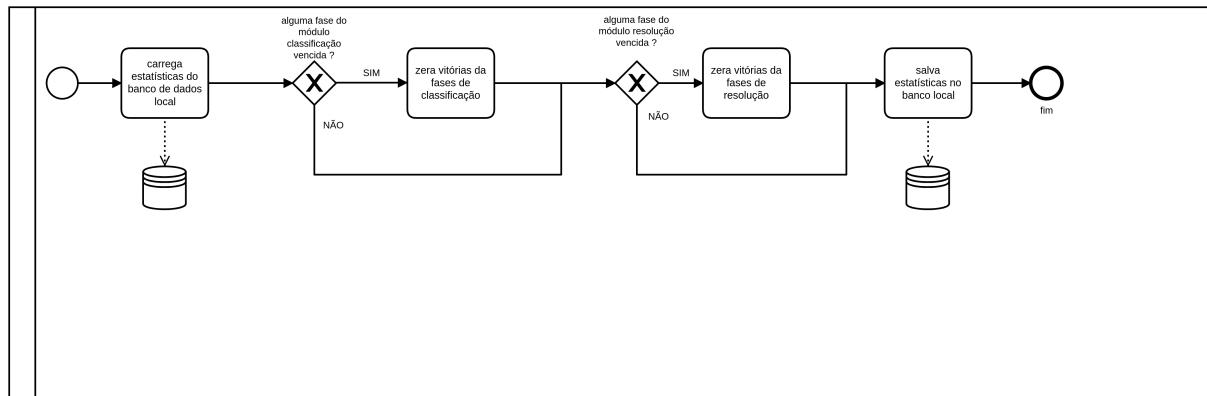
Figura 9 – Envio de estatísticas



Fonte: do próprio autor

É necessário que o jogador digite sua matrícula corretamente, pois esta será validada com a quantidade de 8 algarismos. Caso a matrícula seja validada e haja a conexão com a internet ocorrerá o envio dos dados, qualquer erro pelo caminho haverá o aviso específico ao jogador. Após a persistência dos dados no servidor, os dados locais são zerados para que não sejam redundantes no próximo envio. O processo de zerar estatísticas é apresentado na figura 10.

Figura 10 – Zerar estatísticas após envio

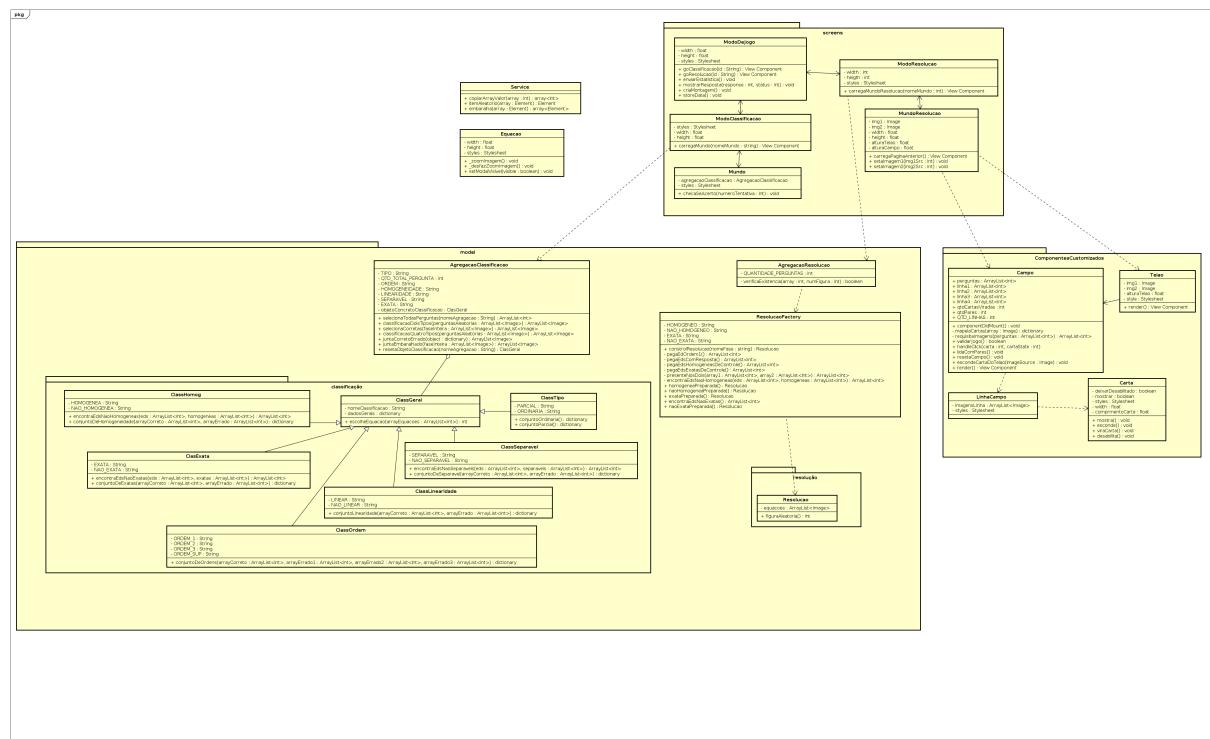


Fonte: do próprio autor

Zerar estatísticas percorre as fases de classificação e resolução verificando qual fase já foi vencida ao menos uma vez e trocando o valor atual para 0. Após o loop nos módulos os dados são persistidos na memória do celular.

Outro artefato de documentação desenvolvido foi o diagrama de classes. Este foi modelado de modo a facilitar o desenvolvimento e o entendimento da pasta src, pois deu uma guia no que precisava ser feito e como as classes do jogo e os componentes se relacionam.

Figura 11 – Diagrama de Classe do AprEnDO



Fonte: do próprio autor

A pasta src apresenta o código fonte escrito em React (node.js). A modelagem da aplicação é possível visualizar no diagrama de classe. Tem-se a pasta dos componentes customizados, a pasta model (que tem o modelo de perguntas e respostas e classes construtoras da agregação de perguntas e respostas para o jogo), a pasta screens que contém as telas de jogo, sendo que algumas screens contém componentes que estão na pasta de componentes customizados. Tem o arquivo Service.js que realiza algumas tarefas auxiliares necessárias em partes misturadas do jogo, como por exemplo embaralhar um array (conjunto de dados homogêneos), realizar uma cópia de valores de um array e escolher um item aleatório.

A seguir uma explicação das funções desenvolvidas que estão na pasta SRC.

PASTA: SRC

- ARQUIVO:Service

copiarArrayValor: recebe um vetor como parâmetro e copia apenas os seus valores sem manter a referência. Para isso criar um vetor vazio, percorre o recebido e escreve seus valores no novo vetor que será retornado.

itemAleatorio: recebe um vetor como parâmetro. Em caso de não pelo menos um elemento, nulo é retornado. Caso contrário com o apoio da biblioteca Math de matemática, um número flutuante entre 0 e 1 é escolhido e multiplicado pelo comprimento do vetor. O número é arredondado para baixo e então o item nesta posição do vetor é retornado.

embaralha: recebe um vetor e aleatoriamente troca as posições dos valores

- ARQUIVO:Equacao

_zoomImagem : função ativada quando houver um longo clique em cima de algum exemplo de equação. Esta habilita o aparecimento de uma modal para mostrar equação ampliada.

_desfazZoomImagem : função ativada quando o dedo é tirado da tela após um longo clique em cima de algum exemplo de equação. Esta desabilita o aparecimento de uma modal que mostra equação ampliada.

setModalVisivel: função chamada por _zoomImagem e _desfazZoomImagem para habilitar ou desabilitar o aparacimento da modal com equação ampliada.

PASTA:SRC/screens

- ARQUIVO:ModoDeJogo

goClassificacao: a função goClassificacao utiliza a dependência Navigation e serve para redirecionar para a tela de classificação das equações diferenciais. Recebe como parâmetro um ID que é um identificador único do tipo string referente ao nome da screen que é para redirecionar.

goResolucao: a função goResolucao utiliza a dependência Navigation importada no projeto que serve para redirecionar para a tela de resolução das equações diferenciais. Recebe como parâmetro um ID que é um identificador único do tipo string referente ao nome da screen que é para redirecionar.

enviarEstatísticas: a função enviarEstatisticas funciona de modo assíncrono e utiliza da dependência netInfo para checar seu dispositivo está conectado à internet. Caso o dispositivo não esteja, um alerta avisando ao usuário que não está conectado à Internet é acionado e retorna a página inicial da aplicação, caso haja conexão com a internet é checado se a matrícula digitada pelo usuário não é nula e ao remover os caracteres de barra de espaço verifica-se que não está vazio e que contém 8 dígitos. Em caso de matrícula inserida corretamente é acionado um loader enquanto as estatísticas locais do celular são capturados e formatadas para então realizar uma requisição ao servidor do Heroku na rota '/estatísticas'. O conteúdo JSON é formatado para string e então enviado. Após o processamento no servidor uma resposta é retornada para o aplicativo e a função enviarEstatisticas chama a função mostraResposta que tratará de mostrar a resposta adequada a depender do status de retorno do Servidor. Em caso de alguma exceção ocorrer durante todo o processamento um alerta é mostrado ao usuário indicando que houve erro com o servidor e pedindo para retornar mais tarde. O loader é desabilitado e novamente retorna-se a tela inicial da aplicação.

mostraResposta: a função mostraResposta é acionada pela função enviarEstatisticas e também assíncrona. Recebe como parâmetro a resposta do Servidor e o código de status. Caso a resposta tenha sucesso NÃO OKAY é mostrado um alerta para o usuário informando retorno não esperado e retorna-se à tela inicial. Em caso de resposta sucesso OKAY é carregado da memória local as estatísticas atuais e um loop percorre todas as fases do modo de classificação e de resolução zerando a quantidade de vezes que o jogador terminou de jogar, para que no próximo envio seja enviado apenas as variações entre o último envio e o atual. Após as estatísticas estarem zeradas, sstas são salvas na memória do celular e um alerta de estatísticas enviadas com sucesso é apresentado ao usuário, só então retorna-se a tela inicial.

: a função é acionada apenas na primeira vez o que o jogador acessa o jogo para então criar o template das estatísticas de classificação e resolução e todas as suas fases e assim é salvo na memória do celular. O tipo de dado das estatísticas é um Json.

storeData: a função storeData é acionada toda vez que um jogador entra no jogo.

