

**UNIVERSIDADE DE CAXIAS DO SUL  
ÁREA DE CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS**

**ADOLFO RAZZERA GAJARDO**

**SOFTWARE DE GESTÃO PARA APLICATIVO EDUCACIONAL NO ENSINO DE  
PROGRAMAÇÃO PARA CRIANÇAS**

**CAXIAS DO SUL  
2018**

**ADOLFO RAZZERA GAJARDO**

**SOFTWARE DE GESTÃO PARA APLICATIVO EDUCACIONAL NO ENSINO DE  
PROGRAMAÇÃO PARA CRIANÇAS**

Trabalho de Conclusão de Curso para obtenção  
do título de Bacharel em Ciência da  
Computação pela Universidade de Caxias do  
Sul.

Orientador:  
Profa. Dra. Carine Geltrudes Webber

**CAXIAS DO SUL  
2018**

## RESUMO

Este estudo foi realizado tendo em vista o ensino de programação na educação infantil, buscando desenvolver facilidades para o professor no quesito de ensinar e administrar suas turmas. Os temas abordados neste trabalho buscam aprofundar este assunto, utilizando como exemplos cinco ferramentas de ensino de programação, sendo elas: Codie, Codeybot, DuinoBlocks, Tynker e Rob The Mouse, sendo esta última a ferramenta criada por alunos da UCS, a qual este trabalho propõe melhorias. Dentre os temas abordados neste trabalho temos os benefícios e dificuldades no ensino da programação, tanto do lado do aluno quanto o lado do professor, a inclusão da tecnologia na educação básica, as facilidades que ajudam o professor a ensinar programação e por fim uma análise das ferramentas citadas. Com o intuito de melhorar e adicionar funcionalidades à ferramenta Rob The Mouse, este trabalho tem como objetivo o desenvolvimento de um banco de dados na nuvem e a criação de uma funcionalidade para professores administrarem suas turmas. Através da ferramenta Firebase Realtime Database foi desenvolvido um banco de dados na nuvem onde é possível salvar os dados gerados na utilização do aplicativo e carregá-los em outro dispositivo, sendo também possível conectar diversos dispositivos ao banco simultaneamente. Com o uso desta ferramenta, também foi possível desenvolver uma forma de sempre manter todos os dados de dispositivos diferentes sincronizados entre si e com o banco de dados na nuvem. Para os professores, foi também proposta uma funcionalidade adicional no aplicativo, que os permitam observar os resultados obtidos pelos seus alunos nas tarefas, a média de cada turma, bem como ver quais são suas turmas e quais alunos estão em cada turma. Através desta funcionalidade, professores podem identificar quais alunos estão com mais dificuldades no aprendizado da programação, ao observar seu histórico de resoluções dos cenários. Por fim, o projeto foi experimentado em sala de aula, com alunos da educação infantil. Os alunos foram avaliados em sua percepção dos problemas propostos, na forma em que resolveram o problema e no tempo de resolução do problema.

**Palavras-chave:** Software Educacional. Robótica Educacional. Programação para Crianças. Ensino de Programação. Plataforma de Programação.

## ABSTRACT

This study was accomplished with a view of teaching programming in early childhood education, seeking to develop facilities to the teacher in the question of teach and manage their classes. The topics covered in this work seek to deepen this subject, using five programming teaching tools as examples: Codie, Codeybot, DuinoBlocks, Tynker and Rob The Mouse, the latter being the tool created by UCS students, to which this work proposes improvements. Among the topics covered in this work we have the benefits and difficulties in the teaching of programming, both on the student's side and on the teacher's side, the inclusion of technology in basic education, the facilities that help the teacher to teach programming and finally an analysis of the tools. In order to improve and add new functionalities to the Rob The Mouse tool, this work aims to develop a database in the cloud and to create a functionality for teachers to administer their classes. With the Firebase Realtime Database tool, a database was developed in the cloud where it is possible to save the data generated by using the application and download it to another device, and it is also possible to connect several devices to the database simultaneously. With the use of this tool, it was also possible to develop a way to always keep all the data of different devices synchronized with each other and with the database in the cloud. For teachers, an additional functionality was proposed in the application, which allows them to observe the results obtained by their students in the tasks, the average of each class, which are their classes and which students are in each class. Through this functionality, teachers can identify which students are struggling in programming learning by looking at their scenario resolution history. Finally, the project was tried in the classroom, with children's education students. The students were evaluated in their perception of the proposed problems, in the way they solved the problem and in the time they resolved the problem.

**Keywords:** Educational Software. Educational Robotics. Programming for Children. Teaching Programming. Programming Platforms.

## LISTA DE FIGURAS

Figura 1 – Exemplo de linguagem visual . . . . .	11
Figura 2 – Codie . . . . .	13
Figura 3 – Codie . . . . .	13
Figura 4 – Codeybot . . . . .	14
Figura 5 – Codeybot App . . . . .	14
Figura 6 – Codie - mBlockly . . . . .	15
Figura 7 – DuinoBlocks . . . . .	17
Figura 8 – DuinoBlocks for Kids . . . . .	18
Figura 9 – Tynker - Linguagens Disponíveis . . . . .	19
Figura 10 – Tynker - Drones . . . . .	19
Figura 11 – Tynker - Exemplo de código . . . . .	20
Figura 12 – Tynker - Kit LEGO WeDo 2.0 . . . . .	21
Figura 13 – Tynker - Classroom Management . . . . .	21
Figura 14 – Rob - The Mouse . . . . .	22
Figura 15 – Arquitetura da ferramenta . . . . .	23
Figura 16 – Tela Inicial do aplicativo . . . . .	23
Figura 17 – Exemplo de cenário da tarefa . . . . .	24
Figura 18 – Exemplo de resolução do cenário . . . . .	25
Figura 19 – Tela Principal da Aplicação . . . . .	27
Figura 20 – Menu Principal . . . . .	27
Figura 21 – Criação de Cenário . . . . .	28
Figura 22 – Tela Jogar Cenário . . . . .	29
Figura 23 – Cadastro de Escola . . . . .	29
Figura 24 – Cadastro de Turma . . . . .	30
Figura 25 – Cadastro de Aluno . . . . .	30
Figura 26 – Tela do Joystick . . . . .	31
Figura 27 – Tela de Consultas . . . . .	32
Figura 28 – Telas do Software de Gestão . . . . .	33
Figura 29 – Regras de Segurança . . . . .	35
Figura 30 – Banco de Dados Firebase . . . . .	36
Figura 31 – Estrutura das Tabelas Firebase . . . . .	37
Figura 32 – Tabela de Controle . . . . .	37
Figura 33 – Diagrama Explicativo . . . . .	38
Figura 34 – Diagrama Atividades . . . . .	39
Figura 35 – Banco de dados . . . . .	39
Figura 36 – Tela de turmas . . . . .	40
Figura 37 – Tela da turma . . . . .	40
Figura 38 – Tela Aluno . . . . .	40
Figura 39 – Bookshelf App . . . . .	41

Figura 40 – Tarefa 1 . . . . .	43
Figura 41 – Tarefa 2 . . . . .	44
Figura 42 – Tarefa 3 . . . . .	44
Figura 43 – Tempos dos Alunos nos Cenários . . . . .	45
Figura 44 – Tabelas Firebase . . . . .	47
Figura 45 – Cenário 1 . . . . .	50
Figura 46 – Cenário 2 . . . . .	50
Figura 47 – Cenário 3 . . . . .	50
Figura 48 – Tempos dos Alunos nos Cenários . . . . .	51
Figura 49 – Software de Gestão . . . . .	52

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>7</b>
<b>1.1</b>	<b>Objetivos</b>	<b>8</b>
1.1.1	Objetivo Geral	8
1.1.2	Objetivos Específicos	8
<b>1.2</b>	<b>Organização do Documento</b>	<b>8</b>
<b>2</b>	<b>PLATAFORMAS PARA ENSINO DE PROGRAMAÇÃO</b>	<b>9</b>
<b>2.1</b>	<b>Características das plataformas de programação</b>	<b>11</b>
<b>2.2</b>	<b>Revisão sistemática</b>	<b>12</b>
<b>2.3</b>	<b>Codie</b>	<b>12</b>
<b>2.4</b>	<b>Codeybot</b>	<b>13</b>
<b>2.5</b>	<b>DuinoBlocks</b>	<b>15</b>
<b>2.6</b>	<b>Tynker</b>	<b>17</b>
<b>2.7</b>	<b>App Rob</b>	<b>21</b>
<b>2.8</b>	<b>Considerações Finais</b>	<b>25</b>
<b>3</b>	<b>SOFTWARE GERENCIADOR DE CENÁRIOS E TURMAS</b>	<b>27</b>
<b>3.1</b>	<b>Análise do Software Existente</b>	<b>27</b>
<b>3.2</b>	<b>Visão Geral do Software Atualizado</b>	<b>31</b>
<b>3.3</b>	<b>Persistência dos Dados</b>	<b>34</b>
<b>3.4</b>	<b>Modelagem do Software</b>	<b>38</b>
<b>3.5</b>	<b>Definição dos componentes de Interface</b>	<b>40</b>
<b>3.6</b>	<b>Testes Preliminares de Comunicação Cliente-Servidor</b>	<b>41</b>
<b>3.7</b>	<b>Acompanhamento dos Experimentos na Escola</b>	<b>42</b>
<b>3.8</b>	<b>Considerações Finais</b>	<b>45</b>
<b>4</b>	<b>CONCLUSÃO</b>	<b>48</b>
<b>4.1</b>	<b>Síntese do Trabalho</b>	<b>48</b>
<b>4.2</b>	<b>Contribuições e Resultados</b>	<b>49</b>
<b>4.3</b>	<b>Trabalhos Futuros</b>	<b>51</b>
	<b>REFERÊNCIAS</b>	<b>53</b>

## 1 INTRODUÇÃO

Hoje em dia, quando ouvimos os termos lógica e pensamento computacional, logo os associamos à área da informática. Contudo, eles também estão associados com a resolução de problemas do nosso dia-a-dia (DIAS, 2016). Conforme afirma Wing (2006), o pensamento computacional é uma habilidade fundamental para todos. Ela envolve a resolução de problemas, projeto de sistemas e compreensão do comportamento humano ao utilizar conceitos fundamentais da área da ciência da computação (WING, 2006). O desenvolvimento das habilidades computacionais pode iniciar ainda na infância à medida que as crianças aprendem a usar o computador. O pensamento computacional estimula o aprendizado de diversas habilidades, tais como: decomposição de problemas em sub-problemas, planejamento, abstração, resolução de problemas, entre outras.

Ao observar-se a forma como a tecnologia é utilizada pelos alunos nas escolas de ensino fundamental e médio, nota-se que não há contribuição para o desenvolvimento do pensamento computacional propriamente dito. Isto ocorre porque normalmente os alunos utilizam o computador em tarefas bem controladas e definidas, sem a necessidade de reflexão e de organização do trabalho. Segundo Valente (2016), os recursos utilizados pelos alunos se resumem basicamente ao que é chamado de *software de escritório*, como editor de texto e planilhas. Assim, habilidades básicas da ciência da computação não são exploradas, como por exemplo, a habilidade de decompor um problema em subproblemas, e assim recursivamente, a fim de reduzir a complexidade da tarefa (VALENTE, 2016).

De forma similar, o estudo de Zaharija et al (2015), aponta que as crianças estão sendo expostas cada vez mais cedo à tecnologia, programas de computadores e jogos. Mas isto gera apenas um conhecimento superficial sobre o assunto, pois elas ainda são muito novas e não são capazes de compreender completamente o funcionamento da tecnologia (ZAHARIJA; MLADENOVIC; BOLJAT, 2015).

Como uma proposta para aprimorar o ensino do pensamento computacional, encontra-se em desenvolvimento um aplicativo educacional para inserir a lógica computacional para crianças das séries iniciais (LIMA, 2017). Esse aplicativo prevê o uso de um robô que recebe instruções para se movimentar através de um labirinto. As instruções são montadas pelas crianças na forma de um código simples de programação e enviadas ao robô via Bluetooth. Por meio da realização destas tarefas, as primeiras noções do pensamento computacional podem ser apresentadas de forma exploratória aos alunos.

Para o efetivo uso deste aplicativo, surgiu a necessidade de recursos de gestão da turma e acompanhamento das tarefas dos alunos de uma turma de aprendizagem da lógica de programação usando a robótica, para que o professor da turma possa acompanhar o desenvolvimento das tarefas pelos alunos. Assim, dando continuidade a esse projeto, o presente trabalho propõe um sistema que permite o gerenciamento das atividades das crianças, para que o professor possa administrar as aulas de forma satisfatória. Outra necessidade que surgiu foi a de um ser-



vidor na nuvem, para que os dados sejam salvos e carregados a partir de qualquer dispositivo que pretenda ser utilizado nas aulas. Sendo assim, o sistema também faz a conexão entre os dispositivos através deste servidor.

Por fim o software foi utilizado em um experimento com uma turma de educação infantil, onde as crianças foram avaliadas na forma em que elas resolveram os problemas propostos. A funcionalidade do servidor foi avaliada em uma simulação utilizando dois dispositivos diferentes rodando o software simultaneamente.

## **1.1 Objetivos**

### **1.1.1 Objetivo Geral**

Desenvolver componentes de um software educacional para gerenciar as aulas de introdução a programação para crianças. O software deverá fazer a comunicação entre o dispositivo utilizado pelo professor e o banco de dados na nuvem, permitindo ao professor salvar seus dados na nuvem e também baixá-los para o aplicativo.

### **1.1.2 Objetivos Específicos**

Para realizar o objetivo geral proposto serão seguidos os seguintes objetivos específicos:

- Conhecer as necessidades do ensino de programação nas séries iniciais, compostas de crianças de 4 a 5 anos.
- Conhecer e identificar plataformas de comunicação e gerência de softwares e aplicações.
- Realizar a implementação do software.
- Realizar testes com o software. Coletar e analisar os resultados.

## **1.2 Organização do Documento**

O trabalho está organizado em quatro capítulos. O primeiro apresenta a introdução e os objetivos que pretendem ser alcançados com este projeto. O segundo capítulo descreve as plataformas para ensino de programação e suas características, a revisão sistemática utilizada para recuperar trabalhos e estudos ligados ao ensino de programação e também algumas ferramentas existentes para o ensino de programação. O terceiro capítulo apresenta a análise do software existente apontando as funcionalidades presentes antes da realização deste projeto, bem como a definição do software que foi implementado, a arquitetura utilizada e a modelagem do banco de dados utilizado. O quarto e último capítulo apresenta as considerações finais e os resultados obtidos ao término deste projeto.

## 2 PLATAFORMAS PARA ENSINO DE PROGRAMAÇÃO

Muitos pesquisadores e professores concordam que a inclusão da ciência, tecnologia, engenharia e matemática nas primeiras séries do ensino fundamental promove uma forte motivação para futuros estudos e uma grande melhoria na velocidade de aprendizado.

Conforme aponta Balaji et al (2015) em seu trabalho, a inclusão dessas matérias nos ensinos fundamental e médio pode motivar o aluno a buscar uma graduação nas áreas de engenharia e ciência. O autor utiliza como exemplo que algumas escolas nos Estados Unidos, que oferecem cursos de robótica para motivar os alunos a buscarem uma graduação em engenharia, mostrando as oportunidades que esta área oferece (M.BALAJI et al., 2015).

Por outro lado, segundo o estudo de Scaradozzi (2015), muitas escolas primárias incluem conceitos que cobrem ciência, matemática e engenharia em seus currículos, mas pouco se ensina sobre tecnologia, computação e robótica (SCARADOZZI et al., 2015). Porém, essa situação tem melhorado com o passar dos anos, e com o avanço da tecnologia surgiram recursos para melhorar o ensino sobre tecnologia e programação nas escolas.

A arte de programar, explicada de uma maneira simples, consiste em ordenar uma série de instruções a fim de solucionar um problema. Quando se trata de situações reais da vida, nem sempre as instruções serão executadas por um computador. Muitas vezes nos deparamos com problemas para quais podemos resolver pensando computacionalmente. O que poucos se dão conta é de que todos os dias as pessoas programam a si mesmas. Afinal, todos os dias pela manhã elas se programam para sair de casa utilizando as seguintes instruções por exemplo: acordar, escovar os dentes, tomar o banho, tomar café e sair.

Essa habilidade de pensar computacionalmente, que ocorre quase involuntariamente com tarefas tão triviais, torna-se de grande ajuda para tarefas mais complexas, que exigem mais atenção e raciocínio lógico. Todavia, para o efetivo uso desta habilidade, ela deve ser desenvolvida. Uma das melhores formas de desenvolvê-la é aprendendo a programar.

Aprender a programar é extremamente importante. Segundo Scaico (2013), desenvolver algoritmos é a atividade principal para todas as áreas ligadas a computação. Porém, esta atividade deveria ser estudada e desenvolvida por todos os estudantes, quando ainda estão em idade escolar, e não apenas por estudantes da área da computação (SCAICO et al., 2013).

Em primeiro lugar, este tipo de educação permite o desenvolvimento de diversas capacidades que contribuem para melhorar o raciocínio lógico dos estudantes. Programar envolve a habilidade de desenvolver uma solução para um problema, que se for grande requererá o exercício de outras habilidades (como dividir o problema em subproblemas e criar uma solução central). Além disso, mais aproximação com essa área pode gerar uma influência importante para a escolha das carreiras dos adolescentes [...] (SCAICO et al., 2013)

Outra razão para ensinar programação na escola mencionada por Scaico, é que estando em

contato com a programação, esta contribui para melhorar as atitudes e a visão dos jovens com relação ao uso das tecnologias. Apenas fazer o uso das tecnologias não os torna grandes conhecedores dela, mas apenas consumidores da tecnologia. Agora, quando aprendem como ela funciona de verdade, os jovens irão compreender o potencial de criação presente na programação (SCAICO et al., 2013).