É tentado recuperar da memória uma variável chamada ??, caso esta variável seja nula significa que é a primeira vez do usuário no jogo, esta variável é então criada e salva na memória com valor falso para nas outras vezes não cair neste caso, chama-se a função 4.4 para gerar o template das estatísticas. Caso a variável ?? já exista e retorne o valor de falso, significa que não é a primeira vez do usuário no jogo e nada acontece.

- ARQUIVO:ModoClassificacao

carregaMundo: função assíncrona que tem como parâmetro o nome de mundo que será criado para ser repassado ao objeto agregação classificação. Este objeto serve para a função na hora de chamar a nova tela de mundo passando as propriedades: vetor de perguntas aleatórias criadas, vetor do número das imagens corretas, vetor das fases embaralhadas, o total de questões e o nome do mundo.

- ARQUIVO:ModoResolucao

carregaMundoResolucao: função assíncrona que recebe como parâmetro o nome do mundo para criar objeto agregaçãoResolução e passar para próxima screen chamada ?? junto com a propriedade ?? que é um vetor provido pelo objeto agregaçãoResolução

- ARQUIVO:Mundo

checaSeAcerto: função assíncrona é ativada assim que alguma opção de classificação é clickada. Recebe como parâmetro o número da tentativa da rodada, que varia de 1 a 4. Caso o número da tentativa clicada seja igual ao da pergunta correta o número da perguntaAtual é incrementado e o número da quantidade de perguntasCorretas também é incrementado apenas em caso de não ser a última pergunta. Em caso de ser a última pergunta e tentativa estiver correta as estatísticas são atualizadas com o incremento de uma vitória na fase jogada do módulo de classificação. Ao vencer a fase um alerta de parabenização é mostrado e abre-se uma janela que ao ser confirmada retorna à tela de classificação. No caso de tentativa incorreta é mostrado um alerta de tente novamente e retorna-se à questão pendente.

- ARQUIVO:MundoResolucao

carregaPaginaAnterior: utiliza a dependência Navigation e apenas fecha a página atual que a página anterior já está na pilha e passa então a ser a atual.

PASTA:SRC/model

PASTA:SRC/ComponentesCustomizados

- ARQUIVO:Campo

mapeiaCartas: recebe o array retornado da requisitaImagens o qual inicia-se com uma pergunta e sucessivamente sua solução. Um loop é feito no array para criar um dicionário com chaves as perguntas e valor as respostas. Por isso o loop é iterado apenas nas perguntas, que são as posições pares. Ao final da iteração e população do dicionário, o mesmo é retornado.

requisitaImagens: recebe um array com o número das figuras aleatórias sorteadas para esta partida e faz o import através da função require procurando no arquivo index das perguntas e das respostas. O retorno é um array com 20 figuras, sendo 10 perguntas e 10 soluções das perguntas. OBS: o array de retorno é ordenado com uma pergunta e a sua solução sucessivamente.

validarJogo: é chamada sempre que duas cartas estão viradas para cima. A função serve para decidir se duas cartas formam um par. A verificação ocorre comparando com o array de cartas mapeado chave e valor. O procedimento checa se a primeira carta virada é chave ou valor. Se for chave, então compara a segunda carta com o valor, se for valor, procura-se então a chave para comparar com a segunda carta para assim confirmar se a dupla forma um par gêmeo ou não. Em caso de ser um par, pela regra do jogo é decrementado uma unidade dos pares restantes, se for o último par já acontece a atualização das estatísticas de qual fase do módulo resolução foi vencido e a parabenização ao usuário com retorno para a tela de escolha de fase. Se não for o último par, a função apenas retorna informando se as cartas atuais formam um par ou não.

handleClick: é acionada quando ocorre o click em qualquer carta enviando a informação se a carta é para ser virada ou escondida (true ou false). Se for um click para mostrar a carta, então a variável quantidadeDeCartasViradas é incrementado em 1. OBS: Se esta for a única carta virada então na posição reservada para a imagem 1 será renderizada a equação da carta OBS1: Se for a segunda carta virada, a carta será desenhada no espaço reservado para a imagem 2 e será aplicado o procedimento do jogo da memória para validar se as 2 cartas foram um par.

Se for um click para não mostrar a carta é acionada a função para esconder a carta do telão (espaço reservado para renderização das equações), e se o valor da variável não cair em nenhum dos casos previstos um novo erro é lançado.

lidaComPares: é chamada pelo handleClick deste mesmo arquivo. Recebe como parâmetro um booleano que representa se as duas cartas viradas para cima formam um par ou não. Em caso de serem um par, as cartas tem a função de click desabilitadas, e são travadas viradas pra cima com a imagem da equação sendo mostrada. Em caso de não serem um par as duas são viradas para o lado inverso escondendo o seu conteúdo.

resetaCampo: esta função limpa as 2 equações renderizadas no telão, altera o estado de quantidadeDeCartasViradas para 0 e limpa o vetor de 2 posições que armazena as cartas viradas no momento.

escondeCartaDoTelao: recebe como parâmetro a imgSrc da imagem a ser escondida. Para isso também é necessário decrementar a quantidade de cartas viradas em uma unidade e então checar qual das duas cartas do telão que tem a imgSrc igual ao que foi passado para esta função, para então poder setá-lo como nulo.

- ARQUIVO:Telao
- ARQUIVO:LinhaCampo

encaminha a função handle para cada carta do campo assim como o conteúdo da carta também, ou seja, a sua imagem, seja uma carta de pergunta, ou de resposta.

- ARQUIVO:Carta

mostra: alterna o estado da variável mostrar e troca o valor da animação de 0 graus para 180 graus, realizando o efeito de virar a carta.

esconde: alterna o estado da variável mostrar e troca o valor da animação de 180 graus para 0 graus, realizando o efeito de virar a carta e mostrar o fundo.

viraCarta: checa o estado da variável mostrar para decidir se chama a função mostra ou esconde e chama a função handle que lida com as regras do jogo da memória.

desabilita: troca o estado da variável deixarDesabilitado, a qual é chamada apenas quanto um par de cartas gêmeas é encontrado. Desse modo impede que a carta seja rotacionada desse momento para frente, deixando-a sempre virada com a parte do conteúdo para cima, porém desabilitada para cliques.

4.5 Servidor aprEnDO

O servidor aprendo hospedado no heroku serve para receber estatísticas dos aplicativos enviados pelos alunos. Os alunos para enviar a estatística precisam informar o número da matrícula. As informações são enviadas em formato JSON e armazenadas no banco de dados integrado com o servidor.

O jogo aprEnDO além do aplicativo de celular tem um servidor para contabilizar dados do jogo. O servidor está hospedado no heroku utilizando uma contra free que esteve disponível durante a fase de teste dos alunos, apesar de gratuito comparado com

a necessidade que era necessário deu para suprir todas as expectativas. Junto com a aplicação foi adicionado um plugin de banco de dados mongodb. Este banco armazenou em um documento os metadados do jogo. O mongodb é um add-on no mLab que permite integração com o heroku, ele permite uma conta temporária com uma quantidade limitada de capacidade de dados, mas para o necessário que são apenas arquivo json. O único problema é que o mLab não recomenda utilizar a conta free para a produção por não produzir replicação, porém o banco estava sendo acompanhado muitas vezes por dia para coletar os dados recebidos dos alunos. O servidor foi escrito em node.js. O script index.js do servidor cria uma rota /estatísticas que é a utilizada pelos aplicativos para enviar a requisição POST.

O servidor está hospedado no github do heroku, pela interface da linha de comando (CLI) pode ser clonado o projeto com o comando 'heroku git:clone -a servidor-aprendo'. Para desenvolvimento foi utilizado uma única branch, que foi o necessário. Mais branches seriam criadas, caso fosse necessário. Tendo em vista que o servidor deveria estar pronto até o dia 10 de maio, que era o início da data de aplicação do jogo com os alunos, até esta data o servidor estava sendo desenvolvido e testado, o único problema crucial que poderia ocorrer seria não realizar os commits dos códigos adicionados e removidos. Assim que o servidor estivesse pronto, a expectativa era não precisar alterar mais código nele, apenas se o servidor quebrasse por algum caso inesperado, aí sim o servidor teria de entrar no ar novamente e novas branches de correção seriam criadas para não impedir o funcionamento parcial do servidor. Porém como os logs do servidor estavam sendo acompanhados muitas vezes por dia durante o período de aplicação e o servidor não caiu nenhuma vez, não foi necessário criar branches adicionais para a correção de erros.

Ao receber os dados json enviados do aplicativo do celular para o servidor, o mesmo concatena todos os dados que chegam em uma variável apenas. Ao terminar a leitura dos dados de chegada eles são formatados e escrito uma resposta de sucesso verdadeiro, a função 'lê histórico' é chamada e a resposta é encerrada.

A função 'lê histórico' recupera o documento existente no banco de dados heroku_41w86511 hospedado no mLab, da coleção chamada histórico. A função analisa checa se a matrícula recebida pela requisição já havia alguma vez enviado estatísticas ou se é uma matrícula nova. Em caso de matrícula nova, uma nova linha é adicionada com chave primária a matrícula do aluno e salva no banco. Em caso de matrícula existente é iterado sobre classificação e resolução e todas suas fases e incrementando a quantidade de vitórias a mais que foi concluído das fases no jogo e em seguida o arquivo é atualizado no banco do mLab.

4.6 Acesso ao código

O código do servidor está hospedado no github do heroku, disponível em <<https://dashboard.heroku.com/apps/servidor-aprendo>>.

Todo o código está hospedado no github. O banco de equações e a aplicação para android estão disponíveis no seguinte link <<https://github.com/LeonardoRk/TCC-2>>.

O APP publicado no google play pode ser encontrado através do link: <<https://play.google.com/store/apps/details?id=projeto.aprEnDO&rdid=projeto.aprEnDO>>

E a biblioteca do Wolfram Alpha em javascript pode ser baixada no link: <<https://products.wolframalpha.com/api/libraries.html>>. Porém a biblioteca já está inclusa no github dentro da pasta banco.