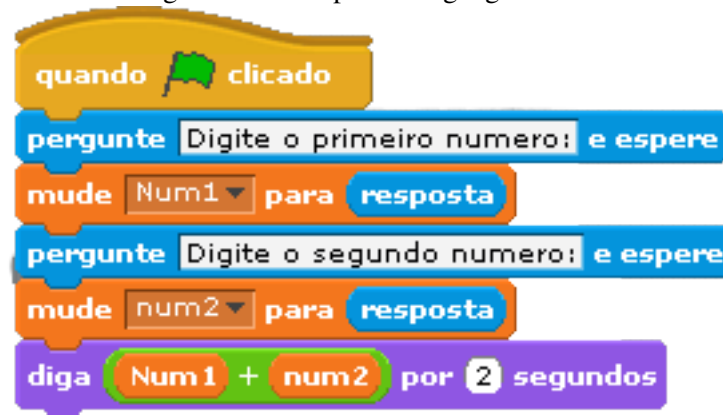
Contudo, algumas dificuldades podem ser observadas na aprendizagem da programação. Conforme apontam Gomes et al (2008), o ensino de programação cobre muitos conceitos dinâmicos que são ensinados de maneira estática, com apresentações, explicações verbais, desenhos no quadro e textos, não promovendo nos alunos uma plena compreensão da dinâmica envolvida. Outro fator apontado é de que os alunos encontram dificuldades para abstrair as informações do problema para construir uma solução (GOMES; HENRIQUES; MENDES, 2008). Segundo as palavras dos autores:

[...] a preocupação principal deveria ser antes de mais nada o desenvolvimento da capacidade de resolução de problemas, aparecendo a linguagem de programação apenas como um veículo para concretizar essa resolução, ou seja, para expressar o algoritmo ou estratégia de resolução. (GOMES; HENRIQUES; MENDES, 2008)

Ainda segundo os autores, dificuldades também podem ser observadas nos métodos de estudos dos alunos. Muitas vezes, eles começam a programar uma solução sem terem compreendido totalmente o problema, os dados e o que é esperado obter como resultado, devido a dificuldades de interpretação e abstração. Outra negligência cometida pelos alunos ocorre na etapa final da programação, os testes. Muitos os fazem bastante superficialmente, com dados muito simples e reduzidos, sem testar os limites do programa criado.

Com o intuito de mitigar estas dificuldades encontradas na aprendizagem, linguagens visuais de programação podem ser utilizadas para um aprendizado inicial. Uma linguagem visual difere-se de uma linguagem de programação convencional pois os comandos são apresentados em forma de blocos, que devem ser montados para criar um programa. Isso permite que o aluno trabalhe com a programação através da montagem de blocos, fato que facilita a autoaprendizagem e a criatividade para resolução de problemas (RIBAS; BIANCO; LAHM, 2016). Na Figura 1, temos um exemplo de uma linguagem visual de programação, utilizada na ferramenta Scratch. As linguagens visuais de programação foram criadas para serem utilizadas por ferramentas ou plataformas de ensino a programação, sendo muitas delas voltadas para a introdução da programação para crianças e jovens.

Figura 1 – Exemplo de linguagem visual



Fonte: (SCRATCH, 2007)

## 2.1 Características das plataformas de programação

Uma plataforma para o ensino de programação é um ambiente de software, geralmente on-line, que disponibiliza uma série de recursos, que dão suporte ao processo de aprendizagem, permitindo seu planejamento, implementação e avaliação. Neste ambiente, o aluno aprende a programar com problemas simples de forma intuitiva, cujo nível de complexidade vai aumentando gradualmente. Para isso, o aluno utiliza uma linguagem que pode ser visual, baseada em blocos de comando, onde o aluno arrasta os blocos e monta o código para resolver certa tarefa. (COSTA; ALVELOS; TEIXEIRA, 2012).

O professor também tem um papel importante no uso de uma plataforma. Caberá ao professor organizar e administrar as aulas de programação, podendo ele criar turmas, incluir alunos nas turmas, definir os problemas e atividades relacionados a aula, atribuir tarefas para os alunos e o mais importante, monitorar as atividades e o processo de aprendizagem dos alunos.

Segundo Costa et al (2012), dentre as características gerais mais importantes presentes nessas plataformas, temos:

- Disponibilizar conteúdo de ensino adequado bem como recursos para auxiliar o processo de aprendizagem, tanto para o aluno quanto para o professor.
- Permitir aos alunos acessar as lições e conteúdos disponibilizados pelo professor, resolver as atividades propostas pelo professor e salvar suas soluções de forma que possam ser acessadas futuramente.
- Permitir aos professores administrar suas turmas, podendo incluir e excluir alunos, disponibilizar lições e atividades para os alunos e acompanhar o andamento do aprendizado dos alunos.

Dentre os conceitos básicos que devem estar disponíveis para serem ensinados, deve haver

todo o básico sobre programação, desde comandos básicos e noção de variável a condicionais, estruturas de repetição e até recursão.

## 2.2 Revisão sistemática

O processo de pesquisa realizado para coletar trabalhos e estudos ligados ao ensino de programação para crianças se deu por meio do site ScienceDirect, onde foram utilizadas as seguintes palavras-chave para a busca: *learning, programming, children, robot, platforms*. Junto a estas palavras-chave foram utilizados os critérios de ano para artigos apenas a partir de 2014, e artigos de acesso aberto. Com estes critérios foram recuperados 74 artigos, dentre os quais 62 foram descartados por não apresentarem ligação com a área deste trabalho, e 12 foram selecionados para leitura mais detalhada. Após a leitura, 3 dentre os 12 foram selecionados por apresentarem um conteúdo mais significativo para este trabalho, enquanto os outros foram descartados. Para complementar o referencial teórico, outros trabalhos de diferentes fontes foram recuperados.

## 2.3 Codie

O robô Codie, desenvolvido com base no Arduino, pode ser muito útil para ajudar no aprendizado de programação e no desenvolvimento do pensamento computacional. O robô conta com um aplicativo para celular que possibilita a programação do robô para diversas funções.

Para a criação do programa, é necessário arrastar e montar os blocos de forma sequencial e, após o programa ser criado, basta executar o código, que é enviado ao robô via bluetooth. Os blocos são compostos por movimentos, giros, sons, condicionais, espera, e as luzes LED do robô. Logo, é possível montar programas bastante elaborados com o Codie.

Para o devido uso deste robô na sala de aula, inicialmente deve ser feito com a orientação de um professor, que orientará a criança em como programar o robô e que tipo de algoritmos serão ensinados (LIPECZ, 2015). O Codie não possui uma funcionalidade implementada em seu aplicativo para a elaboração de tarefas e cenários, devendo então o professor criar cenários e problemas no ambiente físico da sala de aula. Contudo, são inúmeras as possibilidades de cenários que podem ser criados, como por exemplo, um circuito com obstáculos para ser percorrido, ou utilizar o robô para carregar objetos de um ponto a outro, entre outros.

Na Figura 3 podemos observar um exemplo de código criado na plataforma Codie. O exemplo mostra um programa simples onde o robô irá testar se o sensor está captando algo a sua frente e moverá para frente caso o caminho esteja livre senão irá mover-se para trás e fará um giro de 90 graus. A visualização da execução do programa se dá apenas no robô, que irá se comportar conforme o código enviado a ele.

Figura 2 – Codie



Fonte: (LIPECZ, 2015)

Figura 3 – Codie



Fonte: (LIPECZ, 2015)

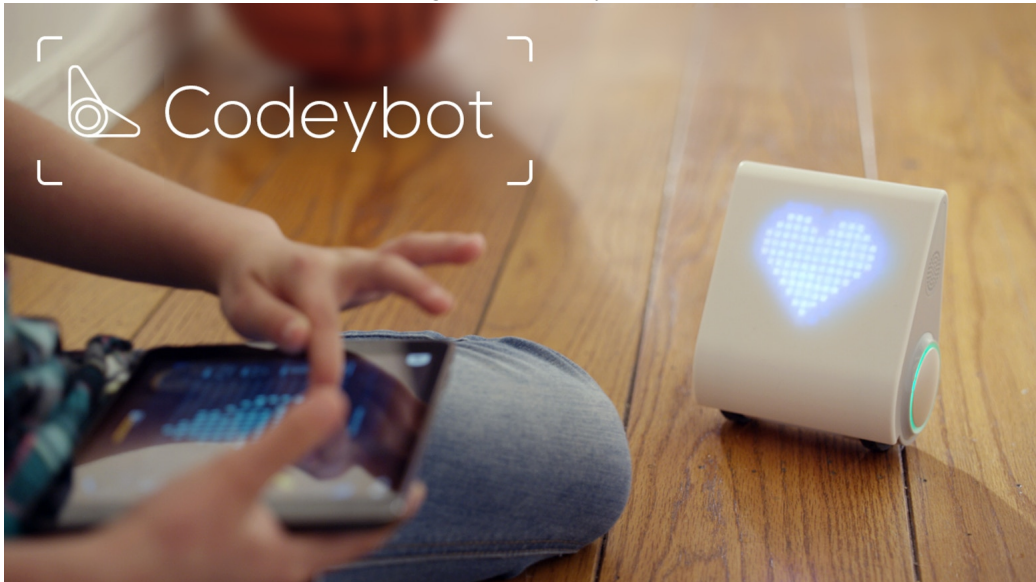
## 2.4 Codeybot

Similar ao Codie, o Codeybot é um robô educacional que ensina o básico de programação através de várias características interativas que prendem a atenção da criança. Criado por Jasen Wang, o projeto começou em 2012 no Kickstarter e cresceu bastante ao receber parcerias de grandes corporações como Microsoft, Intel e MIT (Instituto de Tecnologia de Massachusetts).

O robô possui um painel de LED para mostrar caras e desenhos, speakers para tocar música e fazer vozes engraçadas e até mesmo um modo de batalha. O Codeybot possui algumas funcionalidades pré-programadas, como desenhar caras no painel de LED e tocar música e dançar,

mas é possível programá-lo através de dois aplicativos, o Codeybot App e o mBlockly App (MAKEBLOCK, 2016).

Figura 4 – Codeybot



Fonte: (MAKEBLOCK, 2016)

O Codeybot App, mostrado na Figura 5, é uma interface bastante intuitiva, com a qual é possível comandar diretamente o robô. Com a interface é possível controlar o robô por um controle joystick virtual ou por comandos de voz, tendo até mesmo a opção de aumentar a velocidade de movimento por um curto período (Boost). Também é possível trocar a cor das luzes nas laterais do robô bem como criar carinhas e desenhos para serem mostrados no painel LED. Até oito desenhos podem ser criados e salvos para serem usados em futuros programas utilizando o menu central na interface.

Figura 5 – Codeybot App



Fonte: (MAKEBLOCK, 2016)

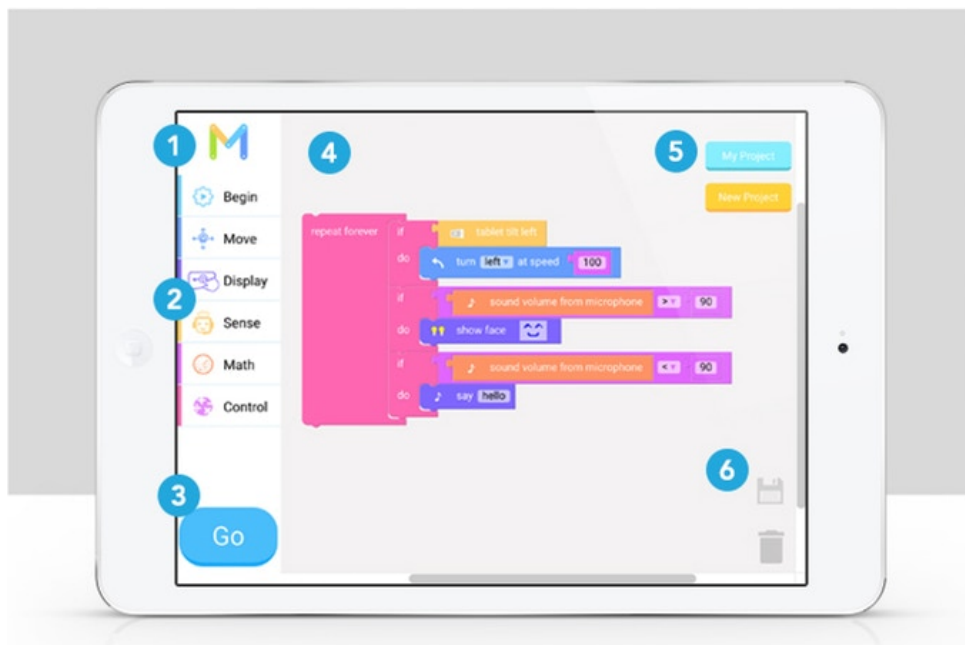
Com o mBlockly App, Figura 6, é possível programar o robô utilizando uma linguagem

de programação visual baseada em blocos de programação, onde o usuário arrasta e solta os comandos na posição desejada. Dentre os comandos disponíveis há comandos de condicionais, de repetição, de sensores do robô e de características do robô, como mudar o desenho no painel LED, tocar uma música ou falar através dos speakers.

Para montar o programa, o usuário escolhe o bloco que deseja adicionar ao código utilizando o menu a esquerda, selecionando entre as diferentes categorias de blocos disponíveis. Dentre as categorias podemos encontrar blocos de início do programa, para controlar a movimentação do robô, para ler os dados captados pelos sensores, para mostrar desenhos na tela LED, para tocar músicas e sons através dos speakers e para executar operações matemáticas.

No exemplo mostrado na Figura 6, temos três condicionais que são repetidos por tempo indeterminado. No primeiro condicional, o programa observa se o dispositivo utilizado esta inclinado para a esquerda e, se caso esteja, então o robô faz uma curva para a esquerda com a velocidade indicada. No segundo condicional, o programa observa se o volume do som do microfone do dispositivo utilizado é maior que 90 e, se caso seja, então o robô mostrará uma carinha feliz no painel LED. No último condicional, o programa observa se o volume do som do microfone do dispositivo utilizado é menor que 90 e, se caso seja, então o robô emitirá um som através dos speakers, ele dirá "Hello".

Figura 6 – Codie - mBlockly



Fonte: (MAKEBLOCK, 2016)

## 2.5 DuinoBlocks

O DuinoBlocks foi desenvolvido tendo como objetivo propor e desenvolver um ambiente de programação que utiliza uma linguagem visual, permitindo aos usuários iniciantes na pro-



gramação desenvolverem suas habilidades programando um dispositivo robótico baseado no hardware Arduino. Segundo o estudo de Alves (2013), os testes realizados no ambiente mostraram que professores e alunos se sentirão mais confiantes ao trabalhar com a linguagem visual do ambiente ao invés da linguagem textual Wiring, padrão do Arduino (ALVES, 2013).

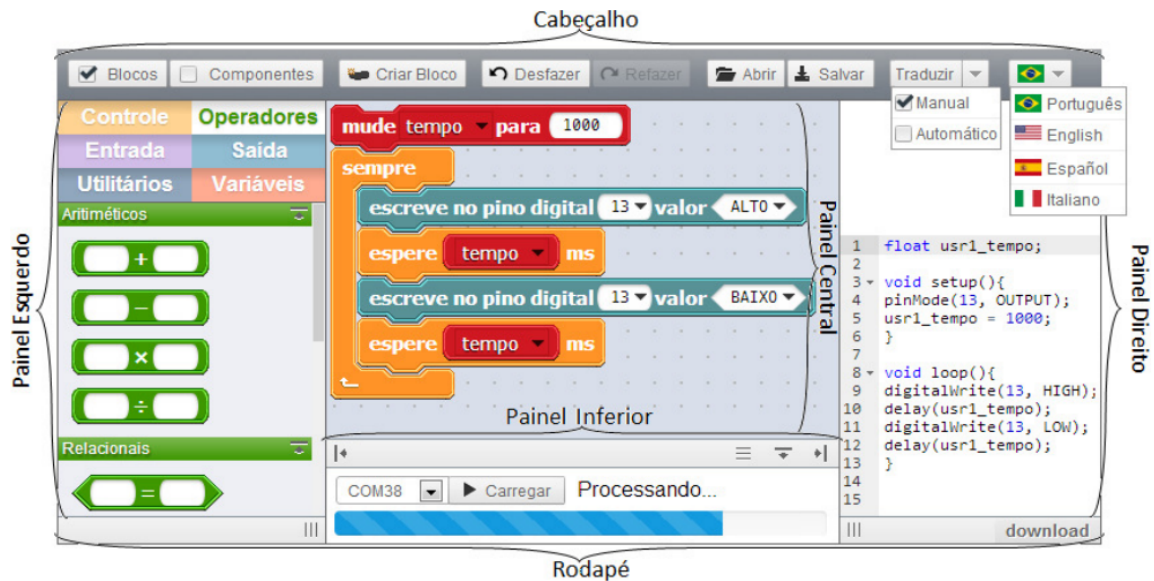
Como a maioria dos kits de robótica disponíveis no mercado e que são utilizados pelas escolas possuem um alto custo e geralmente apresentam uma linguagem de programação textual, o trabalho dos alunos e professores iniciantes em programação se torna árduo. Outro objetivo por trás da criação do DuinoBlocks foi justamente utilizar a plataforma Arduino para que o custo final ficasse bastante acessível para as instituições de ensino.

A Figura 7 mostra o layout do ambiente DuinoBlocks. A tela esta dividida em 6 seções, sendo elas:

- **Cabeçalho:** No cabeçalho encontramos a barra de ferramentas, com algumas funções básicas do software, como opções de alterar entre módulos de blocos e componentes, abrir e salvar projetos, desfazer e refazer passos, traduzir o algoritmo em blocos para texto e selecionar o idioma do ambiente.
- **Painel Esquerdo:** O painel esquerdo possui uma lista com os blocos separados por categorias e subcategorias. Dentre as principais categorias temos **Controle** (cor laranja), que possui blocos que representam repetição e desvio condicionais; **Operadores** (cor verde), que possui blocos para construir expressões matemáticas; **Entrada** (cor roxo) e **Saida** (cor azul claro), que possuem blocos para realizar leitura e escrita analógica ou digital através de uma porta do Arduino, onde um componente está conectado; **Utilitários** (cor azul escuro), que possui blocos para funções extras, como funções matemáticas e trigonométricas, comandos alfanuméricos, comunicação serial e outros; **Variáveis** (cor vermelha), que possui blocos para criação de variáveis.
- **Painel direito:** O painel direito possui a tradução do código visual para o código textual, na linguagem Wiring.
- **Painel Central:** No painel central fica a área de construção do código.
- **Painel Inferior:** No painel inferior encontram-se os recursos necessários para carregar o código no Arduino.
- **Rodapé:** No rodapé encontram-se botões para controlar a disposição dos painéis, permitindo ocultar, mostrar e redimensionar os painéis esquerdo, direito e central. Também esta incluído um botão "download" para baixar o código textual traduzido.