4.7 Empacotamento

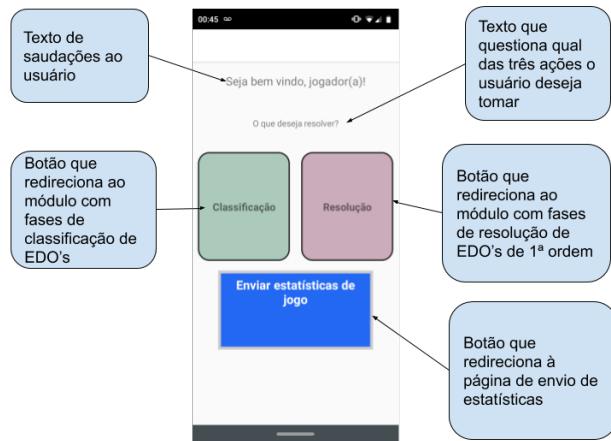
O jogo é empacotado para criar um arquivo .apk, e este que é instalado nos celulares. Para submeter o jogo ao Google Play para os jogadores poderem baixá-lo é necessário criar o .apk. O mesmo só é criado sempre que tem alguma nova atualização no jogo, seja no banco de equações ou manutenção corretiva ou evolutiva do jogo.

O empacotamento do jogo ocorre dentro da pasta do projeto react native. É utilizado o comando 'npm run android'.

5 Manual do aprEnDO

O jogo é composto em 2 pacotes, o pacote de exercícios e o pacote de estatísticas. É na página inicial que você pode escolher o que deseja fazer.

Figura 12 – Tela inicial



Fonte: do próprio autor

5.1 Pacote 1

No pacote de exercícios temos 2 módulos, o de classificação e o de resolução.

5.1.1 Módulo 1

O módulo de classificação objetiva fixar o reconhecimento e classificação de EDs. Este é composto de 6 fases: tipo, ordem, homogeneidade, linearidade, separável e exata respectivamente. Essas fases foram classificadas como sendo da mais fácil para a mais complexa. Para navegar entre as fases basta arrastar a tela para a direita. Cada fase tem o domínio das suas perguntas e o número fixo de 20 equações que são selecionadas aleatoriamente de acordo com a fase.

As perguntas de tipo são:

- Escolha a ED ordinária
- Escolha a ED parcial

As perguntas de ordem são:

- Escolha a ED de ordem 1

- Escolha a ED de ordem 2
- Escolha a ED de ordem 3
- Escolha a ED de ordem superior

As perguntas de homogeneidade são:

- Escolha a ED homogênea
- Escolha a ED NÃO homogênea

As perguntas de linearidade são:

- Escolha a ED linear
- Escolha a ED NÃO linear

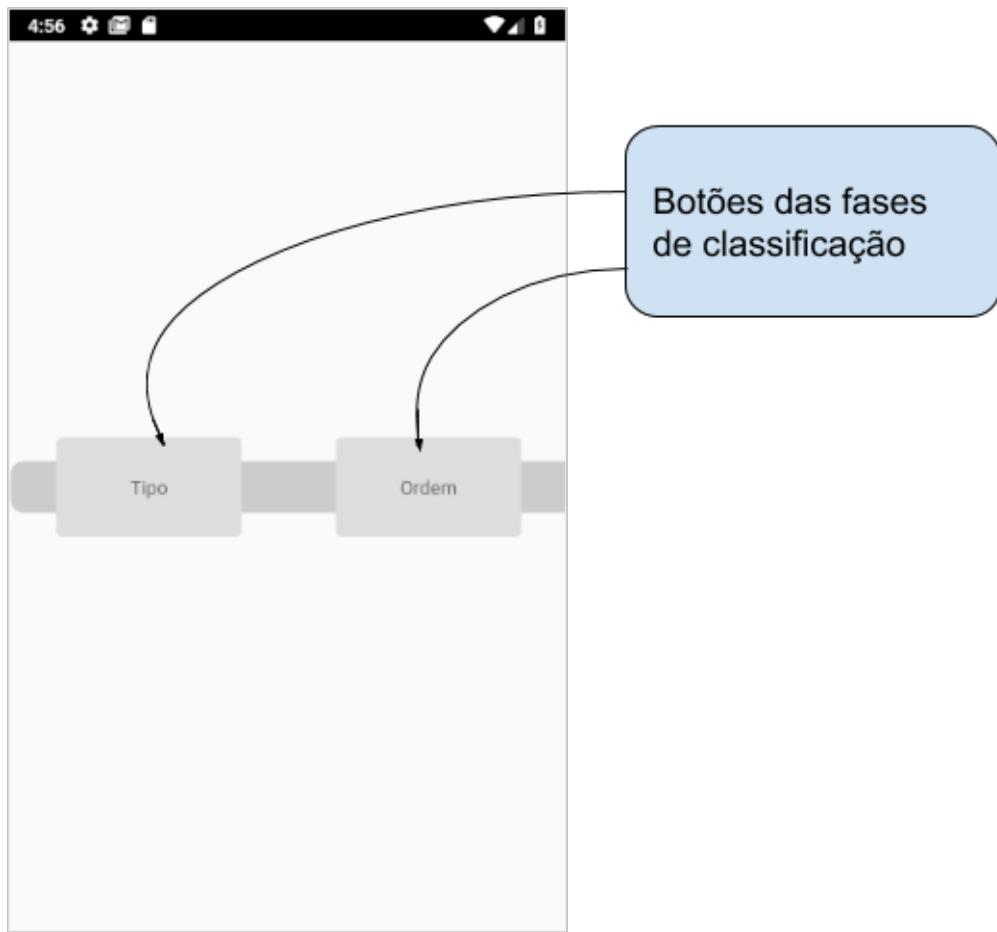
As perguntas para a fase separável são:

- Escolha a ED separável
- Escolha a ED NÃO separável

As perguntas se exata são:

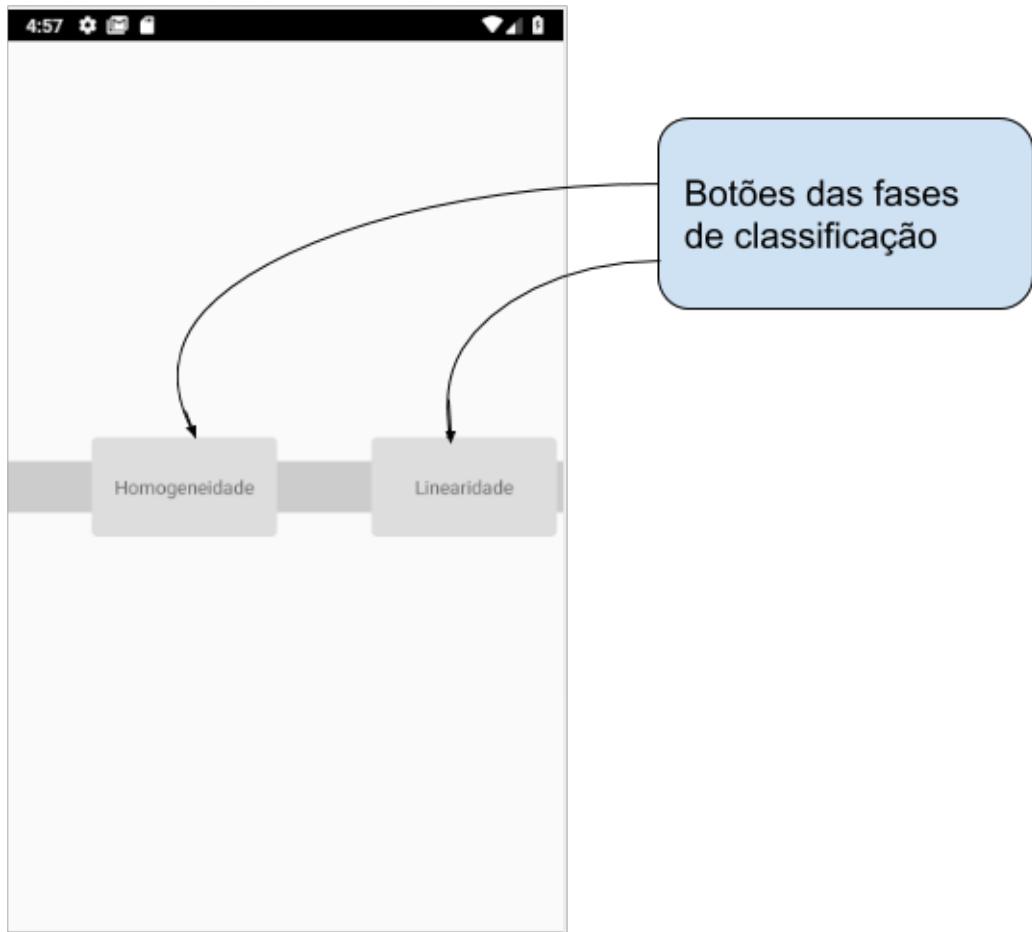
- Escolha a ED exata
- Escolha a ED NÃO exata