A versão atual do ambiente roda na nuvem e seu acesso é feito via navegador web. Versões futuras deste ambiente prometem uma forma de salvar os códigos criados na nuvem e compartilhá-los em comunidades. O que pode ser utilizado por professores para manter um controle sobre o andamento das atividades realizadas pelos alunos (ALVES; SAMPAIO, 2014).

Figura 7 – DuinoBlocks



Fonte: (ALVES; SAMPAIO, 2014)

O DuinoBlocks também serviu de inspiração para o DuinoBlocks for Kids, uma versão simplificada do DuinoBlocks que possui menos funções e uma interface mais agradável. Criado por Queiroz e Sampaio (2015), o DuinoBlocks for Kids está mais voltado para o ensino de conceitos básicos de programação a crianças do ensino fundamental através da robótica educacional.

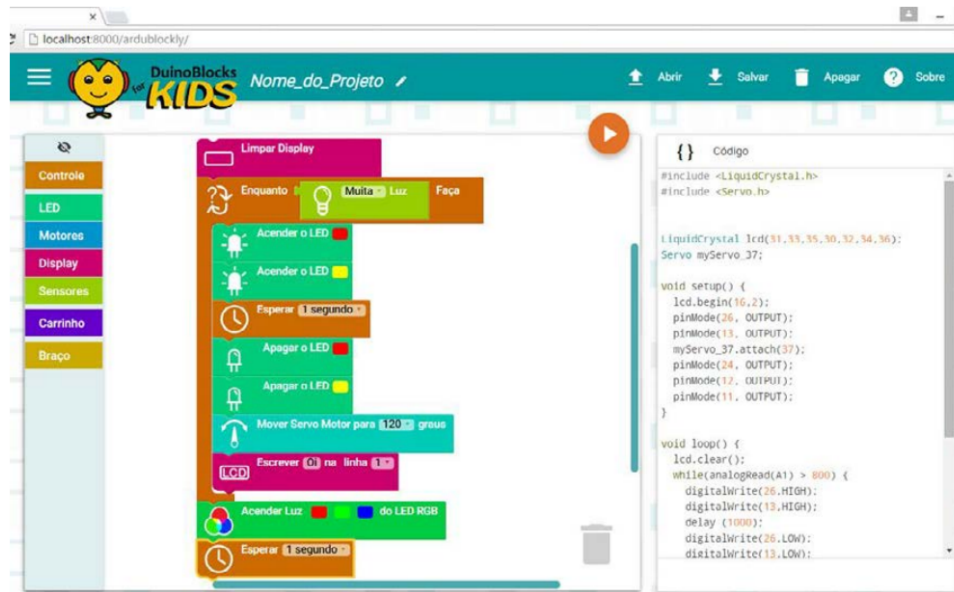
Como podemos observar na Figura 8, foram mantidos apenas os painéis para selecionar as categorias de blocos, o painel para construção do código e o painel que mostra a tradução da linguagem visual para a linguagem textual. Além de tudo, a interface também foi bastante modificada para atrair melhor a atenção das crianças do ensino fundamental (QUEIROZ; SAMPAIO, 2015).

## 2.6 Tynker

Criado em 2013, Tynker é uma criativa plataforma online para ensinar crianças a como programar através de atividades que elas adoram, como jogos e histórias. A plataforma possui mecanismos pré-definidos que são basicamente blocos que representam comandos ou propriedades, que são montados para criar um código. Desse modo, as crianças aprendem o fundamental da programação e construção de código através de uma linguagem visual simples e intuitiva e, conforme vão evoluindo no aprendizado, são apresentadas a desafios mais complexos e até mesmo são introduzidas às linguagens textuais JavaScript e Python. O objetivo desta plataforma é prover para as crianças uma fundação sólida para o modo de pensar computacionalmente, para prepará-las para o futuro (TYNKER, 2017).

Conforme aponta Matos et al (2016), o intuito desta plataforma é passar noções de como

Figura 8 – DuinoBlocks for Kids



Fonte: (QUEIROZ; SAMPAIO, 2015)

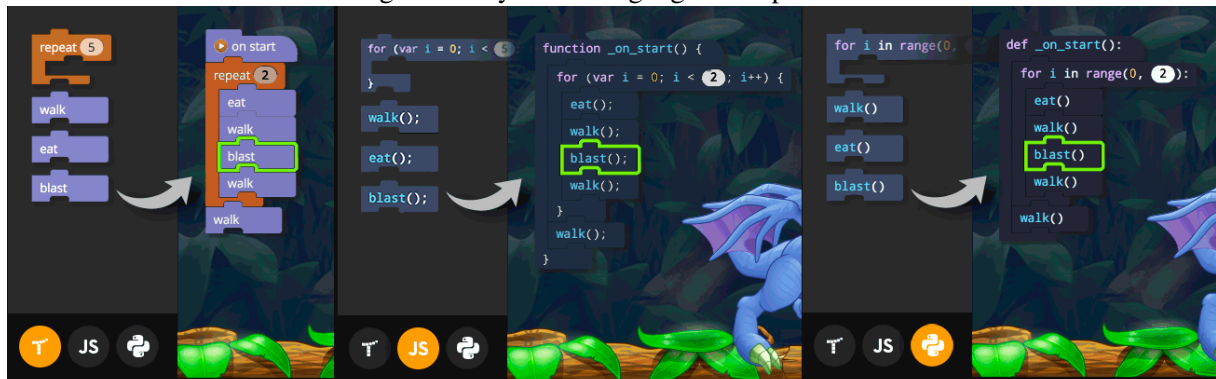
se monta um algoritmo de forma visual e intuitiva em diversos contextos. O serviço oferecido pelo Tynker incentiva não apenas jogar, mas também, e principalmente, criar jogos. O usuário pode criar suas próprias histórias, jogos e desafios e disponibilizá-los no site. O Tynker também possui uma cartilha de boas práticas, dicas de como projetar e tutoriais para auxiliar na parte técnica (MATOS et al., 2016).

O Tynker oferece diferentes formas para aprender a programação, chamadas *Learning Paths*. A mais simples, e recomendada para iniciantes, pode ser utilizada acessando o site do Tynker ou também utilizando um aplicativo disponível apenas para Ipad. Esse *Learning Path* é focado em ensinar os conceitos básicos da programação através da resolução de problemas e quebra cabeças utilizando uma linguagem visual baseada em blocos de comandos. A Figura 9 mostra a interface de criação de código onde é possível escolher a linguagem de programação desejada, tendo disponíveis uma linguagem visual, JavaScript e Python.

Outro *Learning Path* utiliza minidrones como apoio para o ensino. Desta forma a criança aprende a programar criando rotas de voo, manobras e interação com componentes externos do minidrone, como garras para pegar objetos. Na Figura 10 temos os drones disponíveis para serem utilizados.

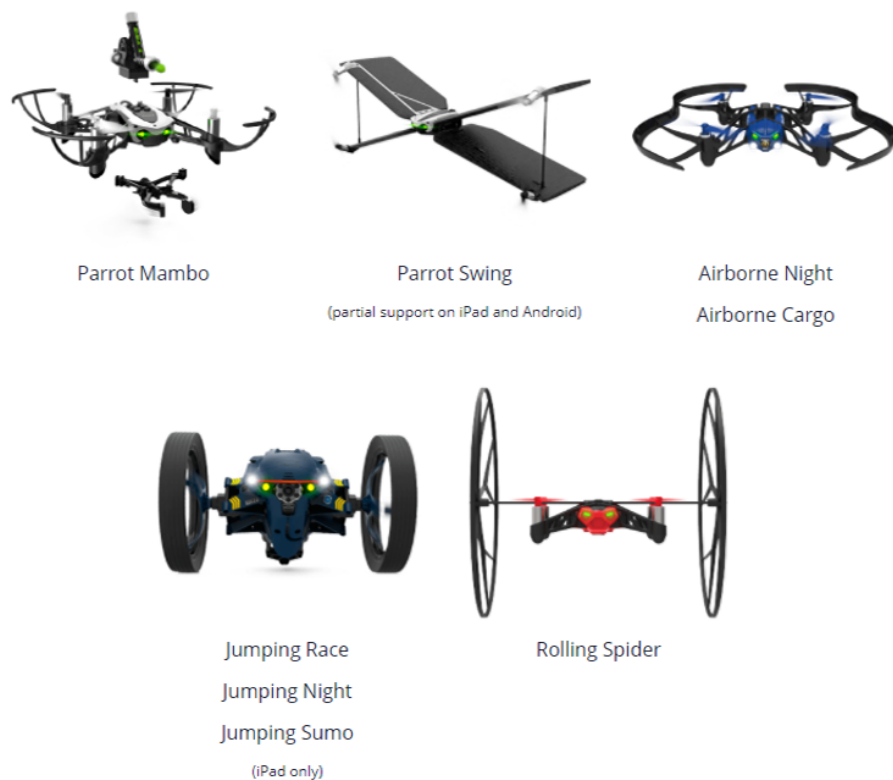
A Figura 11 mostra um exemplo de código criado para o minidrone Mambo. O código representa uma rota de voo que o minidrone seguirá quando ordenado, onde inicialmente ele levantará voo, se movimentará para frente por 1 segundo, fará um giro de 180 graus, novamente se movimentará para frente por 1 segundo e pousará.

Figura 9 – Tynker - Linguagens Disponíveis



Fonte: (TYNKER, 2017)

Figura 10 – Tynker - Drones



Fonte: (TYNKER, 2017)

Outro *Learning Path* utiliza um kit LEGO WeDo 2.0 para apoiar o ensino da programação em conjunto com a robótica. Nesta forma o aluno aprende a programar utilizando um robô LEGO, que possui um motor, sensores de movimento, distância e de toque, luzes LED e speakers. Junto com o kit vem um manual contendo instruções para montar um robô simples, mas a criança é incentivada a fazer suas próprias criações com o kit. Esta forma é bastante utilizada por escolas devido ao seu custo relativamente baixo e a uma ótima taxa de aprendizado.

Figura 11 – Tynker - Exemplo de código



Fonte: (TYNKER, 2017)

A Tynker disponibiliza 11 lições com 65 atividades utilizando o kit WeDo 2.0, onde os alunos aprendem a criar códigos de blocos de comandos, reforçando conceitos importantes na educação STEM (Science, technology, engineering and mathematics). O mesmo kit LEGO também foi utilizado por Scaradozzi (2015) em seu estudo, onde as crianças aprenderam o básico sobre robótica e montaram seu próprio robô LEGO.

A plataforma Tynker também oferece suporte para que crianças possam aprender a programar sozinha em casa ou na escola, em turmas e sob a tutela de um professor. Para ambas as situações, Tynker oferece ferramentas de suporte e controle para os pais e professores. Para os pais é disponibilizado o *Parent Dashboard*, uma ferramenta simples para acompanhar o progresso da criança nas lições, ver os projetos que ela criou e os conceitos que ela aprendeu. A plataforma salva automaticamente o progresso da criança conforme ela avança e aprende através das lições.

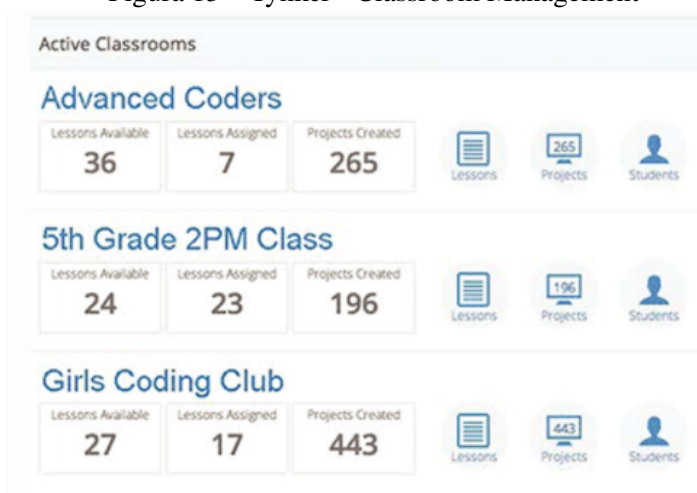
Para os professores, uma ferramenta mais complexa é disponibilizada, o *Classroom Management*. Com esta ferramenta o professor consegue facilmente criar turmas, adicionar e remover alunos, atribuir lições e atividades aos alunos e acompanhar o progresso de todos os alunos, tanto como uma turma quanto individualmente. A ferramenta também permite ver estatísticas sobre a turma, como quantos alunos já fizeram a tarefa, quantos projetos cada um já criou, entre outras estatísticas.

Figura 12 – Tynker - Kit LEGO WeDo 2.0



Fonte: (TYNKER, 2017)

Figura 13 – Tynker - Classroom Management



Fonte: (TYNKER, 2017)

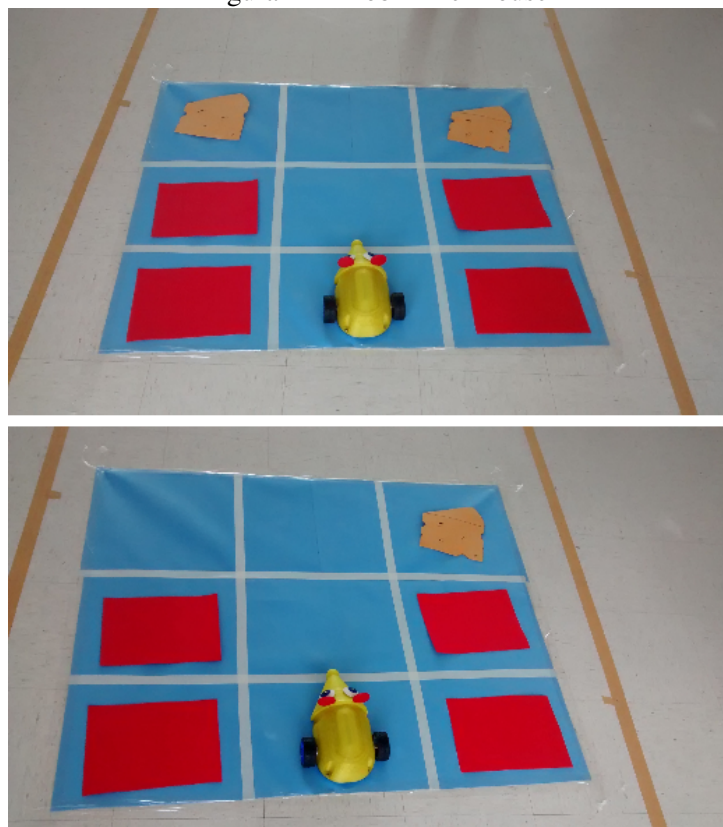
## 2.7 App Rob

Desenvolvido pelos alunos Guilherme Duso, da área de engenharia, e por Luan Lima, da área da ciência da computação, ambos da Universidade de Caxias do Sul (UCS), a ferramenta teve como objetivo promover a inserção da lógica computacional no ensino primário. Para tal, a ferramenta conta com um robô e um aplicativo para plataforma Android (LIMA, 2017).

A plataforma funciona da seguinte forma, a criança utiliza o aplicativo para resolver um

labirinto, onde o ratinho robô deve pegar todos os queijos presentes no labirinto, desviando de obstáculos. Para isso, a criança deve criar um algoritmo para movimentar o robô, utilizando comandos para avançar e girar o ratinho. Este código é enviado para o robô, permitindo assim que o resultado do código criado possa ser visto tanto pelo aplicativo quanto no cenário real, através da movimentação do robô. A Figura 14 mostra o labirinto real por onde o robô se movimentará.

Figura 14 – Rob - The Mouse



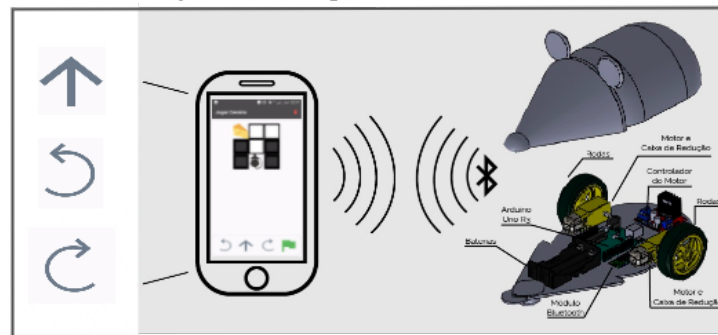
Fonte: (LIMA, 2017)

O aplicativo foi desenvolvido na linguagem Java para o sistema operacional Android por ser uma plataforma móvel e gratuita. Para a comunicação entre o aplicativo e o robô é utilizada uma conexão bluetooth, usada para enviar os comandos ao robô, conforme representado na Figura 15 (LIMA, 2017).

Para que o ensino de programação seja efetivo, o professor deverá criar diversos cenários para os alunos resolverem, além de criar e organizar as turmas e alunos. Começando por cenários simples e pequenos, para a criança se familiarizar com o aplicativo e o robô, avançando para cenários maiores e mais complexos gradualmente.

Ao iniciar o aplicativo, a tela inicial que é apresentada é a tela de cenários (Figura 16). Nessa tela podem ser encontrados e acessados todos os cenários já criados. Através do menu no canto superior esquerdo podemos acessar opções para cadastrar escolas, turmas e alunos, bem como fazer consultas sobre a resolução dos cenários. Para fazer uma consulta, podemos definir

Figura 15 – Arquitetura da ferramenta

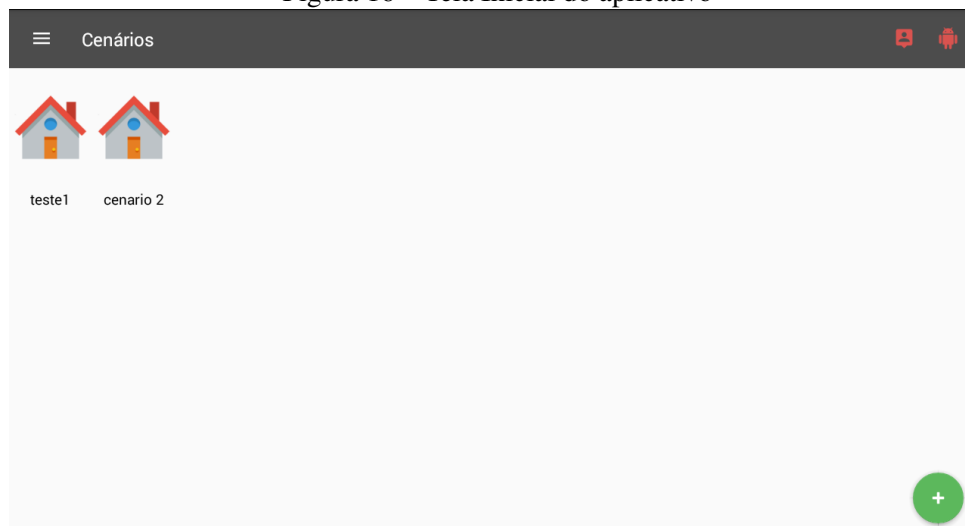


Fonte: (LIMA, 2017)

os parâmetros mais importantes de acordo com a pesquisa desejada. Dentre os parâmetros possíveis temos

- Faixa de datas, definida por uma data inicial e outra final.
- Cenário, podendo ser um cenário específico ou todos os cenários.
- Status, referente ao cenário, se foi resolvido com sucesso ou se ocorreu alguma falha na resolução.
- Aluno, podendo ser um aluno individual ou todos os alunos.
- Gênero, referente ao gênero do aluno, masculino ou feminino.
- Idade, referente a idade do aluno.