Figura 13 – Modo de classificação parte 1



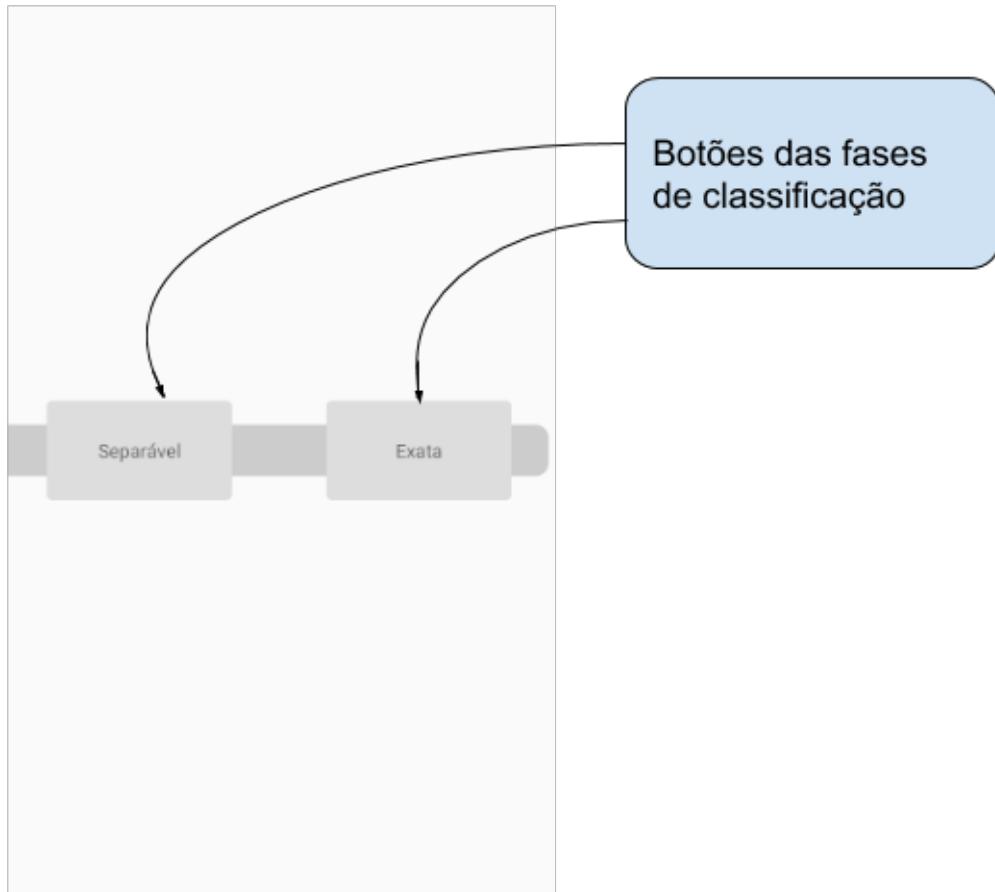
Fonte: do próprio autor

Figura 14 – Modo de classificação parte 2



Fonte: do próprio autor

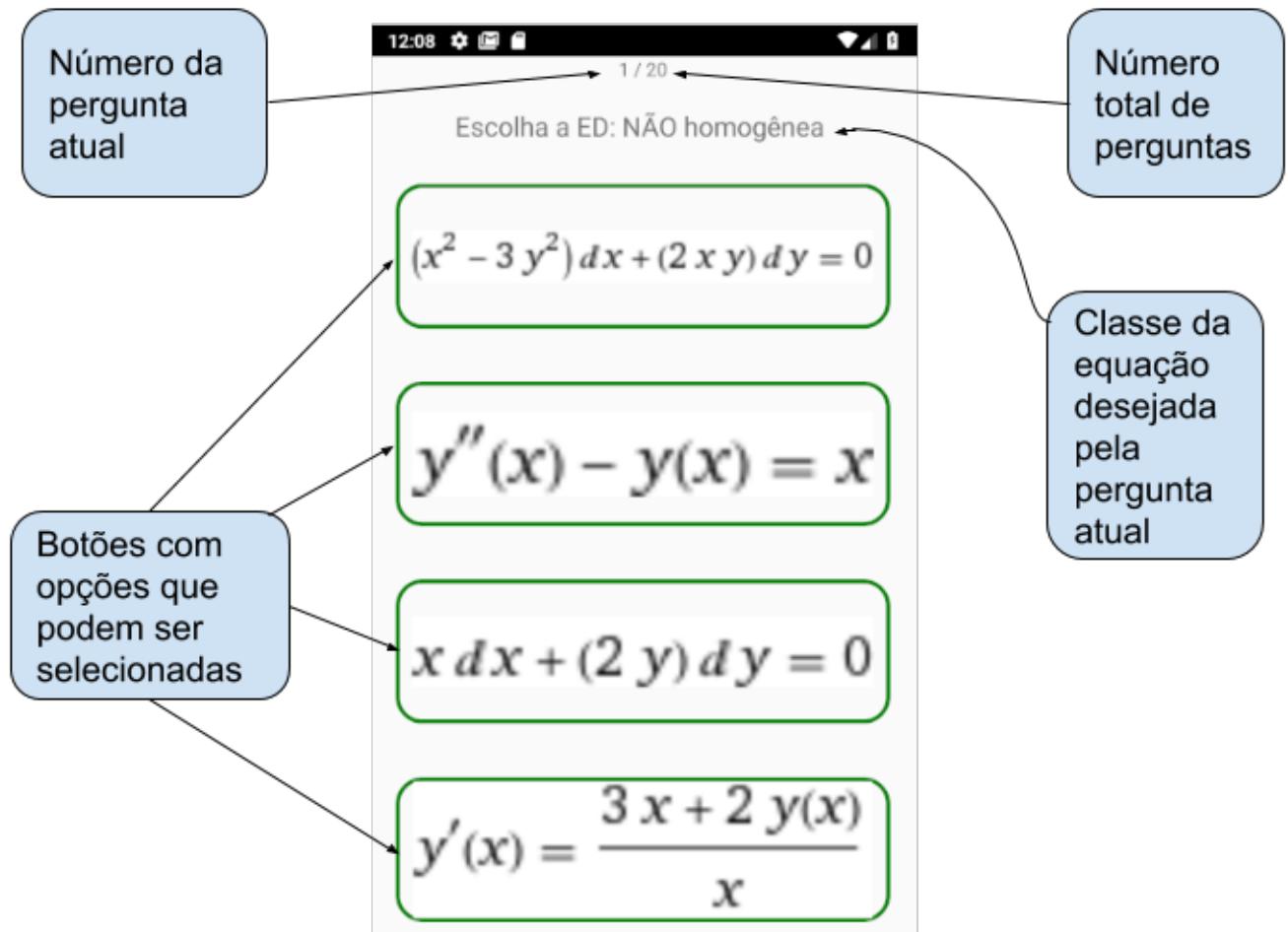
Figura 15 – Modo de classificação parte 3



Fonte: do próprio autor

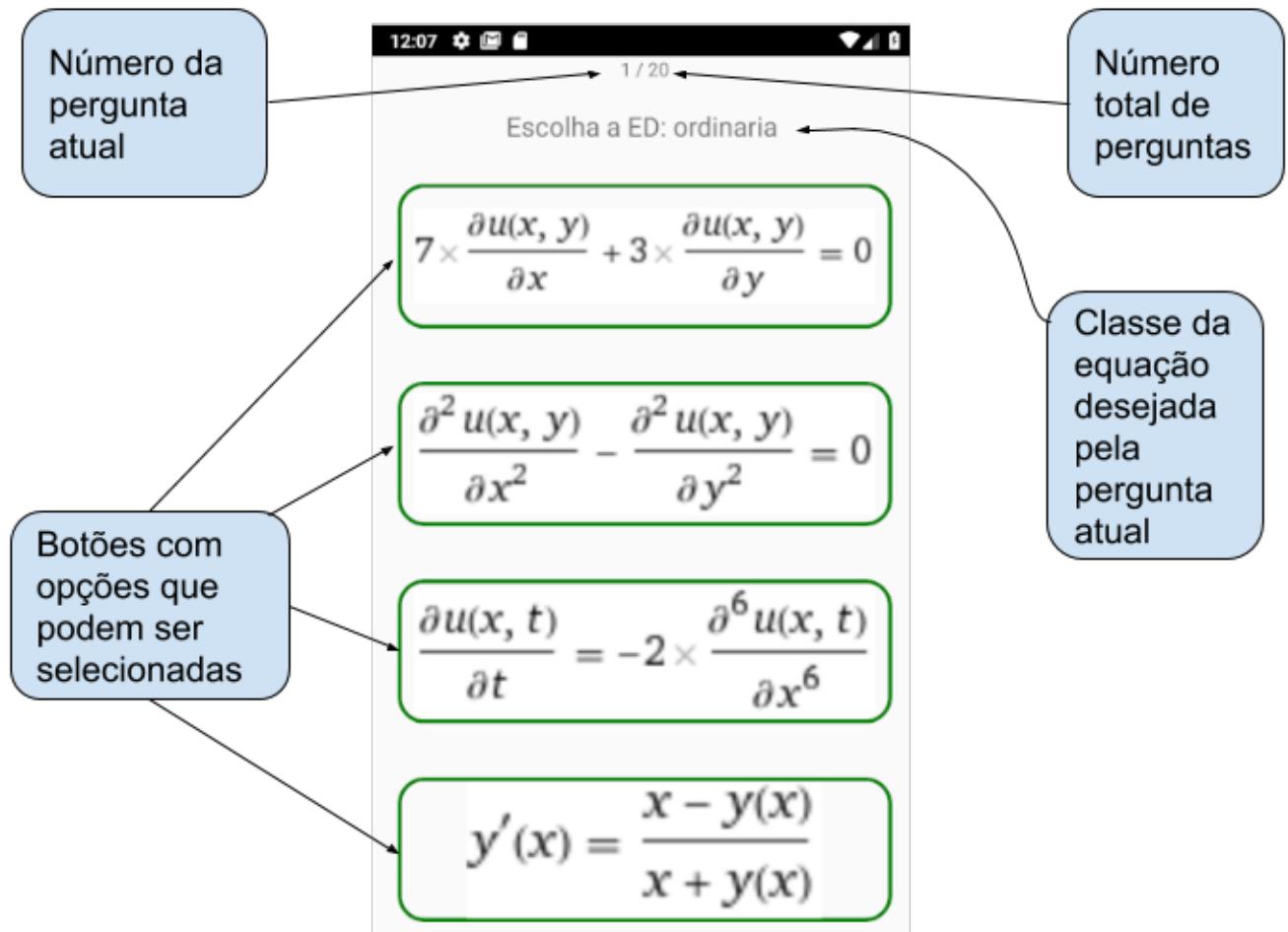
As fases de classificação apresentam 4 equações como opção sendo que apenas 1 é a correta. Ao pressionar cada opção por um tempo aparecerá a imagem da equação em uma janela modal para tentar melhorar a visualização. Para escolher uma opção basta dar um toque na opção desejada. Em caso de acerto da equação carregará automaticamente a próxima pergunta. Em caso de erro aparecerá o *feedback* da escolha da opção inválida. Ao fim da fase será redirecionado novamente para o modo de classificação, após a parabenização do jogador, onde é possível escolher outra fase para jogar ou voltar para trocar o módulo.