Figura 16 – Tela Inicial do aplicativo



Fonte: (LIMA, 2017)



Para criar um cenário, basta o professor acessar a tela de cenários e apertar o botão verde com sinal de adição para adicionar um cenário. No processo, deverá ser escolhido um nome para o cenário e qual o tamanho do tabuleiro desejado para ele, podendo ter um tamanho máximo de 10 x 10. Após criado o cenário, o professor pode apertar o ícone do cenário e segurar por alguns segundos para abrir opções para editar o cenário. Com isso, o professor pode modificar o tabuleiro adicionando obstáculos, como paredes e buracos, o ponto de início do robô e os queijos, que devem ser pegos para completar o cenário.

Para adicionar esses itens ao tabuleiro, primeiro deve ser selecionado o item que se deseja adicionar no pequeno menu na parte de baixo da tela, e apertar a posição do tabuleiro onde deseja-se adicionar o item. A Figura 17 mostra um exemplo de cenário de uma tarefa criada, onde o rato representa o ponto inicial, o raio vermelho representa um buraco, os blocos cinza-escuro representam paredes e o queijo representa um ponto final para o cenário.

Figura 17 – Exemplo de cenário da tarefa



Fonte: (LIMA, 2017)

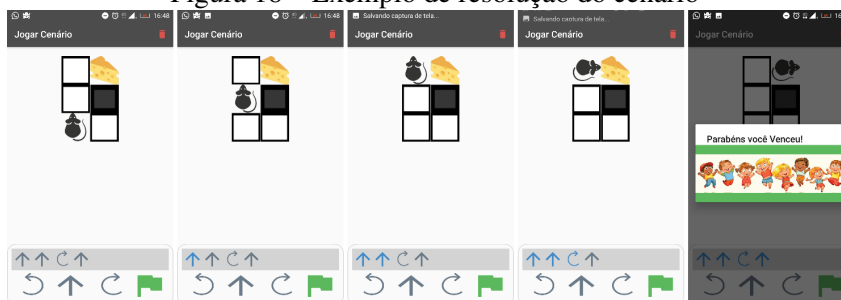
Para entrar na tela de resolução de cenário, basta apertar no cenário que deseja-se resolver. Na tela de resolução, temos os comandos disponíveis para uso na parte de baixo da tela. Conforme o aluno vai criando o código, todos os comandos já escolhidos aparecerão logo acima, com um fundo cinza.

Existem quatro comandos disponíveis para resolver o labirinto, duas setas, uma para cada direção, que giram o robô em 90 graus, seta para frente, que movimento o robô um espaço para frente e a bandeira verde, que ao ser pressionada começa a rodar o código, que pode ser visto em ação tanto pela tela do aplicativo quanto pela movimentação do robô.

A Figura 18 demonstra um exemplo de uso do aplicativo. Ao selecionar os comandos desejados e após selecionar a bandeira verde, podemos ver o rato percorrendo o labirinto de acordo

com os comandos selecionados.

Figura 18 – Exemplo de resolução do cenário



Fonte: (LIMA, 2017)

## 2.8 Considerações Finais

O pensamento computacional e o raciocínio lógico são habilidades fundamentais para todos e introduzi-las às crianças nas séries iniciais é fundamental para seu desenvolvimento. Ao observar como a tecnologia é utilizada nas escolas, nota-se que não há muita contribuição para o desenvolvimento dessas habilidades. Os alunos estão sujeitos a tarefas previamente definidas, sem necessidade de precisarem refletir sobre o problema proposto.

Com o intuito de propor uma melhoria para este sistema de ensino e desenvolvimento das habilidades de raciocínio lógico, surgiu a ideia de ensinar programação para as crianças do ensino fundamental. Para que o ensino seja efetivo, o método a ser aplicado deve ser atrativo para as crianças e professores, por isso é utilizado um robô, um brinquedo aos olhos da criança, para que aprendam a programar brincando com robô. Em conjunto com o robô é utilizado um software que permite a programação e controle sobre o robô.

Várias ferramentas com as características descritas acima foram estudadas. A tabela 1 mostra um resumo das principais características das ferramentas. Entre elas está o robô Codie, que pode ser programado através de um aplicativo para *smartphone*, utilizando uma linguagem de programação visual simples e intuitiva. Apesar de não possuir funcionalidades voltadas para o professor, como criação de tarefas e atividades, as possibilidades de cenários que podem ser representados são inúmeras, pois o robô consegue subir rampas e até carregar objetos. Porém estes cenários devem ser pensados e testados previamente pelo professor.

Outra ferramenta estudada foi o Codeybot, que é similar ao Codie. O que chama a atenção neste robô são algumas funcionalidades bastante atrativas para as crianças, como acender luzes, fazer sons engraçados e mostrar desenhos no painel de LED. A interface é bastante agradável e intuitiva e possibilita salvar e carregar diferentes códigos criados.

O DuinoBlocks difere-se das outras ferramentas por não possuir um robô específico para programar, mas pode ser utilizado em conjunto de qualquer robô que utilize o Arduino como base. A interface é bastante compacta e um pouco complexa no início, mas uma funcionalidade

interessante presente nesta ferramenta é poder ver o código criado em uma linguagem visual ser convertido em uma linguagem textual. O DuinoBlocks também permitirá, em versões futuras, poder salvar e compartilhar os códigos criados. Com esta funcionalidade, professores poderiam acompanhar as atividades das crianças e avaliar seus códigos criados.

Outra ferramenta estudada e uma das mais completas, é a plataforma Tynker. Nesta plataforma as crianças podem aprender a programar de diversas maneiras. Elas podem aprender utilizando apenas o software ou utilizando um minidrone ou robô em conjunto com o software. Em todas essas diferentes maneiras é utilizado uma linguagem de programação visual simples e uma interface intuitiva e agradável. Porém, a funcionalidade mais importante observada nesta plataforma foi recursos de suporte e controle para pais e professores, que permitem acompanhar o aprendizado da criança. Para os professores também estão disponíveis recursos para a gestão de turmas e atividades. Com isso, o professor consegue criar turmas, adicionar alunos a essas turmas, atribuir atividades e lições aos alunos da turma, além de poder acompanhar quem já entregou a atividade, quantos projetos cada um criou, entre outras estatísticas.

A última ferramenta estudada, e também a mais importante para este estudo, é o App Rob, que conta com um robô e um aplicativo. Nesta plataforma, o professor é responsável pela elaboração das tarefas que serão passadas aos alunos, que deverão construir um código utilizando comandos básicos para resolver o problema proposto e movimentar o robô.

Tabela 1 – Resumo das Ferramentas Analisadas

<b>Autor</b>	<b>Robô</b>	<b>Linguagem</b>	<b>Faixa Etária</b>	<b>Programação</b>	<b>Recursos para o Professor</b>
Adam Lipecz (2015)	Codie	Linguagem Visual	5-12 anos	Blocos de Comandos	Permite definir quais serão os algoritmos que serão estudados.
Makeblock (2016)	Codeybot	Linguagem Visual	A partir de 6 anos	Blocos de comandos e Interface	Características interativas que chamam a atenção da criança.
Rafael Machado Alves (2013)	DuinoBlocks	Linguagem Visual e Textual	A partir de 7 anos	Blocos de comandos	Possui opções de salvar e compartilhar códigos na nuvem.
Tynker (2017)	Parrot Minidrones + Lego WeDo 2	Linguagem visual + JavaScript e Python	7-12 anos	Blocos de comandos e Linguagem textual	Possui um sistema de gestão e monitoramento de aulas.
Luan Lima e Guilherme Duso (2017)	Rob the Mouse	Linguagem Visual	4-6 anos	Blocos de comandos	Possui recursos para cadastrar turmas e alunos. Permite a criação de cenários

### 3 SOFTWARE GERENCIADOR DE CENÁRIOS E TURMAS

Este capítulo descreve a implementação de novas funcionalidades ao aplicativo para ensino de programação proposto por Luan Lima (2017). Estas funcionalidades são: um componente para gerenciar as aulas de introdução a programação para crianças, e a integração do aplicativo com o banco de dados Firebase, na nuvem.

#### 3.1 Análise do Software Existente

No software existente os professores têm disponíveis funções para cadastrar escolas, turmas e alunos, criar cenários para serem resolvidos e designar cenários aos alunos de uma turma. Já os alunos têm apenas a função de resolver o cenário (LIMA, 2017). Todos os dados gerados ficam salvos localmente no dispositivo, em um banco de dados SQLite.

A tela inicial do aplicativo, Figura 19, apresenta os cenários já criados, com um botão para adicionar novos cenários no canto inferior direito. Na barra superior se encontram o menu principal, Figura 20, no lado esquerdo, e as opções para selecionar o aluno, que irá tentar resolver os cenários, e o robô a ser utilizado nos cenários. Para selecionar o robô, o dispositivo deverá estar com o Bluetooth ligado.

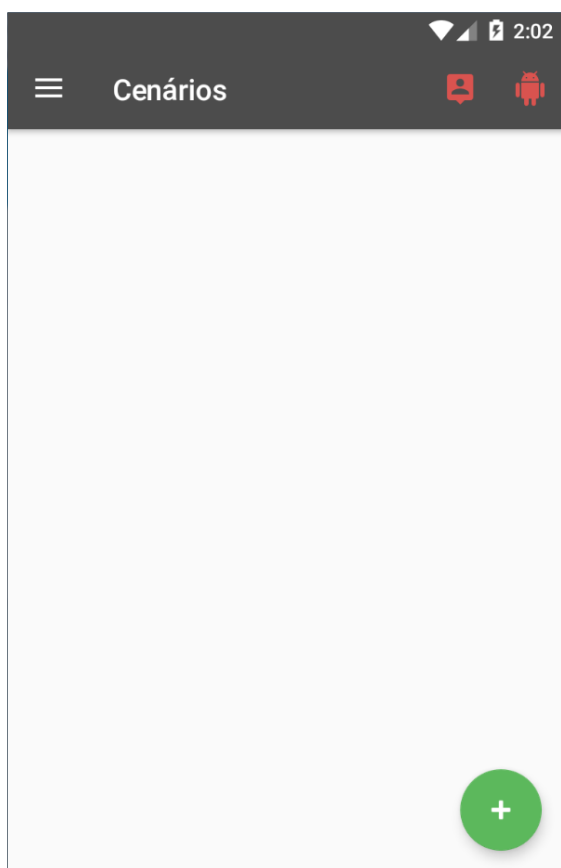


Figura 19 – Tela Principal da Aplicação

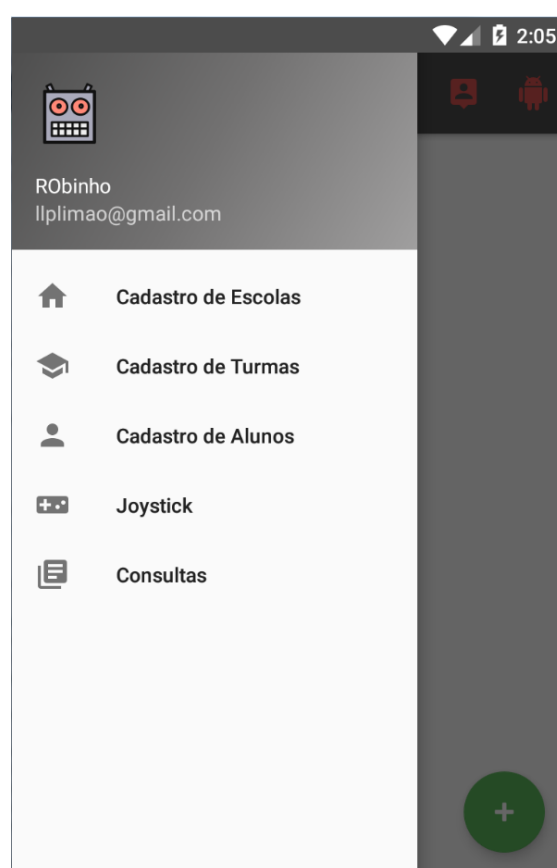
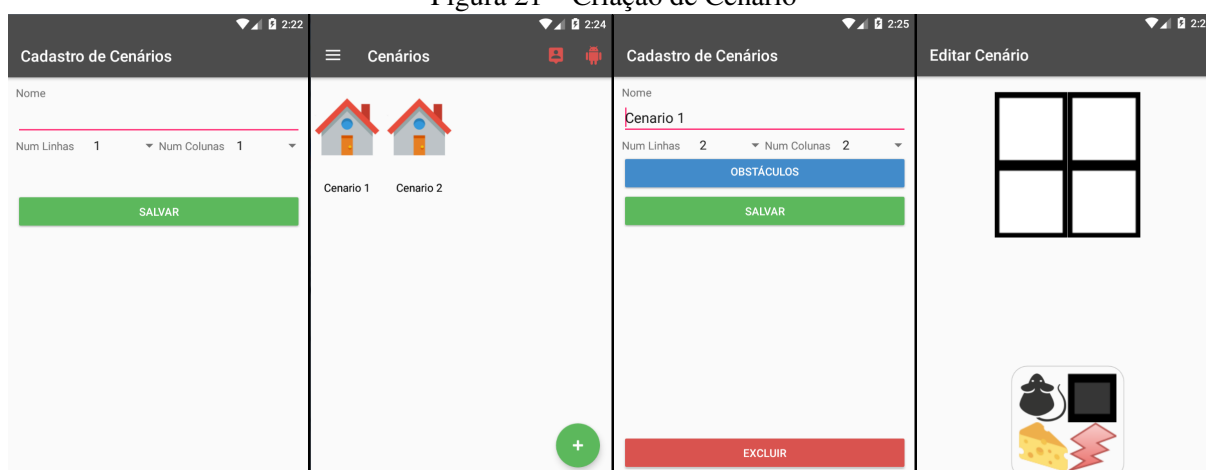


Figura 20 – Menu Principal

Para criar novos cenários, o professor deverá tocar no botão de cadastro de cenário na tela principal, o que o levará para a tela de cadastro de cenário, mostrada na Figura 21. Nesta tela o professor irá definir um nome para o cenário e o tamanho dele, definindo também quantas linhas e colunas o cenário terá. Após criado o cenário, é necessário editá-lo para adicionar obstáculos ao cenário. Para isso, o professor deverá segurar tocado por alguns segundos o cenário desejado. Na tela de edição é possível mudar o nome e o tamanho do cenário, e adicionar obstáculos ao cenário, pressionando o botão obstáculos.

A próxima tela exibirá o labirinto vazio, sem nenhum obstáculo, e as opções de obstáculos na parte inferior. Para adicionar um obstáculo, basta tocar no obstáculo desejado e após na posição a qual deseja-se colocá-lo. Após, basta voltar para a tela anterior e salvar as alterações. A Figura 21 mostra um exemplo de utilização desta funcionalidade. Este exemplo mostra a sequência de criação de cenários pelo professor, primeiramente criando dois cenários (Cenário 1 e Cenário 2) e depois os editando, para adicionar os pontos inicial e final e os obstáculos.

Figura 21 – Criação de Cenário



Para jogar um cenário, é necessário ter um aluno e o robô selecionados através das opções no lado direito da barra superior, na tela principal. Após selecionados, basta tocar no cenário a ser resolvido pelo aluno. Isso o levará para a tela Jogar Cenário, mostrada na Figura 22, que mostrará o cenário na parte central e os comandos disponíveis na parte inferior. Conforme o aluno vai selecionando os comandos, os mesmos irão aparecendo logo abaixo do cenário. É possível apagar um comando da lista tocando no mesmo. Após criado os comandos, basta pressionar a bandeira verde para o robô começar a se mexer pelo cenário real e também pelo cenário na tela do dispositivo. Ao final da execução do código é mostrada uma mensagem informando o resultado do código, se chegou ao objetivo ou não. A Figura 22 também mostra um exemplo de uma aluno resolvendo o cenário, mostrando o código feito pelo aluno no retângulo cinza, acima dos comandos disponíveis.

Através do menu principal, o professor tem disponíveis as funções de cadastrar escolas, turmas e alunos, acessar o joystick para o controle direto sobre o robô e a opção consulta, para

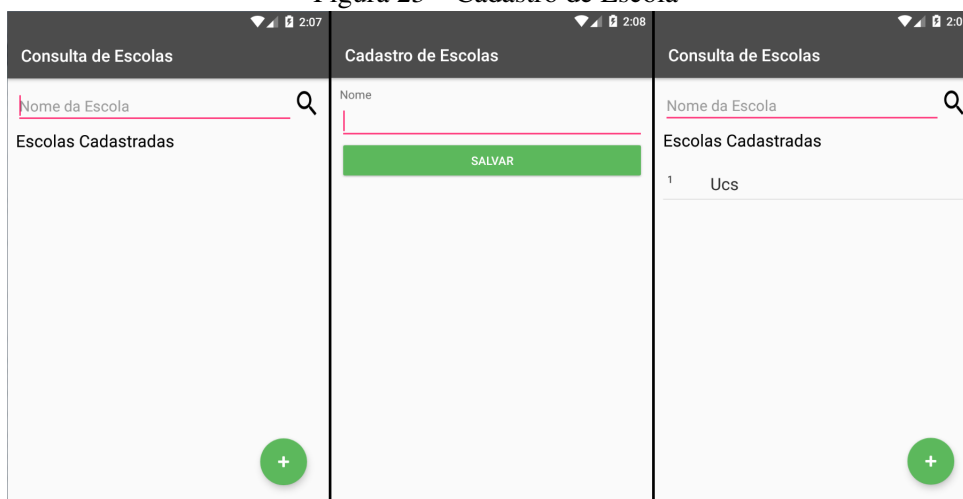
Figura 22 – Tela Jogar Cenário



consultar os cenários desenvolvidos pelos alunos.