Figura 16 – Exemplo da fase de classificação de homogeneidade



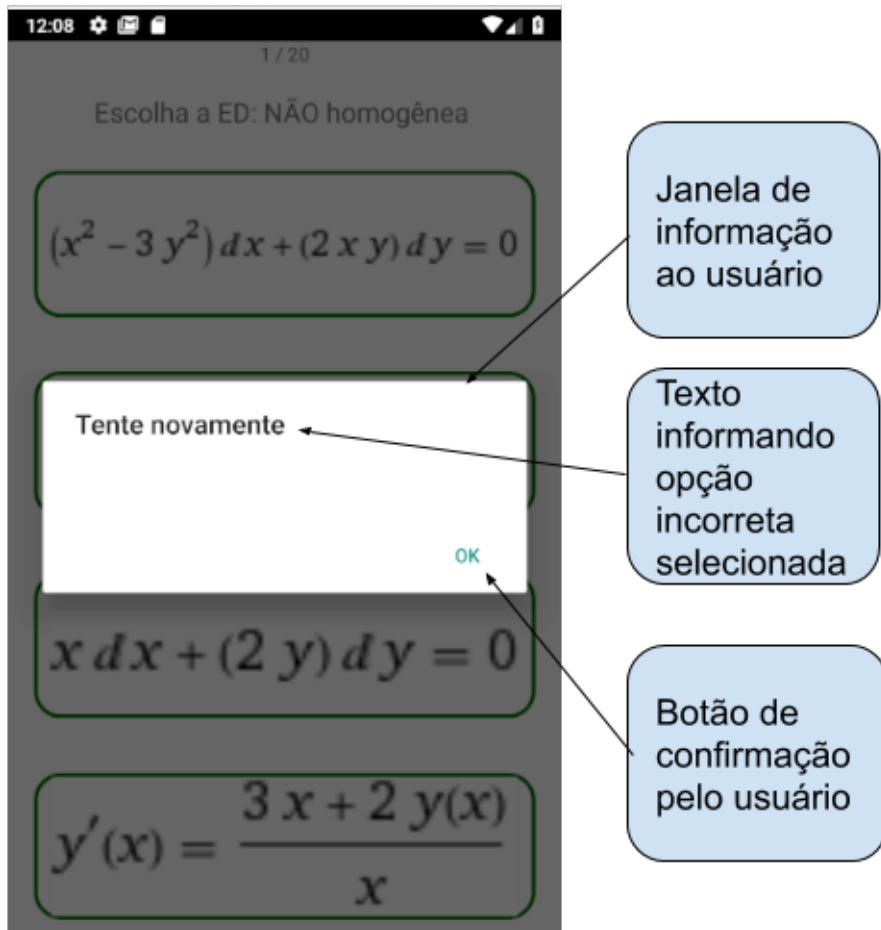
Fonte: do próprio autor

Figura 17 – Exemplo da fase de classificação de tipo



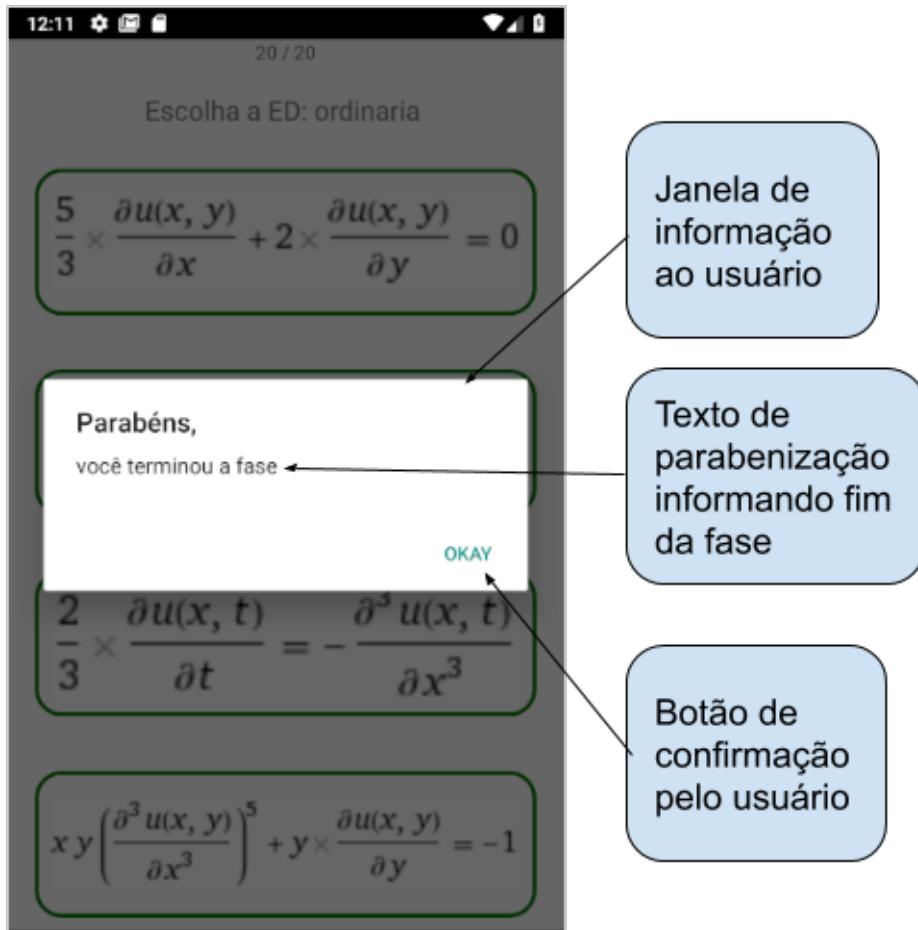
Fonte: do próprio autor

Figura 18 – Feedback fornecido à uma tentativa inválida



Fonte: do próprio autor

Figura 19 – Tela de finalização do modo de classificação

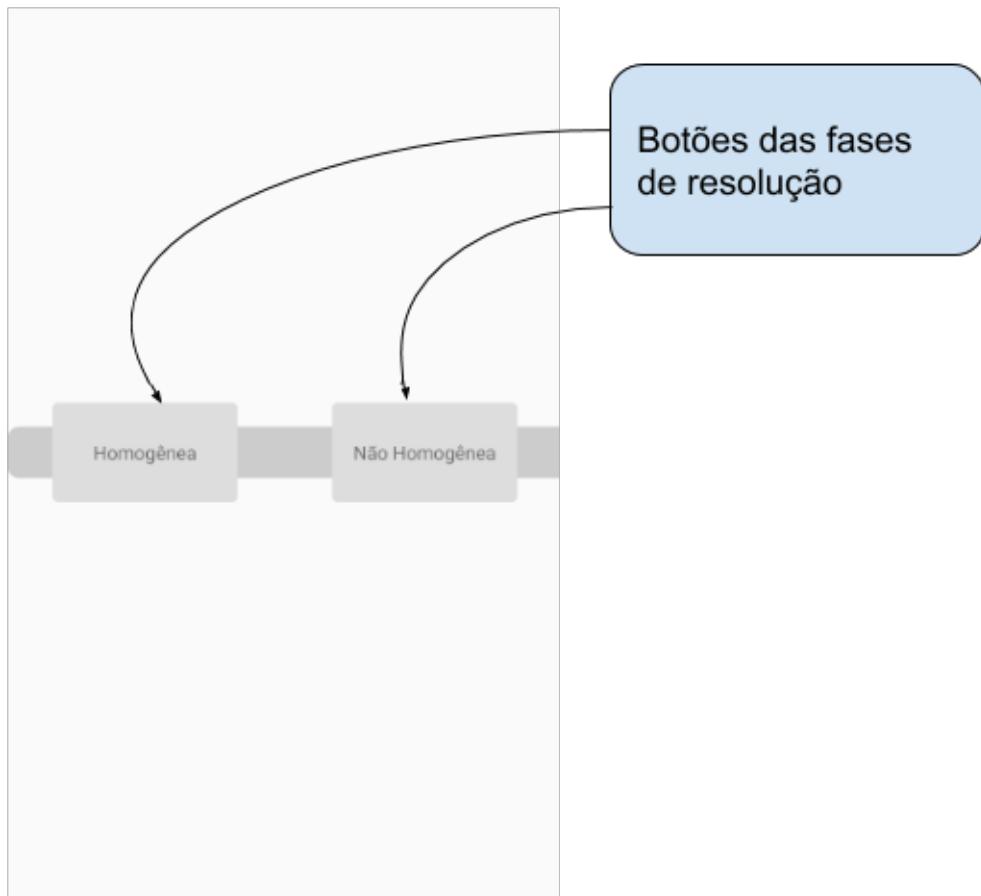


Fonte: do próprio autor

5.1.2 Módulo 2

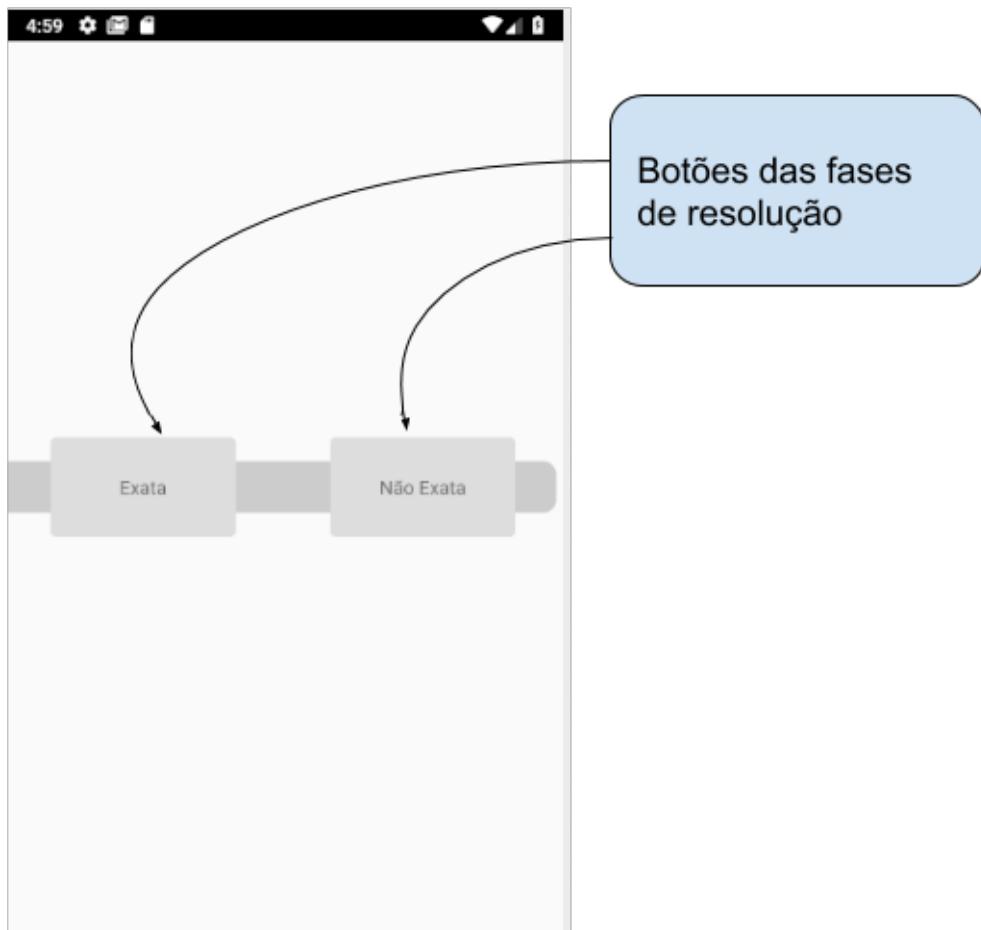
O módulo de resolução visa fixar o conhecimento de resolução de EDOs de 1^a e tem 4 fases. São elas homogênea, NÃO homogênea, exatas e NÃO exatas. Neste módulo estão presentes APENAS equações diferenciais ordinárias de primeira ordem que contenham respostas.

Figura 20 – Modo de resolução parte 1



Fonte: do próprio autor

Figura 21 – Modo de resolução parte 2

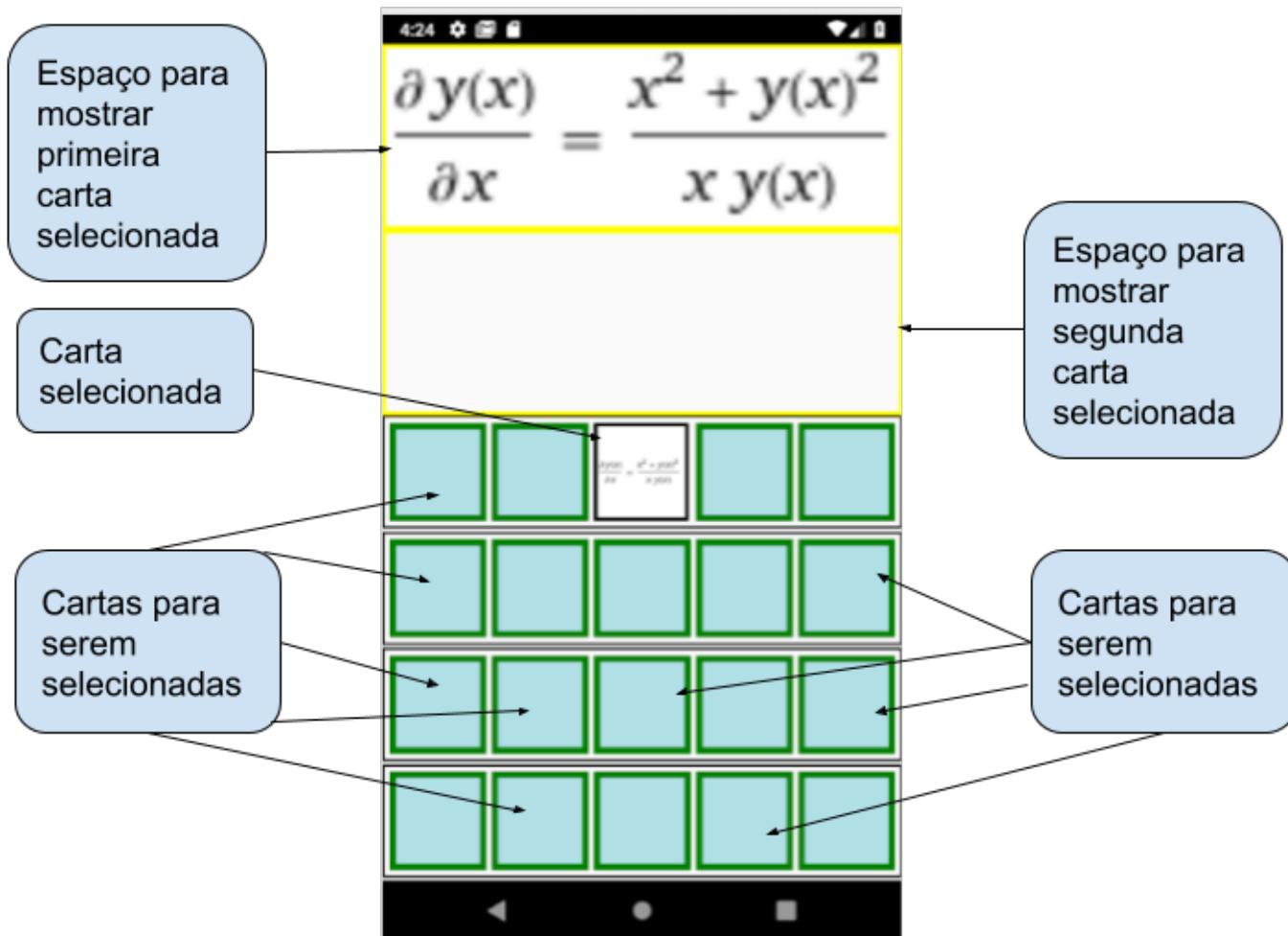


Fonte: do próprio autor

O módulo de resolução trata-se de um jogo da memória, o qual tem o objetivo de encontrar as cartas gêmeas. Uma carta contém a EDO proposta e a carta gêmea contém a solução correta para a equação proposta.

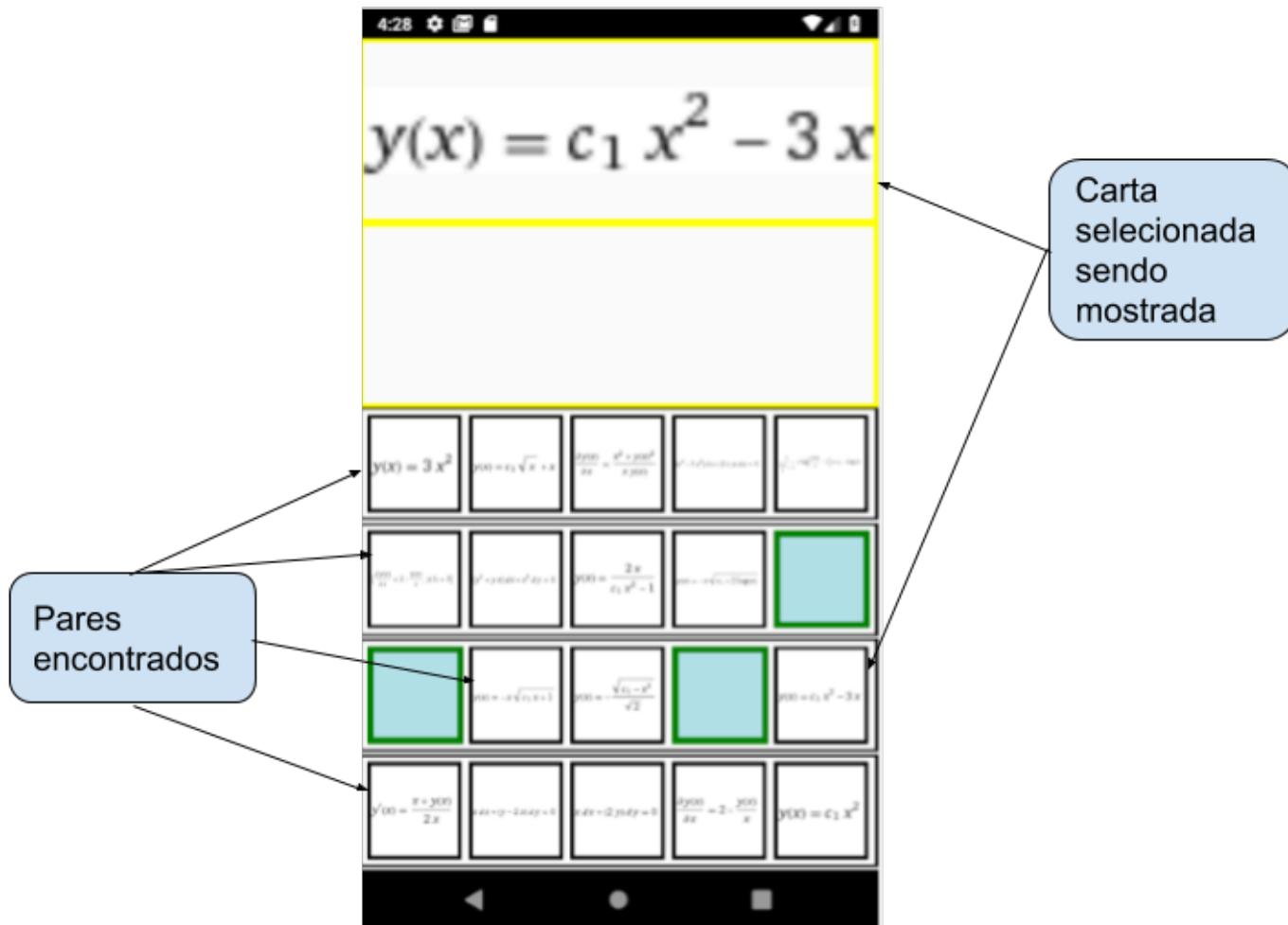
Existem 2 campos reservados na tela para mostrar as cartas selecionadas. O jogo é composto de 20 cartas, 10 EDOs e 10 soluções de EDO. Cada carta é selecionada com um toque sobre ela, onde sua face será mostrada e seu conteúdo se mostrará em 1 dos 2 campos reservados. Cada clique inverte o lado da carta que está sendo mostrado. Ao selecionar 2 cartas haverá a comparação se são complementares (ou seja, pergunta e resposta). Em caso afirmativo, as cartas congelarão não sendo mais possível interagir com elas, em caso negativo apenas esconderão sua face para que inicie outra tentativa do jogador.