Ao acessar a opção Cadastro de Escolas, o professor será apresentado a tela de cadastro de escolas. Esta tela contém a lista com todas as escolas já cadastradas, a opção para buscar uma escola cadastrada, bem como um botão para incluir uma nova escola. Para incluir uma nova escola o professor precisa inserir apenas um nome para a escola. A Figura 23 mostra a sequência de telas para o cadastro de escolas, bem como um exemplo, onde o professor cadastra uma escola no aplicativo.

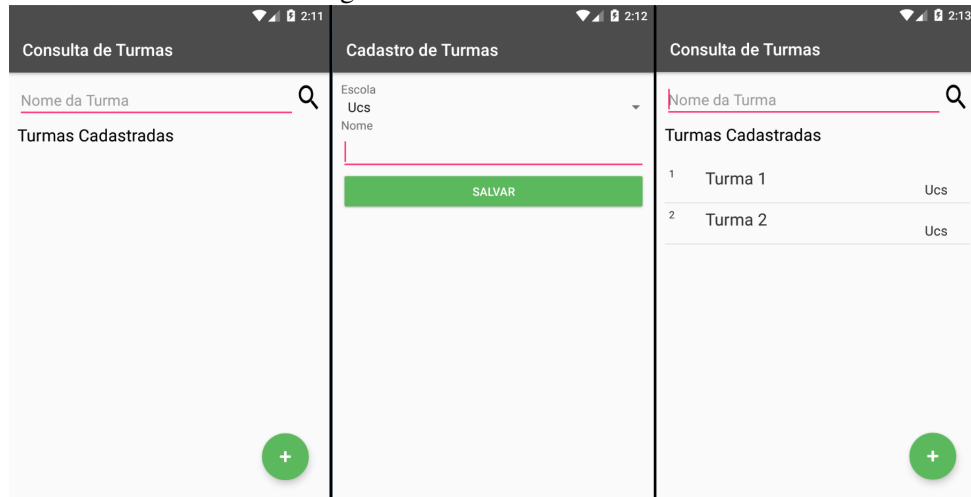
Figura 23 – Cadastro de Escola



Ao acessar a opção Cadastro de Turmas, o professor terá uma tela similar ao cadastro de escolas, com uma lista das turmas já cadastradas, com a escola a qual a turma pertence, a opção de buscar uma turma já cadastrada e o botão para adicionar uma nova turma. Para adicionar uma nova turma, o professor deverá inserir um nome para a turma e a escola que a turma irá pertencer. A Figura 24 mostra a sequência de telas deste processo. Para cadastrar uma turma

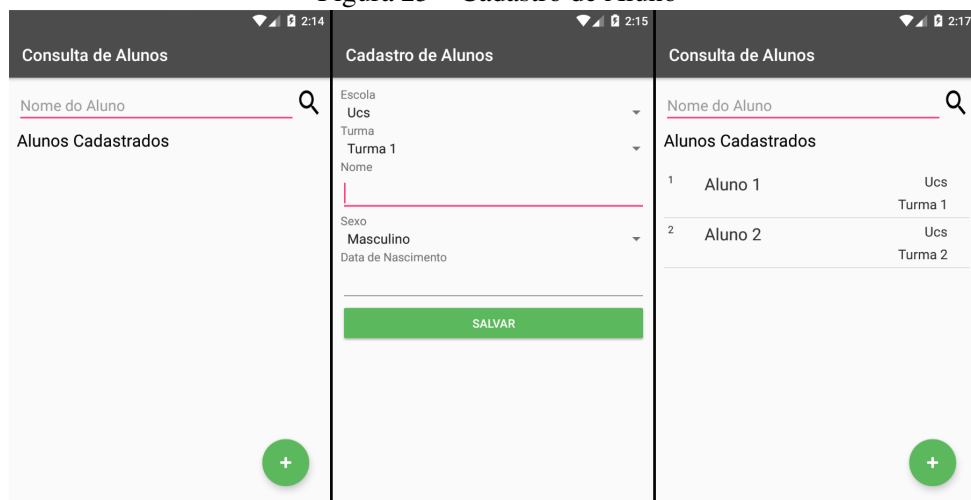
é obrigatório ter pelo menos uma escola cadastrada. A Figura 24 também mostra um exemplo de uso, onde o professor cadastra duas turmas, Turma 1 e Turma 2, para a escola Ucs.

Figura 24 – Cadastro de Turma



Ao acessar a opção Cadastro de Alunos, o professor terá uma tela similar ao cadastro de turmas, com a lista dos alunos já cadastrados, as opções de buscar por um aluno cadastrado e adicionar um novo aluno. Ao adicionar um novo aluno, o professor deverá inserir a escola e turma as quais o aluno pertencerá, o nome do aluno, o sexo (Masculino ou Feminino) e a data de nascimento. A Figura 25 mostra a sequencia de telas deste processo. A Figura também mostra um exemplo de uso para cadastrar alunos. No exemplo são cadastrados os alunos 1 e 2, cada um em uma turma diferente, da mesma escola. Para cadastrar um aluno é obrigatório ter pelo menos uma escola e uma turma cadastrada no aplicativo.

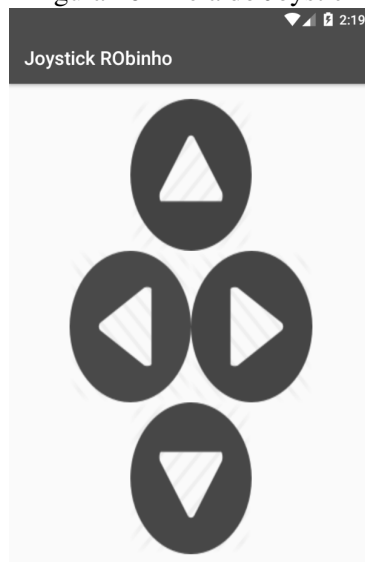
Figura 25 – Cadastro de Aluno



A tela do Joystick, Figura 26, apresenta um controle simples com quatro botões de setas, indicando as quatro direções que o robô pode se mover. A cada botão pressionado o aplicativo

já envia o comando para o robô, permitindo que ele seja controlado diretamente pelos alunos e professores.

Figura 26 – Tela do Joystick



Ao escolher a opção Consultas, o professor será direcionado a tela de consultas, Figura 27, onde poderá definir filtros para sua busca. Dentre os filtros disponíveis temos, data inicial e final da resolução dos cenários pelas crianças, cenário desejado, status do cenário, aluno que resolveu o cenário, gênero do aluno e faixa etária dos alunos. Todos os filtros também possuem a opção para selecionar todos. Após selecionados os filtros e realizada a busca, será exibida a tela de resultados, com todos os resultados existentes de acordo com os filtros selecionados. A Figura 27 também mostra um exemplo de busca que retorna todos cenários resolvidos por todos os alunos. Para obter estes resultados, mostrados na segunda imagem, foram utilizados os filtros mostrados na primeira imagem da Figura. Todos os resultados são ordenados por ordem cronológica do mais antigo ao mais recente.

### 3.2 Visão Geral do Software Atualizado

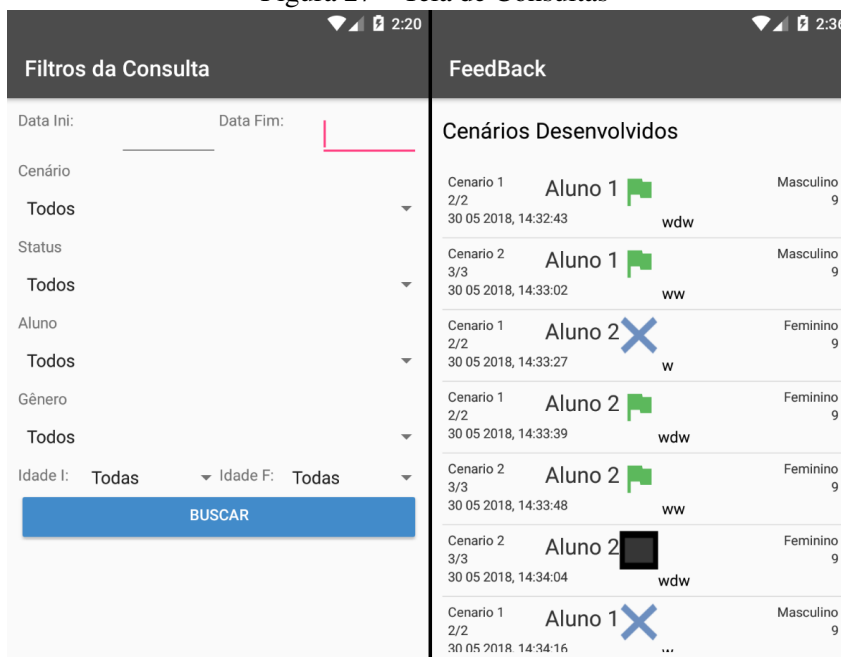
Na versão anterior do software, os dados gerados ficavam salvos apenas localmente em um banco de dados SQLite, no dispositivo em uso. Caso o dispositivo venha a apresentar problemas ou se o professor desejar trocar o dispositivo, todos os arquivos do aplicativo deverão ser passados manualmente de um dispositivo para o outro.

Para sanar este problema foi proposto como solução, um software que permita ao professor salvar seus dados em um banco de dados na nuvem e também acessá-los pelo aplicativo em outro dispositivo, além de prover ao professor meios para o gerenciamento das atividades da turma.

Desta forma, O software desenvolvido adicionou duas funcionalidades ao aplicativo existente. Uma delas permite ao professor monitorar o processo de aprendizagem por parte dos



Figura 27 – Tela de Consultas



alunos. O professor também pode ver suas turmas, quais alunos estão em suas turmas e também a situação de cada aluno em relação às tarefas previamente distribuídas. Outra funcionalidade desenvolvida foi a criação de um banco de dados na nuvem para o professor salvar seus dados. Ambas as funcionalidades foram desenvolvidas em Java, através da IDE Android Studio.

O software de gestão foi desenvolvido como uma nova opção no menu principal com o nome de Gerenciamento, conforme mostra a primeira imagem da Figura 28. Ao selecionar esta opção, o professor é levado a primeira tela, segunda imagem da Figura 28, que apresenta todas as turmas existentes com o número de alunos presente em cada turma e uma média geral da turma. Através desta lista, o professor pode observar quais turmas estão aprendendo as lições mais facilmente e quais estão com mais dificuldades através da média.

A média é calculada pelo total de tentativas bem sucedidas de resolução de cenário, de todos os alunos, dividido pelo número total de tentativas de todos alunos. Uma turma com média baixa pode indicar que os alunos podem estar com dificuldades no aprendizado e estão precisando de muitas tentativas para realizar um cenário até encontrar a solução. Por outro lado, uma turma com média alta indica que os alunos estão aprendendo com facilidade, encontrando as soluções dos cenários em poucas tentativas.

No topo da tela, o professor tem disponível um campo para realizar uma busca pela turma desejada, bastando inserir o nome da mesma. Assim, o professor pode buscar por uma turma rapidamente quando existirem muitas turmas cadastradas.

Ao selecionar uma turma, o professor tem informações mais detalhadas sobre a mesma, como a lista dos alunos e a quantidade de cenários tentados e a quantidade de cenários resolvidos com sucesso por cada aluno, conforme mostra a terceira imagem da Figura 28. A quantidade de cenários tentados representa a soma de todas as tentativas de todos os cenários resolvidos pelo

aluno, podendo existir diversas tentativas para o mesmo cenário. Da mesma forma, a quantidade de cenários resolvidos com sucesso representa a soma de todas as soluções encontradas pelo aluno, também podendo existir mais de uma solução encontrada para o mesmo cenário.

Com estes dados o professor pode observar quais alunos tem um aprendizado mais fácil e rápido, e quais estão encontrando dificuldades na resolução dos cenários. Uma diferença pequena entre a quantidade de tentativas e a quantidade de soluções indica que o aluno tem facilidade em aprender. Por outro lado, uma grande diferença entre tentativas e soluções pode indicar dificuldades no aprendizado.

No topo desta tela, também está disponível um campo para realizar uma busca pelo aluno desejado, bastando inserir o nome do mesmo. Assim, o professor pode buscar por um aluno rapidamente quando existirem muitos alunos na turma.

Ao selecionar um aluno, o professor tem os detalhes de quais foram os cenários que o aluno tentou e o resultado daquela tentativa, conforme mostra a quarta imagem da Figura 28. Dentre os resultados possíveis para a situação do cenário existem:

- **OK:** Significa que o aluno encontrou com sucesso a solução para o cenário.
- **Não encontrou a solução:** Significa que o aluno não encontrou a solução, faltando alguns comandos para o robô chegar ao objetivo.
- **Extrapolou o Limite:** Significa que os comandos do aluno levaram o robô a sair dos limites do cenário.
- **Rato caiu no buraco:** Significa que os comandos levaram o robô a atingir o obstáculo "buraco" durante sua execução.

Figura 28 – Telas do Software de Gestão

Gerenciamento de Turmas				Turma 1			Aluno 3	
PESQUISAR				PESQUISAR			Cenários	Situação
Turmas	Alunos	Média		Alunos	Cenários Tentados	Resolvidos com Sucesso	Cenario 1	Não encontrou a solução
1 Turma 1	2	50%		1 Aluno 1	4	2	Cenario 1	OK
2 Turma 2	2	67%		5 Aluno 3	4	2	Cenario 2	Extrapolou Limite
							Cenario 2	OK

### 3.3 Persistência dos Dados

O banco de dados na nuvem foi desenvolvido utilizando a plataforma Firebase, provida pela Google (FIREBASE, 2017). Esta plataforma foi escolhida por oferecer os serviços que o projeto necessita, estar disponível para uso imediato e oferecer um plano de uso gratuito. O Firebase disponibiliza diversas ferramentas para desenvolver aplicativos de alta qualidade e ampliar a base de usuários, desde ferramentas de autenticação de usuário, até ferramentas que monitoram a atividade do aplicativo. Para desenvolver a funcionalidade do banco de dados foi utilizada apenas a ferramenta Realtime Database, que permite armazenar e sincronizar dados com um banco de dados na nuvem. Os dados são armazenados na nuvem no formato JSON e são sincronizados com todos os clientes em tempo real. Os dados também ficam salvos localmente, para que permaneçam disponíveis quando o aplicativo está desconectado da internet (offline).

O formato JSON é um modelo para armazenamento e transmissão de informações em forma de texto. A ideia utilizada pelo JSON para representar informações é bastante simples. Para cada valor representado, atribui-se um nome, também conhecido como rótulo, que descreve o seu significado. Para representar um dia do mês, por exemplo, pode-se utilizar o rótulo "dia" associado ao número do dia, formando a seguinte sintaxe: "dia": 12. Da mesma forma, é possível representar dados mais complexos, com diversos valores, utilizando um conjunto de rótulos associados a um conjunto de valores, entre chaves. Para representar um livro, por exemplo, pode ser utilizada a seguinte sintaxe:

```
{  
  "titulo": "O Senhor dos Anéis",  
  "ano": 1995,  
  "genero": ["aventura", "fantasia"]  
}
```

Um ponto importante a ser ressaltado sobre o banco na nuvem, é de que ele não substitui o banco de dados SQLite local. A principal função do banco de dados na nuvem é servir como base para a sincronização de todos os bancos de dados locais, independentes em cada instância da aplicação. Dessa forma, sempre que o aplicativo é iniciado, todos os dados locais serão sincronizados com os dados na nuvem, ocorrendo o mesmo quando algum dado é acrescentado, editado ou removido do banco na nuvem. Desta forma, todas as instancias da aplicação sempre terão os mesmos dados em seus bancos de dados locais.

Dentre os principais recursos oferecidos pelo Realtime Database temos:

- **Sincronização em tempo real:** sempre que os dados são alterados, todos os dispositivos conectados recebem essa atualização.
- **Responsividade mesmo off-line:** o Firebase Realtime Database mantém seus dados em disco quando estiver sem conexão. Quando a conectividade é restabelecida, o dispositivo

recebe as alterações perdidas e faz a sincronização com o estado atual do banco de dados.

- **Acessível em dispositivos clientes:** o Firebase Realtime Database pode ser acessado diretamente de um dispositivo móvel ou navegador da Web, sem um servidor de aplicativos. A segurança e a validação de dados são feitas por meio de regras de segurança baseadas em expressão do Firebase Realtime Database, executadas quando os dados são lidos ou gravados.

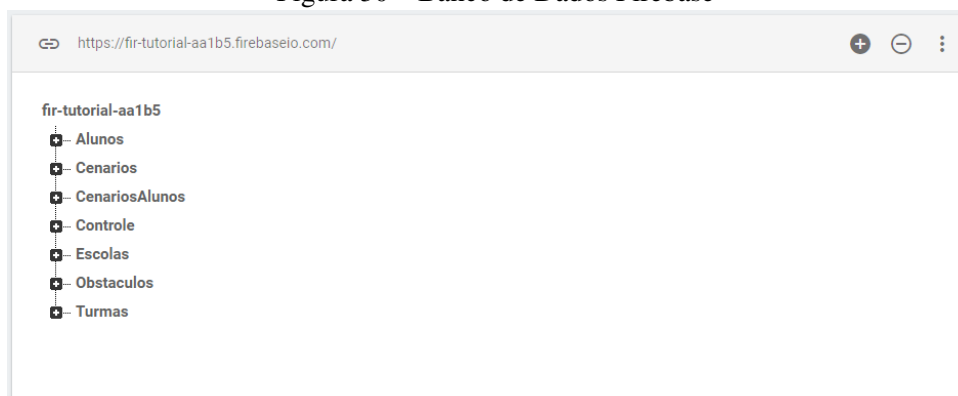
Ao criar o banco Realtime Database, foi necessário criar regras de segurança para o banco de dados. Essas regras permitem definir quem pode ler e escrever dados no banco, bem como validar se os dados a serem salvos possuem a formatação correta para o banco. Conforme mostra a Figura 29, as regras criadas para o banco desta aplicação definem que todos que possuem a referência do banco poderão ler e gravar dados no mesmo. Por padrão, as regras são definidas para acesso completo de leitura e gravação ao banco de dados somente a usuários autenticados, porém, como o banco foi utilizado apenas para testes desta aplicação e o aplicativo não possui a função de autenticar usuários, foi dada liberdade total de escrita e leitura. Estas regras não são recomendadas para uma aplicação real.

Figura 29 – Regras de Segurança

```
1 {
2   "rules": {
3     ".read": true,
4     ".write": true
5   }
6 }
```

Durante o desenvolvimento do banco, buscou-se criá-lo de forma similar ao banco de dados SQLite já utilizado pela versão anterior da aplicação. Após criado o banco e definidas as regras, é gerado um link que representa uma referência a raiz do banco de dados, conforme mostra a Figura 30. Na raiz do banco, foram criadas sete nodos filhos, cada qual representando uma tabela do banco, sendo eles Alunos, Cenarios, CenariosAlunos, Escolas, Obstaculos, Turmas e Controle, onde cada dado é salvo dentro de seu respectivo nodo pai. Cada nodo criado também possui sua própria referência, formada pela referência da raiz do banco mais a sintaxe "/NomeDoNodo", o que permite uma maior facilidade e rapidez na hora de acessar os dados em um dos nodos. Por exemplo, na implementação das funções de leitura e escrita do banco, são utilizadas as referências diretas para cada nodo pai, uma delas é a referência do nodo Alunos, que possui a sintaxe: <https://fir-tutorial-aa1b5.firebaseio.com/Alunos/>.

Figura 30 – Banco de Dados Firebase



Para fazer a integração da aplicação com o banco, a IDE Android Studio oferece um assistente para conexão com o Firebase, para tornar este processo simples e rápido. Basta inserir a conta Google utilizada para criar o banco de dados que todos os bancos Firease criados naquela conta aparecerão para escolha. Após selecionar o banco criado, a IDE fará automaticamente todo o processo de adicionar os pacotes de programas necessários e a integração do banco com o projeto do aplicativo.

Todos os dados são salvos no formato de um HashMap, que é basicamente uma lista onde cada elemento possui uma chave e um valor associado, muito similar ao JSON. Por exemplo, ao cadastrar um aluno no aplicativo, será criada uma lista de chave-valor com as mesmas informações que serão inseridas no banco de dados local. A lista gerada então terá um dos elementos com a chave "Nome" associada a um valor, que nada mais é do que o nome do aluno. Esta lista então será inserida como um filho no nodo Alunos, utilizando um valor de identificação único para o aluno.

A Figura 31 mostra a estrutura de dados de cada tabela do banco Firebase. Estas estruturas de dados são as mesmas utilizadas pelo banco de dados local, de forma que o banco de dados na nuvem torna-se um reflexo idêntico do banco de dados local. Pode-se notar que a cada dado inserido na tabela, é criado um nodo com seu ID. O dado então é inserido neste nodo, com cada valor sendo indicado pelo seu rótulo. Os rótulos podem ser identificados como sendo as palavras em negrito seguida de dois pontos, dentro do nodo, enquanto os valores se encontram ao lado de cada rótulo, entre aspas.

Durante o desenvolvimento e testes do banco de dados foi constatado que o Realtime Database não implementa uma geração de identificadores únicos como o SQLite, mas a geração de identificadores randômicos. O que significa que sempre que dados forem salvos, um identificador aleatório será gerado para aqueles dados. Para manter os dados no banco ordenados como no banco SQLite, foi criada uma tabela de controle, como mostrada na Figura 32. Nesta tabela ficam salvos os próximos identificadores a serem utilizados para salvar cada tipo de dado. Por exemplo, sempre que uma escola for cadastrada, será pego o identificador de escola na tabela de controle para salvar os dados. Depois que os dados foram salvos, o identificador de escola na ta-

bela de controle será incrementado, de forma que a próxima escola cadastrada pegue o próximo identificador. Desta forma, os dados no banco sempre ficarão ordenados pelo identificador.

Figura 31 – Estrutura das Tabelas Firebase

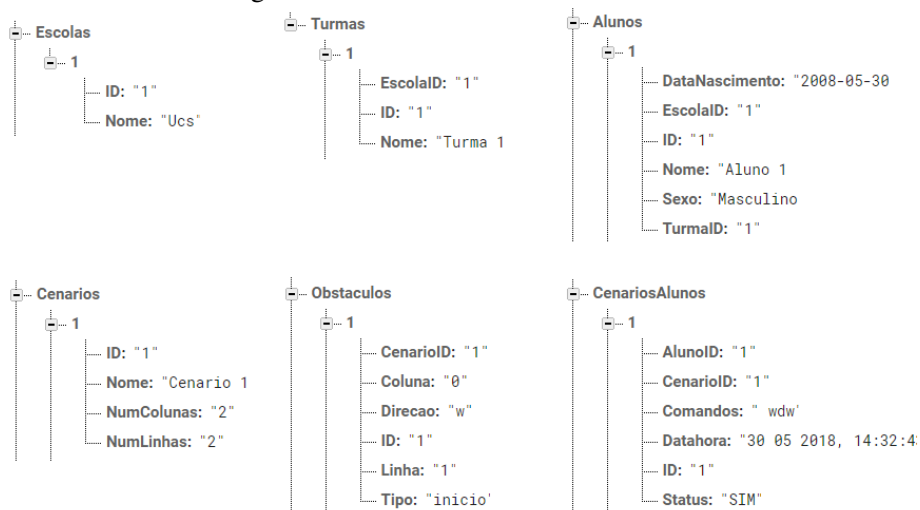
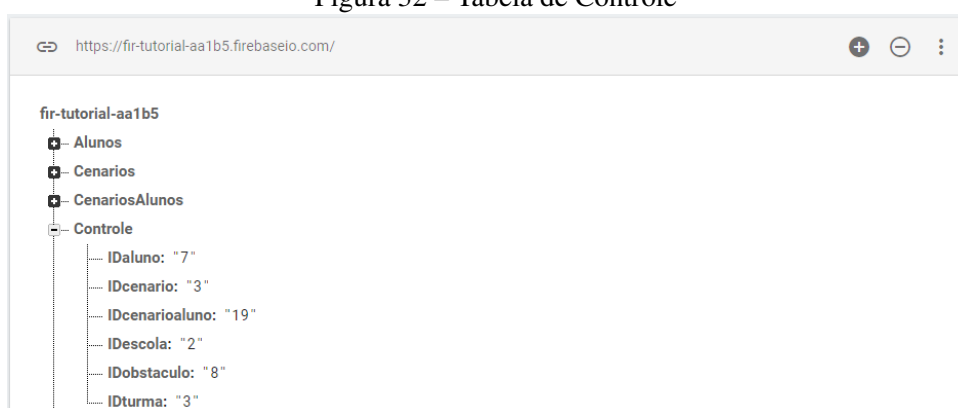


Figura 32 – Tabela de Controle

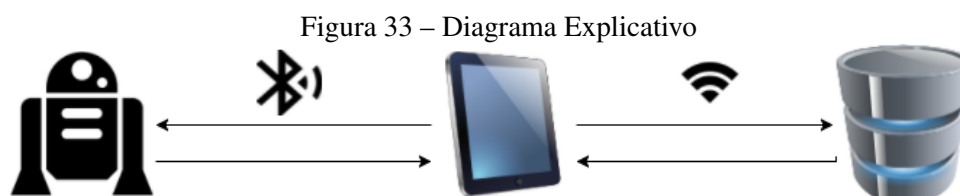


Para fazer a leitura dos dados no banco na nuvem, foi necessário criar funções chamadas de *listeners*, que ficam observando as mudanças nos dados do banco na nuvem. Estas funções são anexadas a uma referência do banco de dados e são chamadas uma vez, na anexação da função a referência do banco, e posteriormente, sempre que ocorrer alguma mudança nos dados dos nodos filhos contidos na referência. Os tipos de mudanças que podem ocorrer nos dados são inserção, alteração e exclusão de dados.

Na aplicação implementada, foram criados seis *listeners*, um para cada referência de tabela, e são chamados uma vez quando a aplicação inicializa e, durante a execução, sempre que um dado é inserido, alterado ou excluído do banco na nuvem. Quando um dado é inserido no banco na nuvem, o *listener* referente aquele tipo de dado será acionado e verificará se aquele dado inserido já existe no banco local. Se o dado não existe localmente, então a função o adicionará ao banco, caso contrário, o dado já existe no banco local, então nada precisa ser feito. O mesmo

ocorre com a edição e exclusão de dados. Na edição, o *listener* atualizará o dado local, caso ele exista, ou irá inseri-lo caso não exista. Já na exclusão, o dado será simplesmente excluído caso exista.

Portanto, enquanto a comunicação entre o dispositivo e o robô é feita por uma conexão bluetooth, a comunicação entre o dispositivo e o banco de dados será feita por uma conexão wireless, como ilustra a Figura 33.



O uso do software permite ao professor administrar suas aulas de forma satisfatória, possibilitando ao professor observar o andamento da turma em relação as atividades propostas. Assim, o professor pode ver, por exemplo, a média de cada turma quanto ao acerto das resoluções, a média de cada aluno individual, quais cenários ele fez e se acertou a resposta ou não.

### 3.4 Modelagem do Software

Conforme mostra o diagrama da Figura 34, ao iniciar o aplicativo, é verificado se o dispositivo esta conectado a internet. Caso esteja então os dados logo são sincronizados com os dados salvos na nuvem. Caso não esteja conectado, o professor poderá utilizar normalmente o aplicativo, com os dados sendo salvos apenas localmente. Porém, caso alguma mudança tenha sido feita por ele em outro dispositivo, pode vir a ocorrer inconsistência nos dados. Por isso, é essencial que o professor busque sincronizar os dados no começo da aula e ao final.

O modelo de banco de dados utilizado segue o mesmo padrão feito pelo Luan em seu aplicativo, mostrado na Figura 35.

As tabelas do banco de dados são descritas abaixo:

- **Aplicacao:** essa tabela guardará as informações que são necessárias pela aplicação, como a identificação do aluno que está utilizando o aplicativo e a identificação do *Arduino* que está sendo utilizado.
- **Escolas:** essa tabela guardará informações referentes a escolas cadastradas, como a identificação e o nome da escola. Esta tabela também tem como objetivo organizar o aplicativo para o professor, para que ele possa assim acompanhar o aprendizado dos alunos por escola.
- **Turmas:** essa tabela guardará as informações referentes as turmas cadastradas, onde cada turma terá uma identificação, a identificação da escola a qual ela pertence e o nome da turma.

Figura 34 – Diagrama Atividades

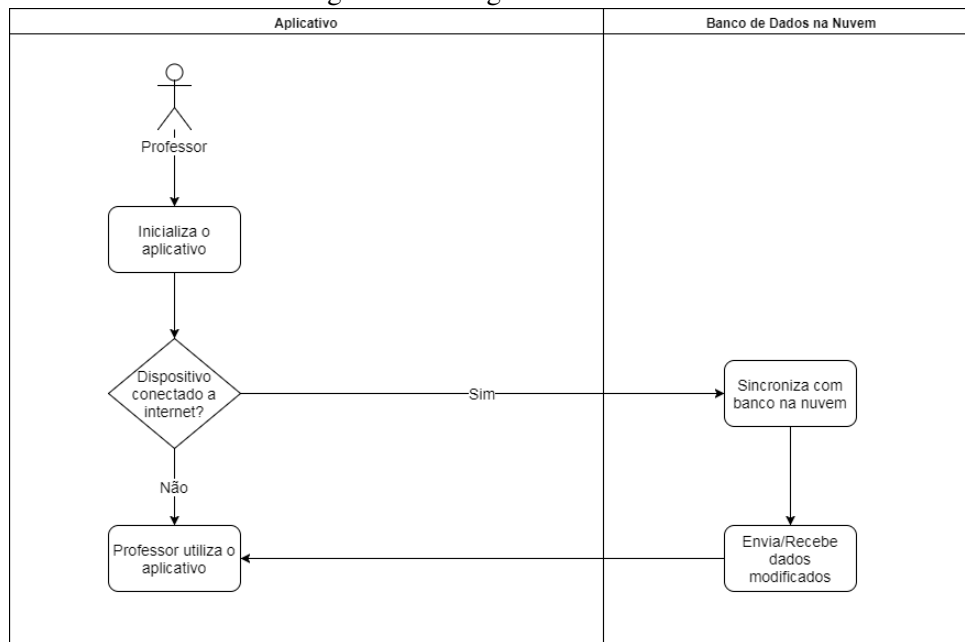
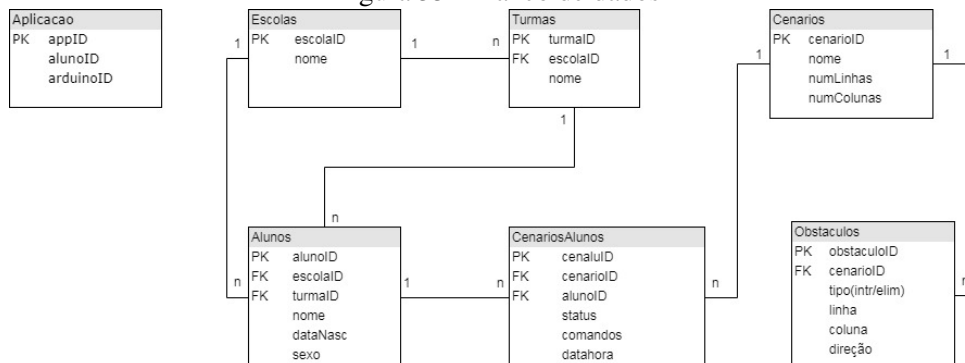


Figura 35 – Banco de dados



- **Cenarios:** essa tabela guardará as informações referentes aos cenários que são cadastrados pelo professor, onde cada cenário terá uma identificação, um nome e o número de linhas e colunas do cenário.
- **Obstaculos** essa tabela guardará as informações referentes aos obstáculos de cada cenário, onde cada cenário terá uma identificação, a identificação do cenário ao qual o obstáculo pertence, o tipo do obstáculo e a posição do obstáculo no cenário. Os tipos definem se o obstáculo é de eliminação ou um intransponível.
- **Alunos** essa tabela guardará as informações referentes aos alunos cadastrados. Cada aluno terá uma identificação, a identificação da escola e da turma a qual o aluno pertence, o nome, a data de nascimento e o sexo do aluno.
- **CenariosAlunos** essa tabela guardará as informações referentes a uma tentativa de resolução de um cenário por um aluno. Cada entrada nesta tabela terá uma identificação, a



identificação do aluno e do cenário, o status do cenário, os comandos utilizados e a data e hora da resolução. Os status refere-se a solução do cenário, se o aluno conseguiu resolver, se faltaram comandos ou se atingiu algum obstáculo.

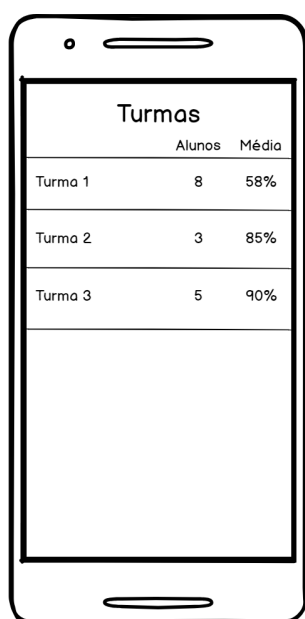
Todas estas tabelas possuem suas versões equivalentes no banco de dados na nuvem, contando com a mesma estrutura de dados. Apenas a tabela de Aplicação não é salva na nuvem, devido a seu uso ser essencialmente para o controle da aplicação local, guardando apenas a identificação do aluno que esta utilizando o aplicativo e a identificação do robô que esta sendo utilizado. Por esse motivo, não há razão para salvar estes dados no banco na nuvem e nem enviá-los a outro dispositivo.

### 3.5 Definição dos componentes de Interface

Nesta seção são apresentados os protótipos de tela utilizados como base para desenvolver as telas da nova funcionalidade que se desenvolveu. Para acessar as funcionalidades, o professor utiliza uma opção no menu principal do aplicativo.

Na Figura 36 temos o protótipo da tela de turmas. Nesta tela o professor pode ver todas as turmas as quais ele leciona, quantos alunos a turma possui e também a média de acertos dos alunos incluídos naquela turma. Um acerto seria uma resolução bem sucedida de um cenário pelo aluno.

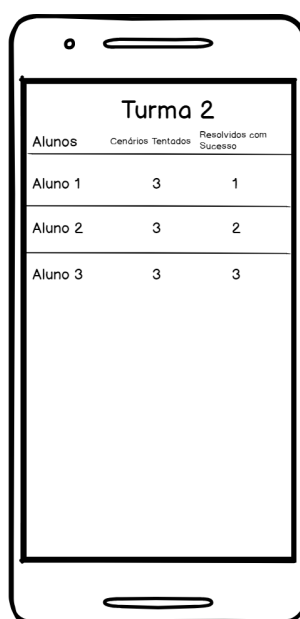
Ao selecionar uma turma na tela anterior, o professor é levado a tela da turma, representada na Figura 37, que apresenta as informação da turma, como o nome da turma, quem são os alunos incluídos na turma, quantos cenários cada aluno tentou resolver e quantos foram resolvidos com sucesso.



Protótipo de tela de turmas. O título é "Turmas". Abaixo dele há uma tabela com duas colunas: "Alunos" e "Média".

	Alunos	Média
Turma 1	8	58%
Turma 2	3	85%
Turma 3	5	90%

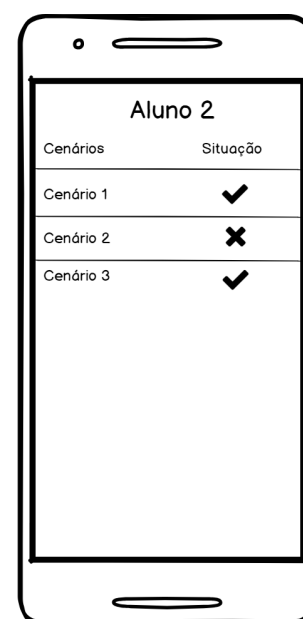
Figura 36 – Tela de turmas



Protótipo de tela da turma. O título é "Turma 2". Abaixo dele há uma tabela com três colunas: "Alunos", "Cenários Tentados" e "Resolvidos com Sucesso".