Figura 22 – Jogo da memória com 1 carta selecionada



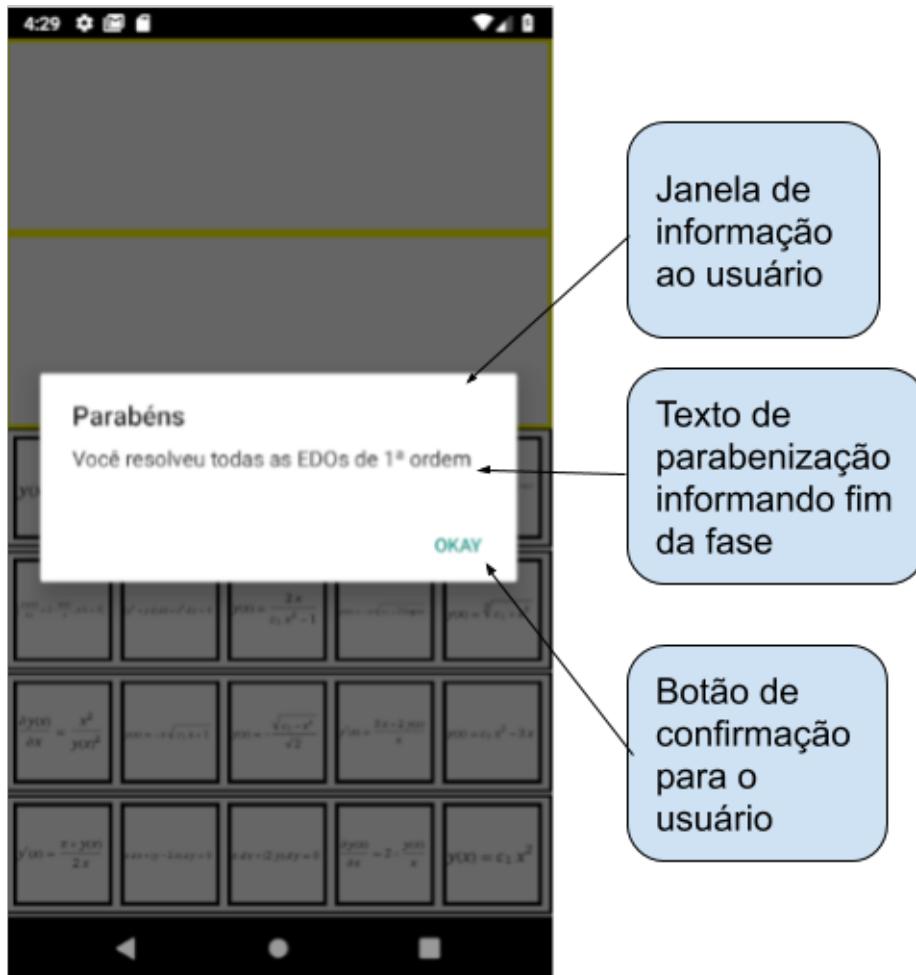
Fonte: do próprio autor

Figura 23 – Jogo da memória com alguns pares encontrados



Fonte: do próprio autor

Figura 24 – Vitória de jogo da memória



Fonte: do próprio autor

Ao terminar qualquer fase do jogo da memória, após a confirmação o jogo retornará à tela de seleção da fase a ser jogada.

5.2 Pacote 2

O pacote de estatísticas está relacionado à coleta de dados para a avaliação da quantidade de jogadores que usaram o jogos e dentre os que usaram, quais fases eles completaram e quantas vezes eles as completaram. O envio das estatísticas para o servidor não ocorrerá sem a ação do usuário, depende do jogador enviar os dados de acordo com a sua vontade de colaboração através de um botão na tela inicial e a confirmação do envio com a sua matrícula.

Ao se clicar no botão para enviar estatísticas do jogo, deve-se redirecionar para a tela igual a figura 25.

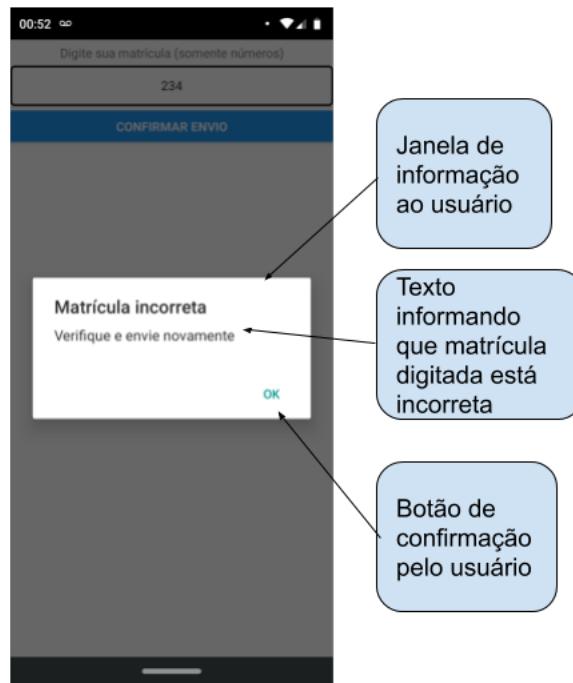
Figura 25 – Tela de envio de estatísticas



Fonte: do próprio autor

A figura 26 apresenta a mensagem de erro ao digitar a matrícula incorreta. A checagem é feita conferindo a quantidade de dígitos que deve ser igual a 8.

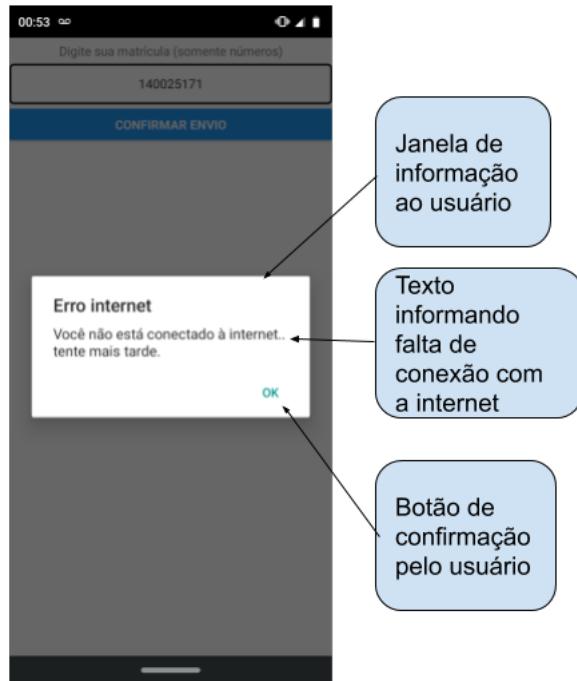
Figura 26 – Alerta de matrícula inválida



Fonte: do próprio autor

A figura 27 apresenta a mensagem de erro quando não há conexão do celular com a internet para o envio dos dados ao servidor.

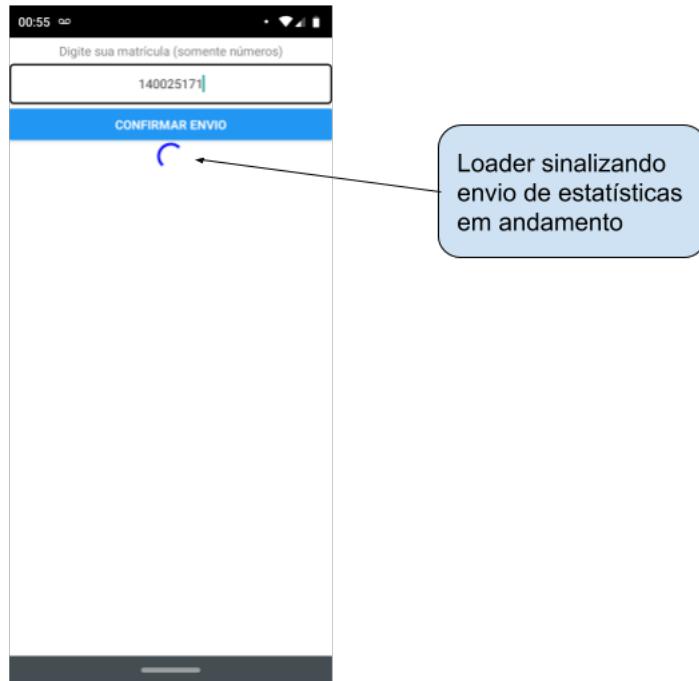
Figura 27 – Alerta de falta de internet



Fonte: do próprio autor

Ao ser digitado a matrícula correta e realizada a submissão do envio das estatísticas, um loader aparecerá enquanto o procedimento não termina, como mostra a figura 28 e ao terminar o envio, uma mensagem é apresentada, como mostra a figura 29

Figura 28 – Loader visível enquanto envia estatísticas



Fonte: do próprio autor

Figura 29 – Alerta de envio com sucesso



Fonte: do próprio autor

Ao fim do envio, é redirecionado para à tela inicial [12](#).

6 Análise

De X alunos na turma, até o momento da última checagem no servidor de dados do heroku, contabilizou-se 13 alunos que enviaram estatísticas, porém não se sabe quantos jogaram. Sabe-se que dos 13 alunos que enviaram, 10 completaram pelo menos uma fase do módulo de classificação e 3 pessoas enviaram as estatísticas sem ter completado nenhuma fase de nenhum módulo. Não se sabe quantas vezes eles tentaram jogar as fases e nem se eles apenas enviaram as estatísticas sem tentar.

A figura 30 apresenta o resultado de classificação dos 13 alunos que contribuíram enviando os dados. Onde está escrito falso significa que não foi vencido nenhuma vez a fase (maioria dos casos). E alguns alunos jogaram apenas 1 vez, outros 2 e um aluno jogou 8 vezes.

Figura 30 – Estatísticas colhidas de classificação

Alunos	C_exata	C_homogeneidade	C_linearidade	C_ordem	C_separável	C_tipo
1	1	1	FALSO	1	FALSO	1
2	1	1	FALSO	1	FALSO	1
3	1	FALSO	FALSO	1	FALSO	1
4	FALSO	1	1	1	1	2
5	FALSO	FALSO	FALSO	FALSO	FALSO	FALSO
6	FALSO	FALSO	FALSO	FALSO	FALSO	1
7	FALSO	FALSO	FALSO	1	FALSO	FALSO
8	FALSO	1	1	1	FALSO	1
9	FALSO	FALSO	FALSO	FALSO	FALSO	FALSO
10	FALSO	FALSO	FALSO	FALSO	FALSO	FALSO
11	FALSO	FALSO	FALSO	1	FALSO	1
12	FALSO	FALSO	FALSO	1	FALSO	1
13	FALSO	FALSO	FALSO	8	FALSO	FALSO

A figura 31 apresenta o resultado de resolução dos 13 alunos que contribuíram enviando os dados. Onde está escrito falso significa que não foi vencido nenhuma vez a fase (maioria dos casos). E alguns alunos jogaram apenas 1 vez algumas fases, porém a minoria venceu este módulo, percebe-se que sua aceitação não foi muito boa por algum motivo. Apenas 3 alunos jogaram pelo menos 1 vez 1 fase.

Figura 31 – Estatísticas colhidas de resolução

Alunos	R_homogêneo	R_não_homogêneo	R_exato	R_não_exato
1	FALSO	FALSO	FALSO	FALSO
2	FALSO	FALSO	FALSO	FALSO
3	1	FALSO	1	FALSO
4	FALSO	FALSO	FALSO	FALSO
5	FALSO	FALSO	FALSO	FALSO
6	FALSO	FALSO	FALSO	FALSO
7	1	FALSO	FALSO	FALSO
8	FALSO	FALSO	1	FALSO
9	FALSO	FALSO	FALSO	FALSO
10	FALSO	FALSO	FALSO	FALSO
11	FALSO	FALSO	FALSO	FALSO
12	FALSO	FALSO	FALSO	FALSO
13	FALSO	FALSO	FALSO	FALSO

Foi planejado coletar mais dados do servidor, porém por falta de organização do tempo não foi possível coletar todos os dados conforme o planejado.

Foram planejados desenvolver casos de teste, porém também com o mau planejamento de tempo, a grande dificuldade em desenvolver e executar os testes unitários e de integração automatizados estes acabaram não sendo desenvolvidos.

Abaixo segue uma breve descrição dos testes que foram planejados testar:

Testes de servidor no heroku:

- Garantir que cada matrícula seja única. **OBS:** Apesar do teste não ter sido implementado, existe a checagem na própria função do arquivo para garantir que a matrícula é nula. Foram realizados testes manuais para confirmar que a funcionalidade está funcionando, porém não há testes escritos.
- Garantir que os dados recebidos sejam salvos no servidor em arquivo. **OBS 1:** inicialmente acreditava-se que salvaria os dados no heroku em arquivo, mas percebeu-se que a cada novo envio uma instancia era iniciada e os dados antigos não eram persistidos, a ideia foi migrar para a utilização de um banco de dados na nuvem com a ajuda do mLab. **OBS 2:** Apesar do teste não ter sido implementado, garante-se que os dados recebidos estão sendo salvos no servidor, porém não foram desenvolvidos testes para melhores confirmações e indicações de falhas.

Testes de banco de equações:

- Garantir que toda resposta tenha uma pergunta **OBS:** no próprio desenvolvimento do código da aplicação existe o filtro de selecionar apenas equações com respostas, porém não há testes para a confirmação que o mesmo está funcionando sempre de acordo com o esperado, apesar das muitas validações manuais.

Testes da aplicação react-native:

Pacote 1

- Garantir que carregue todas fases do módulo classificação. **OBS:** apenas testes manuais e exaustivos foram realizados.
- Garantir que carregue todas fases do módulo resolução. **OBS:** apenas testes manuais e exaustivos foram realizados.
- Testar que quantidade de perguntas de classificação é 20. **OBS:** apenas testes manuais foram realizados.
- Testar que quantidade de cartas de resolução sejam 20. **OBS:** apenas testes manuais foram realizados.
- Testar que jogo da memória tenha apenas 10 equações perguntas. **OBS:** apenas testes manuais foram realizados.
- Testar que jogo da memória tenha apenas 10 equações respostas. **OBS:** apenas testes manuais foram realizados.
- Testar que cada pergunta do jogo tenha 1 e apenas uma resposta. **OBS:** existe apenas o filtro no código da aplicação sem testes escritos.
- Testar que apenas duas cartas estejam viradas para cima mostrando seu conteúdo. **OBS:** apenas testes manuais foram realizados.
- Testar em classificação que cada pergunta só tenha uma resposta correta. **OBS:** apenas testes manuais e exaustivos foram realizados.
- Testar em classificação que cada pergunta tenham 4 alternativas diferentes. **OBS:** apenas testes manuais e exaustivos foram realizados.

Pacote 2

- Garantir que ao enviar estatística a quantidade de vezes jogado zere para o módulo 1 e 2. **OBS:** apenas testes manuais e exaustivos foram realizados.
- Garantir que ao enviar estatística o tempo total de cada fase zere para o módulo 1 e 2. **OBS:** Não foi coletado o tempo jogado de cada fase.
- Garantir que ao enviar estatísticas ao servidor a informação a respeito das fases completas nos módulos permaneça a mesma e zere apenas quantas vezes foi vencido. **OBS:** apenas testes manuais e exaustivos foram realizados.

Não foi medido o tempo total gasto com o desenvolvimento dos códigos e nem o custo. O único custo que o projeto teve foi a licença da google play para poder publicar o jogo. O valor da licença foi de 25 dólares, aproximadamente 110 reais.

7 Conclusão

Para trabalhos futuros podem ser pensadas em mais funcionalidades para o jogo, como a criação de um personagem e ganho de itens para utilizações no jogo.

Poderiam ter sido feitos mais testes, os testes poderiam ter sido feitos junto com as funcionalidades Poderia ter tido um planejamento melhor com o levantamento das funcionalidades do jogo e a descrição dos requisitos. O código pode estar mais limpo e modularizado Os diagramas podem estar mais completos O jogo poderia ter sido pensado melhor com mais detalhes em fases e animações

Referências

- COELHO, V. M. **O jogo como prática pedagógica na escola inclusiva.** 2010. Acessado em: 22/10/2018. Disponível em: <https://repositorio.ufsm.br/bitstream/handle/1/1485/Coelho_Vania_Maria.pdf?sequence=1>. Citado na página 11.
- DELAMARO, M.; MALDONADO, J.; JINO, M. Introdução ao teste de software. Rio de Janeiro, 2016. Citado na página 14.
- DICHEVA, D. *et al.* Gamification in education: A systematic mapping study. **Educational Technology & Society**, v. 18, p. 75–88, 07 2015. Citado na página 8.
- DUPAUL, G. J.; STONER, G. **TDAH nas escolas - Estratégias de Avaliação e Intervenção.** 1^a edição. ed. São Paulo: M. Books do Brasil Editora Ltda, 2007. 130 p. ISBN 978-85-7680-017-0. Citado na página 11.
- ESTADÃO. **Brasil é um dos piores em qualidade de matemática e ciências.** 2016. Accessado em: 20/10/2018. Disponível em: <<https://educacao.estadao.com.br/noticias/geral,brasil-e-um-dos-piores-em-qualidade-de-ensino-de-matematica-e-ciencias,10000061150>>. Citado na página 10.
- FILHO, A. M. D. S. Software everywhere: sobre a demanda de software e da engenharia de software. 2015. ISSN 1519-6186. Disponível em: <<http://periodicos.uem.br/ojs/index.php/EspacoAcademico/article/view/29122/15124>>. Citado na página 13.
- GIACINTI, M. *et al.* A video game based on elementary differential equations. **Scientific Research**, 2013. Citado na página 8.
- IEEE. Ieee standard glossary of software engineering terminology. 1990. Disponível em: <<https://ieeexplore-ieee-org.ez54.periodicos.capes.gov.br/stamp/stamp.jsp?tp=&arnumber=159342>>. Citado na página 13.
- INEP. **Resultados de Leitura e Matemática - equipe nacional.** 2015. 27-29 p. Accessado em: 26/10/2018. Disponível em: <http://download.inep.gov.br/acoes_internacionais/pisa/resultados/2015/pisa_apresentacao_leitura_e_matematica.pptx>. Citado 2 vezes nas páginas 7 e 10.
- INEP. **Resultados de Leitura e Matemática - equipe nacional.** 2015. Accessado em: 20/10/2018. Disponível em: <http://download.inep.gov.br/acoes_internacionais/pisa/resultados/2015/pisa_apresentacao_leitura_e_matematica.pptx>. Citado na página 10.
- MONSALVE, E. S.; WERNECK, V. M. B.; CESAR, J. Simules-w : Um jogo para o ensino de engenharia de software. **Anais do III Fórum de Educação em Engenharia de Software**, p. 17–26, 2010. Citado 2 vezes nas páginas 11 e 14.
- NETO, J. C.; BLANCO, M. B.; SILVA, J. A. da. O uso de gamificação e dificuldades matemáticas: possíveis aproximações. **RENOTE - Revista Novas Tecnologias na Educação**, v. 15, n. 1, 2017. ISSN 1679-1916. Disponível em: <<https://seer.ufrgs.br/renote/article/download/75151/42586>>. Citado 3 vezes nas páginas 7, 8 e 10.

- PRESSMAN, R. S. Engenharia de software. São Paulo, 2011. Citado na página 14.
- SANTOS, L. A. F. **Software gamificado para auxílio ao ensino e aprendizagem de matemática para crianças**. Brasília, Brazil, p. 75, 2017. Citado na página 13.
- SILVA, V. *et al.* Proposta de um aplicativo gamificado para o ensino de cálculo. **Congresso Regional sobre Tecnologias na Educação**, Recife/PE - Brasil, 2016. Disponível em: <http://ceur-ws.org/Vol-1667/CtrlE_2016_AC_paper_14.pdf>. Citado 3 vezes nas páginas 8, 11 e 12.
- SOUZA, L. F. D. de. **Evasão do curso de Licenciatura em Matemática (Noturno) da Universidade de Brasília**. Brasília, 2016. Citado na página 8.
- SOUZA, M.; FRANÇA, C. O que explica o sucesso de jogos no ensino de engenharia de software? uma teoria de motivação. **WEI - 24º Workshop sobre Educação em Computação**, 08 2016. Citado 3 vezes nas páginas 11, 12 e 14.
- VALLE, P. H. D. Jogos educacionais: uma contribuição para o ensino de teste de software. São Carlos, 2017. Citado na página 14.