Alunos	Cenários Tentados	Resolvidos com Sucesso
Aluno 1	3	1
Aluno 2	3	2
Aluno 3	3	3

Figura 37 – Tela da turma



Protótipo de tela Aluno. O título é "Aluno 2". Abaixo dele há uma tabela com duas colunas: "Cenários" e "Situação".

Cenários	Situação
Cenário 1	✓
Cenário 2	✗
Cenário 3	✓

Figura 38 – Tela Aluno

A partir da tela da turma, é possível acessar as informações de cada aluno, que são mostradas na tela do aluno, representada na Figura 38. Nesta tela é mostrado os cenários que o aluno tentou resolver e uma mensagem informando o resultado daquela tentativa.

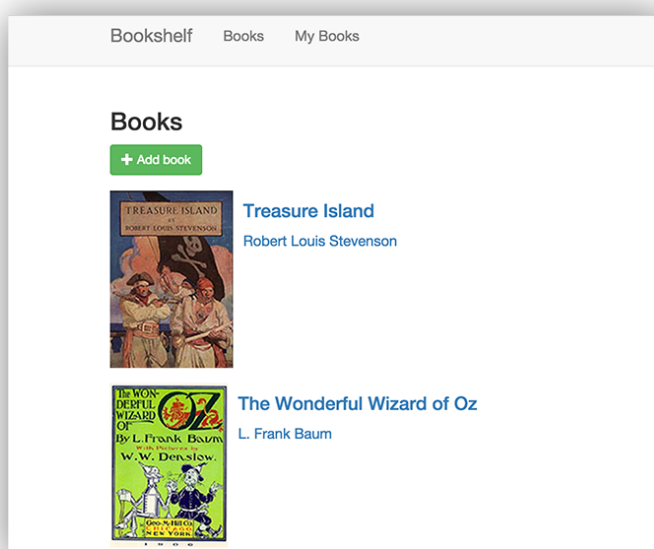
### 3.6 Testes Preliminares de Comunicação Cliente-Servidor

Para a realização deste projeto foi necessária uma ferramenta que conecte o banco de dados do aplicativo com o banco de dados na nuvem. Para isso, foram realizados testes preliminares com algumas ferramentas para tentar criar uma comunicação entre o aplicativo e o banco de dados na nuvem.

A primeira ferramenta testada foi o Google App Engine. Esta ferramenta é uma plataforma para criação de aplicativos Web escaláveis e back-ends móveis. O App Engine, disponibiliza acesso a serviços integrados e APIs, como armazenamentos de dados do NoSQL e uma API de autenticação do usuário, que são comuns para a maioria dos aplicativos. A plataforma também permite escalonar o aplicativo automaticamente de acordo com a quantidade de tráfego que ele recebe.

Para os testes com esta plataforma, foi utilizado um exemplo de aplicação fornecido pela própria Google. Este exemplo, chamado Bookshelf App (Figura 39), trata-se de um aplicativo web que simula uma estante de livros, onde usuários podem a ver lista de livros presentes na estante, adicionar, remover e editar livros. Os usuários também podem enviar uma foto ou imagem para representar as capas dos livros. A aplicação também utiliza um sistema de autenticação de usuário, através das contas Google, que relaciona cada livro adicionado com seu respectivo usuário que o adicionou.

Figura 39 – Bookshelf App



A princípio foi pensado em utilizar esta plataforma para hospedar o banco de dados e o software de gestão, para que assim o professor pudesse acessá-lo de qualquer dispositivo ou computador com acesso à internet. Porém seu uso foi descartado devido a incompatibilidades entre os bancos de dados locais e da nuvem e também devido a complexidade envolvida em criar e hospedar a aplicação. Outro fator que motivou o descarte desta ferramenta foi de que alguns dos recursos que seriam utilizados são recursos pagos, que exigem um pagamento mensal.

A segunda ferramenta testada foi o Heroku. Parecido com o Google App Engine, o Heroku é uma plataforma para criar e implantar aplicações. A plataforma oferece ferramentas para o desenvolvimento de aplicações, de forma que os custos e a complexidade de implantação sejam reduzidos. Dentro da plataforma, essas ferramentas são chamadas de Add-ons e fornecem diversas funcionalidades para a aplicação, como integração com banco de dados, monitoramento das aplicações, implementações de fila, entre outros.

Alguns testes foram feitos com esta plataforma, porém os exemplos encontrados para os testes eram muitas vezes simples demais, não englobando conteúdo suficiente para algo como este projeto. O principal exemplo que mostra como funciona o sistema de implantação de aplicação é basicamente uma aplicação web que mostra a mensagem "Hello World" quando acessada. Outros exemplos mostravam como funciona cada Add-on, de forma separada. Cada exemplo cobria apenas um Add-on, sem explicar muito bem como ele poderia ser utilizado em conjunto a outros Add-ons. Por isso, foi optado por descartar esta plataforma, devido a elevada complexidade envolvida no uso da plataforma e a falta de materiais para apoio.

A partir da análise das ferramentas optou-se por descartá-las, devido a elevada complexidade do uso destas e outros fatores, como necessidade de pagamento para o uso de alguns recursos. Com isso, outra ferramenta foi analisada, o Firebase Realtime Database. Esta ferramenta não chegou a ser testada previamente, mas após uma análise feita nos recursos que ela oferece e como ela funciona, ela foi a escolhida para este trabalho, devido justamente a seus recursos, que apesar de limitados são suficientes para este trabalho.

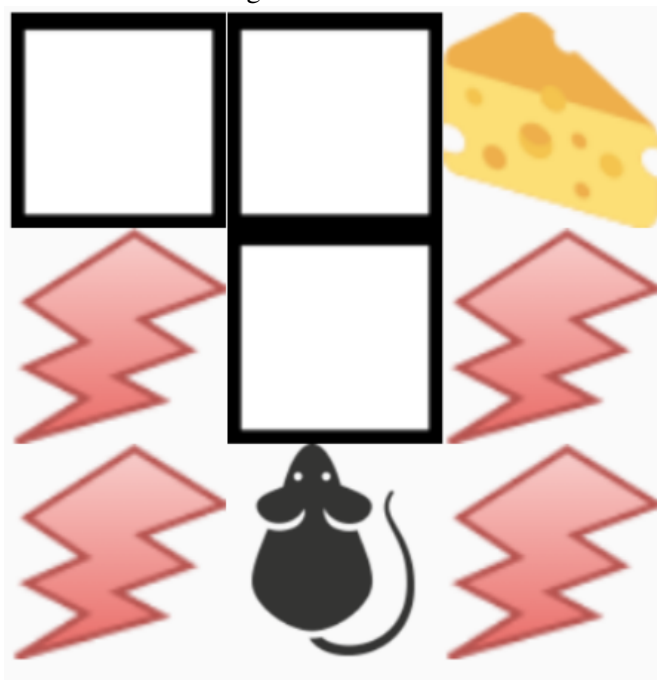
### **3.7 Acompanhamento dos Experimentos na Escola**

Durante o desenvolvimento deste trabalho, foram acompanhados dois experimentos, nos dias 18/10/17 e 25/10/17, com os alunos da escola bilíngue Santa Mônica, com o objetivo de realizar testes de software e hardware do robô Rob e também com os alunos. Durante os testes foram avaliados a recepção das crianças com o hardware e também a compreensão delas na realização das tarefas, tomando o tempo em que cada uma demora para realizar a tarefa.

No primeiro experimento as crianças foram apresentadas ao robô, onde cada uma delas brincou um pouco com robô, controlando-o pelo joystick. Após a brincadeira, foi apresentado às crianças a primeira tarefa, resolver o labirinto da Figura 40.

Nesta tarefa, algumas crianças conseguiram realizá-la em pouco tempo, enquanto as outras demoraram alguns minutos para chegar na solução. A partir desta tarefa foi sendo introduzido

Figura 40 – Tarefa 1



os conceitos de lógica e pensamento computacional, já que para resolver a tarefa as crianças deveriam criar um programa, uma sequência de comandos com todas as instruções para a resolução da tarefa, cujo resultado só pode ser observado em ação no robô ao final da programação, diferente de quando elas controlavam com o joystick e logo viam em tempo real o resultado do comando o qual elas tocavam.

Na segunda tarefa, Figura 41, foi adicionado um segundo objetivo, outro queijo, que deveria ser pego também para completar o cenário. Nesta tarefa as crianças demoraram mais tempo para chegar na solução, pois, como não havia o comando de andar de ré, elas deveriam girar o robô em 180 graus. Mas como os comandos de virar o robô giravam apenas em 90 graus, elas demoraram para compreender que deveriam ser utilizados dois comandos de giro seguidos. Algumas das crianças demonstraram um bom pensamento lógico ao mencionarem que era só repetir os comandos da primeira tarefa e depois virar o robô.

No segundo experimento, realizado uma semana após o primeiro, foi elaborado uma tarefa mais complexa que as anteriores, Figura 42, com o robô voltado de frente para a criança, criando assim um movimento espelhado no momento de movimentar o robô. Como resultado, esta tarefa foi a mais demorada para ser resolvida pelas crianças, onde elas precisaram ser orientadas frequentemente. Uma das orientações dadas as crianças foi para observarem o problema de outro ângulo, podendo se mover livremente pelo ambiente físico, com o intuito de as fazerem se sentar atrás do robô para compreenderem melhor o labirinto. Algumas delas adotaram essa orientação e buscaram observar o labirinto de todos os lados até compreenderem que a melhor posição era atrás do robô. Outras preferiram ficar onde estavam e surpreendentemente mostraram compreender o movimento espelhado e conseguiram resolver o problema.

Figura 41 – Tarefa 2

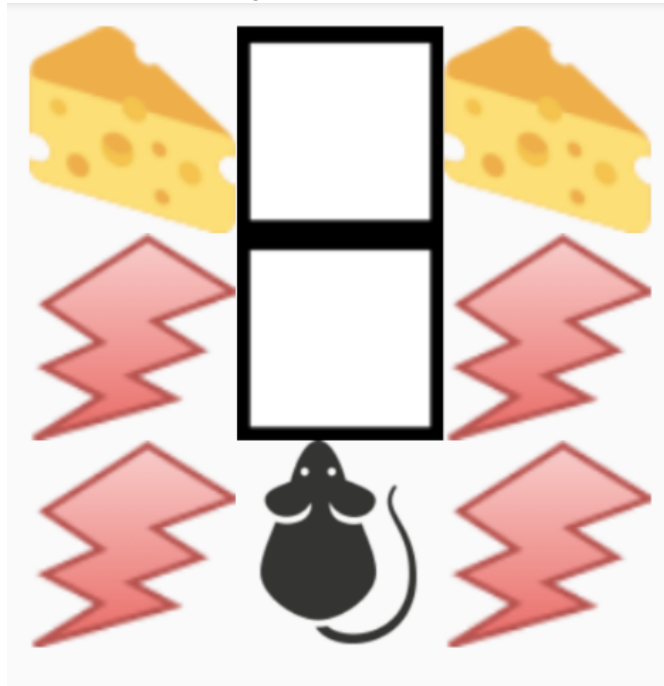
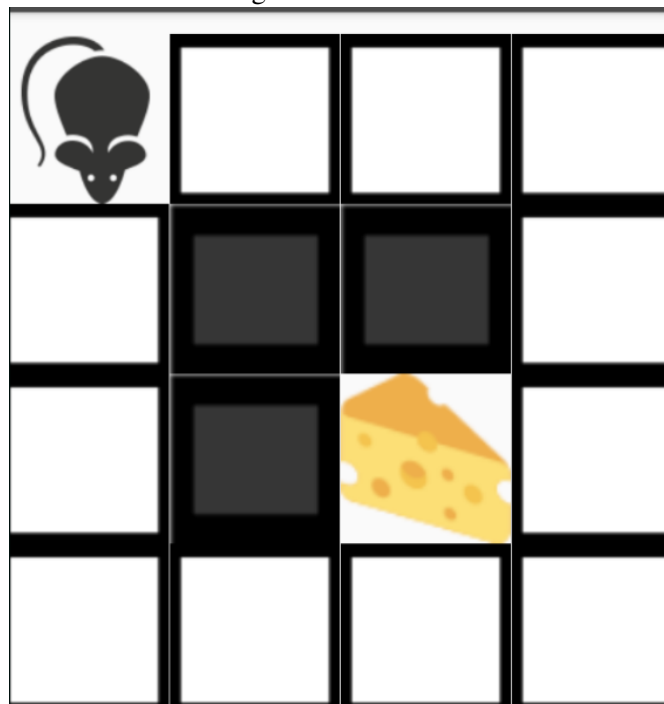


Figura 42 – Tarefa 3



Após, foi realizada novamente a segunda tarefa do primeiro experimento, para observar se as crianças puderam compreender melhor o conceito do pensamento computacional com a tarefa mais complexa e se elas conseguiam realiza-la em menor tempo. Como resultado, muitas crianças conseguiram resolver a tarefa em menos tempo do que no primeiro experimento, necessitando menos orientação dos professores.

A Figura 43 mostra o tempo tomado de cada um dos alunos nas tarefas realizadas, bem como algumas observações sobre a resolução de alguns alunos.

Ao observar a tabela de resultados, alguns pontos importantes são ressaltados:

- Pré e Jardim não tiveram grandes diferenças em seus tempos. Mesmo com os alunos do pré sendo um pouco maiores, seus tempos foram semelhantes aos alunos do jardim.
- Meninos e meninas tiveram seus tempos semelhantes.
- Notória melhora de uma semana para outra na tarefa 2. Após a primeira aula, na segunda oportunidade os alunos resolveram rapidamente a segunda tarefa. Está que no primeiro dia foi o grande desafio.
- O tempo da tarefa 3 mostra que é possível aumentar a dificuldade da aplicação apenas alterando a orientação do ratinho no cenário.
- Durante a resolução da tarefa 3, a maioria dos alunos mudou seu ponto de vista sobre o cenário, o que os levou a compreender mais facilmente o problema.

Figura 43 – Tempos dos Alunos nos Cenários

Planilha de alunos	10/18/17			10/25/17		
	Tarefa 1	Tarefa 2	Observações sobre Tarefa 2	Tarefa 3	Tarefa 2	Observações sobre Tarefa 3
<b>turma do pré</b>						
Martina	00:01:11	00:00:38	Faltou 1 passo	00:05:36	00:02:23	mudou de posição
Enrico	00:02:05	00:04:30	duas tentativas	00:03:33	00:02:13	mudou diversas vezes de posição
Antonio	00:00:53	00:00:42	sem auxílio	00:05:44	00:00:41	mudou diversas vezes de posição
Jordana	00:01:02	00:02:47	duas tentativas	00:02:25	00:01:04	mudou de posição
Caetano	00:02:19	00:02:44		00:06:11	00:04:47	mudou de posição
<b>turma do jardim</b>						
Sofia	00:01:14	00:03:31			00:01:21	
Vinicius	00:01:02	00:03:19				
Alice	00:01:12	00:04:04			00:02:19	
Antônia	00:01:20	00:01:19				
Guilherme	00:01:03	00:02:45			00:01:00	
Maitê	00:01:39	00:01:52		00:04:03	00:04:47	não mudou de posição
Vitor	00:00:55	00:05:40		00:02:18	00:02:06	não mudou de posição
Eduarda	00:01:49	00:02:07		00:02:10	00:01:41	não mudou de posição
Germano	00:01:06	00:02:19				
Bruce	00:02:07	00:02:50				
Pedro	00:06:14	00:06:16				
Matheus+Dora	00:44:00					

### 3.8 Considerações Finais

A fim de avaliar o projeto proposto foi realizado um experimento com turmas do ensino primário. A princípio, o experimento contaria com algumas aulas utilizando o robô, sendo que em cada aula seria utilizado um dispositivo diferente ou também o mesmo dispositivo mas com os dados da aula anterior removidos, com o intuito de testar se o aplicativo salvou os dados no banco de dados na nuvem corretamente na aula anterior e também se o aplicativo consegue carregar estes dados corretamente a partir do banco de dados na nuvem. Também

seriam testadas as funcionalidades de gestão da turma, onde seria observado, ao final da aula, a situação de cada aluno em relação as tarefas realizadas.

Porém devido a falta de tempo e disponibilidade, não foi possível realizar mais de um experimento. Por isso, durante o experimento em sala de aula, foram coletados os dados necessários para uma simulação de uso do aplicativo, testando as novas funcionalidades.

Na Figura 44 temos os dados coletados e utilizados para a simulação, salvos no banco de dados na nuvem. Para simular a sincronização do banco de dados com dispositivos foram utilizados três dispositivos diferentes (dispositivos A,B e C). Primeiramente, foram utilizados os dispositivos A e B para cadastrar a escola, turmas, alunos e cenários, alternando o uso de cada dispositivo.

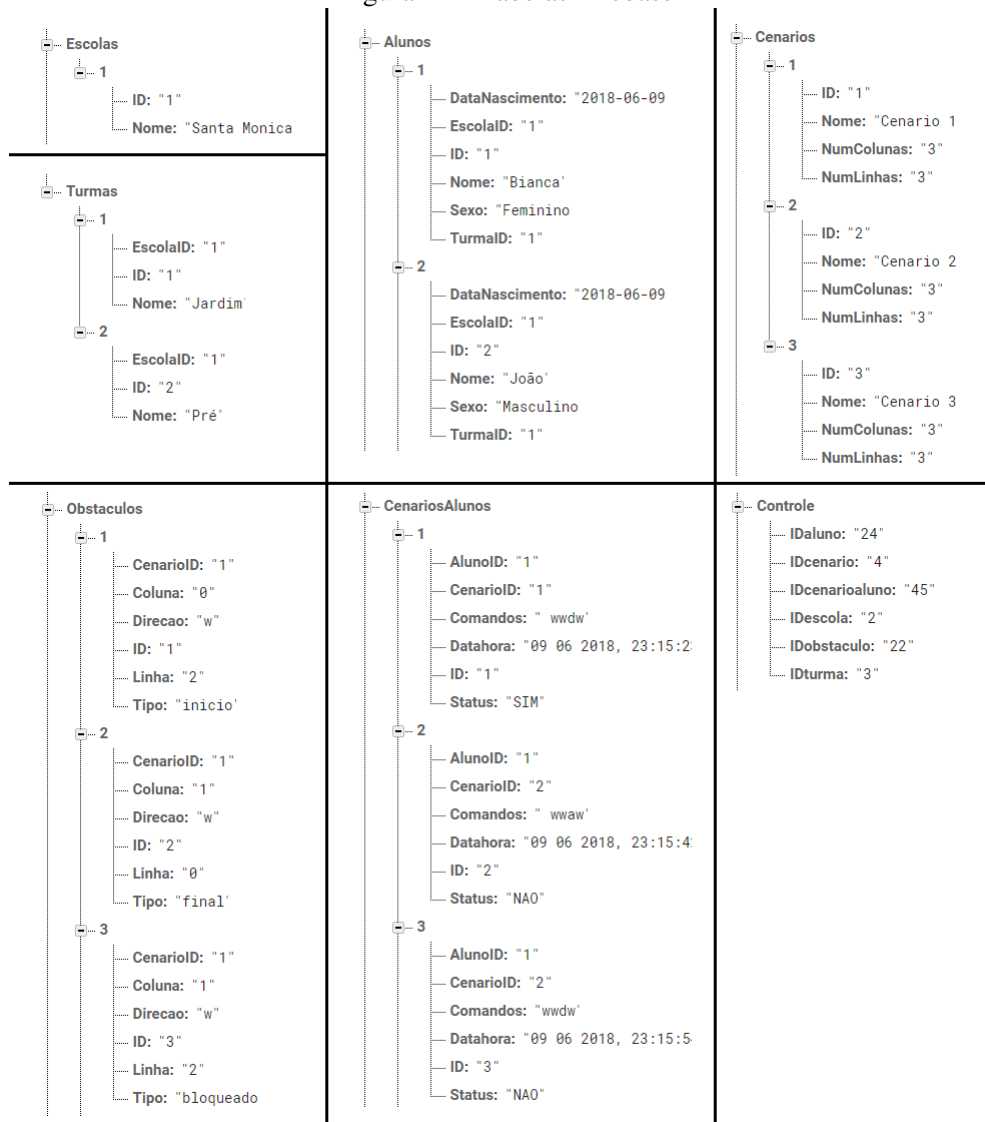
Com isso, foi possível observar que os dados inseridos em um dos dispositivos, logo eram gravados também no outro, mantendo os dois dispositivos sempre com os dados sincronizados entre si e com o banco de dados na nuvem. Depois que todos os dados foram inseridos, o aplicativo foi instalado no dispositivo C, com o qual foi verificado que todos os dados inseridos no banco na nuvem pelos dispositivos A e B foram baixados e gravados também no dispositivo C, mantendo todos os dispositivos com os dados sincronizados.

Similar ao estudo de Scaradozzi (2015), as atividades que foram desenvolvidas buscam estimular a lógica e o raciocínio lógico nas crianças. Estas atividades foram oferecidas na forma de tarefas extra classe com robótica. Através da programação dos alunos, os professores podem monitorar e observar se há uma melhora no desempenho dos alunos em sala de aula.

Em outro estudo similar, de Goran Zaharija (2015), os experimentos elaborados foram bastante parecidos com os realizados na escola bilíngue Santa Mônica. As crianças eram observadas enquanto aprendiam a mexer no robô, durante as tarefas e após elas eram perguntadas sobre como tinha sido a experiência com o robô e com a programação.

Outro estudo mais elaborado, feito por Balaji (2015), foi feito voltado para crianças mais velhas, estudando no ensino médio, para mostrar aos alunos uma visão positiva das oportunidades de carreira com a robótica. Neste estudo, as atividades e conceitos ensinados foram mais complexos, ao longo de várias aulas. Contudo, o método para avaliar o estudo foi parecido com os outros estudos. Foi também utilizado o método de observar os alunos durante as tarefas e também acompanhá-los durante sua experiência com as tarefas e o robô.

Figura 44 – Tabelas Firebase





## 4 CONCLUSÃO

### 4.1 Síntese do Trabalho

Ao longo deste trabalho foi realizado uma pesquisa sobre a importância da inserção da lógica e do pensamento computacional na educação infantil, buscando compreender como acontece esse tipo de ensino nos dias de hoje e quais são as dificuldades encontradas por alunos e professores durante o ensino. Após alguns estudos, foi constatado que essas habilidades são essenciais não apenas a pessoas ligadas a área da informática, mas para todo mundo. Tais habilidades nos ajudam a resolver problemas, a decompor tarefas complexas em várias tarefas mais simples e melhoram nosso planejamento.

Também foi constatado que o momento ideal para o desenvolvimento dessas habilidades é a partir da infância, quando as crianças estão ávidas por conhecer as coisas e começam a aprender a usar um computador. Porém, ao observar como a tecnologia é empregada no ensino nas escolas, nota-se que há muito pouca contribuição para o desenvolvimento do raciocínio lógico e do pensamento computacional.

Contudo, com o surgimento de novas tecnologias, novas formas e meios de aplicar esse tipo de ensino vem surgindo. Uma dessas tecnologias são as plataformas de ensino de programação, que buscam fornecer recursos para o ensino de programação de forma simples e intuitiva, para pessoas de qualquer idade. Ao longo deste estudo foram analisadas cinco destas plataformas, buscando compreender os principais recursos oferecidos por elas. Recursos como, o tipo da linguagem de programação, como as atividades são passadas ao alunos, como eles podem resolve-las e como o professor pode acompanhar o aprendizado de seus alunos.

Uma das ferramentas estudadas foi o Rob The Mouse, desenvolvido por alunos da Universidade de Caxias do Sul (UCS), que tem por objetivo promover a inserção da lógica computacional no ensino primário. Após a análise da plataforma, foi apontada a necessidade de um recurso que permita o professor administrar suas turmas e acompanhar as tarefas de seus alunos, bem como a necessidade de um banco de dados na nuvem, onde ficam salvos todos os dados gerados através da utilização do aplicativo.

Com base nestas necessidades foram desenvolvidas duas funcionalidades que buscam supri-las. Uma das funcionalidades desenvolvidas trata-se de um software de gestão, adicionado na forma de uma opção no menu principal do aplicativo, onde o professor pode visualizar de forma simples os dados pertinentes a gestão de suas turmas, como quais são suas turmas, quantos alunos cada turma possui e a média geral dos alunos. O professor também pode ver quantos e quais foram os cenários resolvidos pelos alunos, bem como o resultado obtido pelo aluno na resolução do cenário.

Outra funcionalidade adicionada foi o banco de dados na nuvem, utilizando a ferramenta Realtime Database. Com a integração do aplicativo com esse banco na nuvem todos os dispositivos que estejam rodando o aplicativo, sempre terão seus dados sincronizados com os dados

no banco na nuvem. Dessa forma, o professor pode utilizar mais de um dispositivo durante a aula, até mesmo simultaneamente se desejar. O banco na nuvem também tem o propósito de agir como um backup de dados, no caso de algum dispositivo apresentar problemas durante a aula.

## 4.2 Contribuições e Resultados

No dia 23/05/18, foi realizado um novo experimento em sala de aula, na mesma escola de educação infantil dos experimentos anteriores, com os mesmos alunos. O experimento teve uma duração de três horas. Durante a aula, as crianças foram avaliadas na percepção dos problemas propostos, na forma como elas chegaram a solução do problema e no tempo necessário para solucionar o problema.

O experimento contou com 23 crianças no total, divididas em duas turmas, sendo 13 crianças na turma do Jardim e 10 na turma do Pré.

A metodologia utilizada foi de dividir os alunos em grupos de três ou quatro para a resolução dos cenários. No primeiro momento foi perguntado às crianças se elas se lembravam do robô e de como ele funcionava. Muitas delas responderam que sim e ficaram bastante animadas para programá-lo novamente.

A primeira turma a ser avaliada foi a do Jardim. Divididas em grupos de três, os alunos foram apresentados ao primeiro cenário, mostrado na Figura 45. O cenário é simples e muitas crianças encontraram a solução na primeira tentativa, em um tempo relativamente curto. Algumas, porém, encontraram dificuldades no momento de girar o rato para pegar o queijo. Elas pensaram que o comando de girar também fazia o rato avançar uma casa para o lado, no mesmo comando.

Para o segundo cenário, foi utilizado um dos cenários do experimento anterior. Neste cenário, um pouco mais complexo, os alunos deveriam pegar os dois queijos utilizando apenas uma sequência de comandos. Alguns alunos, principalmente os do primeiro grupo, tiveram um pouco de dificuldade neste cenário. Porém, foi constatado que a solução que eles encontraram foi a de pegar um queijo de cada vez, utilizando duas tentativas para isso. Em comparação com os tempos tomados em experimentos anteriores, houve uma melhora nos tempos tomados neste experimento.

Devido ao pouco tempo disponível para o experimento e o grande número de alunos, apenas aos dois primeiros grupos da turma do Jardim foi proposto o segundo cenário. Os demais grupos do Jardim receberam apenas o primeiro cenário para resolver.

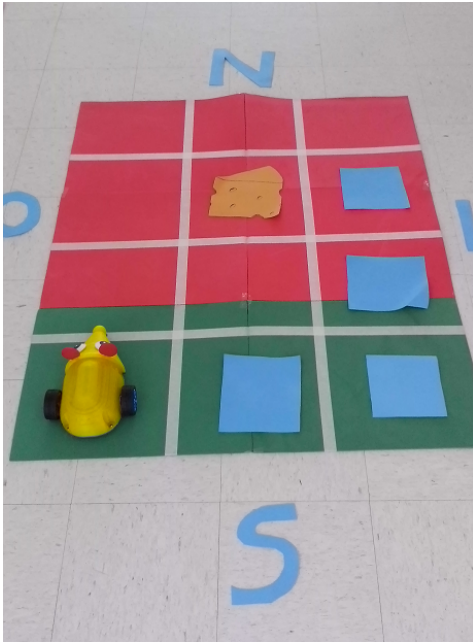


Figura 45 – Cenário 1

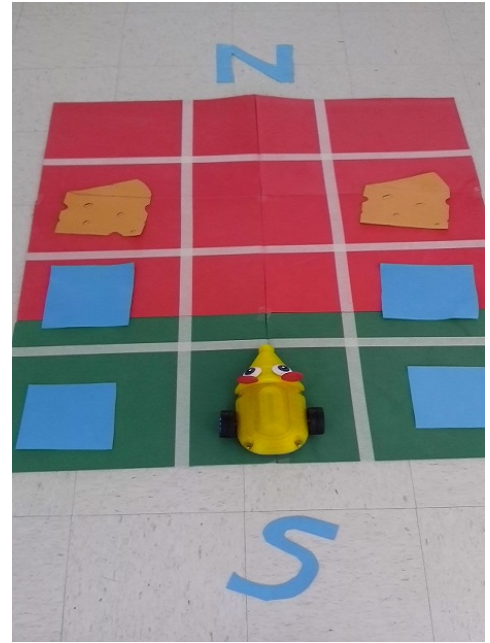
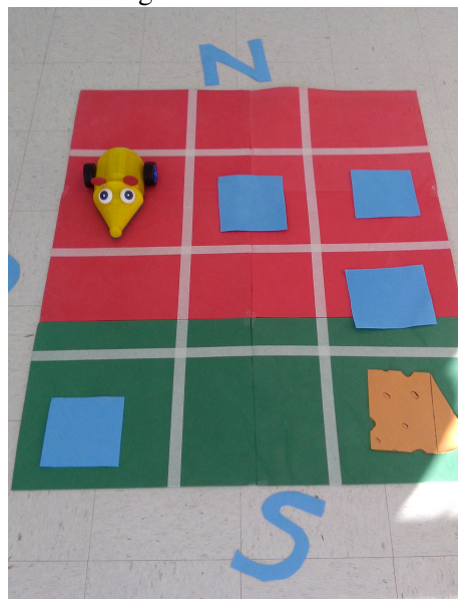


Figura 46 – Cenário 2

Para a turma do Pré foi proposto um cenário mais complexo. Neste cenário o rato se encontra voltado de frente para a criança, criando um movimento espelhado no momento de movimentar o rato. Muitas crianças tiveram dificuldades para entender o problema no início, mas conseguiram solucioná-lo, após serem orientadas a mudarem seu ponto de vista sobre o cenário. Isso as ajudou bastante a compreender os movimentos espelhados do rato.

Figura 47 – Cenário 3



A tabela na Figura 48 mostra os tempos tomados de cada aluno, durante a resolução das tarefas, e também quantas tentativas o aluno precisou para encontrar a solução. Cada tentativa também traz uma observação, mostrando qual foi o resultado daquela tentativa.

Figura 48 – Tempos dos Alunos nos Cenários

	Aluno	Cenário	Tempo	Tentativas		Aluno	Cenário	Tempo	Tentativas	
Grupo 1 - Jardim	Bianca	1	00:42	1 - Resolveu com sucesso	Grupo 5 - Pré	Alice	3	05:25	1 - Comandos não encontraram o objetivo 2 - Comandos não encontraram o objetivo 3 - Resolvido com sucesso	
		2	01:40	1 - Resolveu parcialmente (pegou 1 queijo) 2 - Resolveu parcialmente (pegou o outro queijo)		Germano	3	02:07	1 - Comandos não encontraram o objetivo 2 - Resolvido com sucesso	
	João	1	00:11	1 - Resolveu com sucesso		Mateus	3	04:11	1 - Comandos não encontraram o objetivo 2 - Comandos não encontraram o objetivo 3 - Resolvido com sucesso	
		2	01:26	1 - Resolveu parcialmente (pegou 1 queijo) 2 - Resolveu parcialmente (pegou o outro queijo)		Grupo 6 - Pré	Davi	3	-	1 - Resolveu com sucesso
	Marina	1	00:46	1 - Resolveu com sucesso	Maria Luiza		3	03:02	1 - Resolveu com sucesso	
		2	01:55	1 - Resolveu parcialmente (pegou 1 queijo) 2 - Resolveu parcialmente (pegou o outro queijo)	Guilherme	3	-	1 - Resolveu com sucesso		
	Grupo 2 - Jardim	Inácio	1	02:06	1 - Comandos extrapolam o limite do cenário 2 - Comandos não encontraram o objetivo 3 - Comandos não encontraram o objetivo 4 - Resolvido com sucesso	Grupo 7 - Pré	Vitor	3	01:33	1 - Resolveu com sucesso
			2	03:54	1 - Resolveu com sucesso		Eduarda	3	02:13	1 - Resolveu com sucesso
Olivia		1	01:15	1 - Resolveu com sucesso	Dora		3	00:42	1 - Resolveu com sucesso	
		2	02:26	1 - Resolveu com sucesso	Antonia		3	01:16	1 - Resolveu com sucesso	
Manuela		1	00:21	1 - Resolveu com sucesso						
		2	01:26	1 - Resolveu com sucesso						
Grupo 3 - Jardim	Ana Julia	1	00:42	1 - Resolveu com sucesso						
	Artur	1	00:45	1 - Resolveu com sucesso						
	Isabela	1	00:21	1 - Resolveu com sucesso						
Grupo 4 - Jardim	Gabriel	1	01:53	1 - Comandos não encontraram o objetivo 2 - Resolvido com sucesso						
		1	01:34	1 - Comandos não encontraram o objetivo 2 - Resolvido com sucesso						
	Helena	1	04:28	1 - Comandos não encontraram o objetivo 2 - Comandos não encontraram o objetivo 3 - Resolvido com sucesso						
	Giovana	1	00:20	1 - Resolveu com sucesso						

Após o experimento, os dados coletados foram utilizados para simular o uso do banco de dados na nuvem, bem como a nova funcionalidade de gestão de turmas. A Figura 49 mostra o software de gestão com os dados coletados inseridos. Apartir destes dados pôde-se observar que a turma do Pré teve um desempenho melhor do que o Jardim. Isto se deve ao fato de que muitos alunos do Pré conseguiram resolver o cenário na primeira tentativa, enquanto muitos do Jardim precisaram de várias tentativas para resolver o cenário.

Nas duas imagens da parte de baixo da Figura 49 temos uma comparação entre os alunos de cada turma que possuem o maior número de tentativas para resolver um cenário. Podemos observar que o aluno do Pré, imagem à esquerda, precisou de três tentativas para resolver o cenário 3, que era mais complexo. Enquanto o aluno do Jardim, imagem à direita, precisou de quatro tentativas para resolver o cenário 1, que era mais simples.

Com este trabalho, espera-se oferecer uma ferramenta que ofereça recursos tanto para o professor elaborar aulas e tarefas da melhor forma possível e acompanhar o desenvolvimento do aprendizado de seus alunos, quanto para as crianças aprenderem se divertindo.

### 4.3 Trabalhos Futuros

Com a integração do software com a ferramenta Firebase e o banco de dados na nuvem, uma infinidade de possíveis melhorias futuras surgiram. O Realtime Database, utilizado neste projeto, é apenas uma dentre diversas funcionalidades oferecidas pelo Firebase, que poderiam ser adicionadas a este projeto. Como por exemplo, a funcionalidade de autenticação de usuário, que poderia ser utilizado para definir o acesso ao software para usuários professores e usuários

alunos.

Assim, seria possível utilizar dois dispositivos simultaneamente durante a aula, um com o professor e o outro com os alunos. O professor teria acesso a todas as funcionalidades, enquanto o aluno teria acesso apenas a funcionalidades referentes a resolução de cenários. Uma maior customização de imagens e sons do aplicativo e a utilização de sensores no robô, para testar proximidade, luz do ambiente, também são algumas ideias para o futuro.

Por outro lado se faz necessário também a ampliação dos testes e uma análise estatística dos resultados. Porém, para isso é necessário que sejam realizados mais experimentos com o projeto.

Figura 49 – Software de Gestão

Gerenciamento de Turmas			Pré			Jardim		
		PESQUISAR			PESQUISAR			PESQUISAR
Turmas	Alunos	Média	Alunos	Cenários Tentados	Resolvidos com Sucesso	Alunos	Cenários Tentados	Resolvidos com Sucesso
<sup>1</sup> Jardim	13	55%	<sup>14</sup> Alice	3	1	<sup>1</sup> Bianca	3	1
<sup>2</sup> Pré	10	67%	<sup>15</sup> Germano	2	1	<sup>2</sup> João	3	1
			<sup>16</sup> Mateus	3	1	<sup>3</sup> Marina	3	1
			<sup>17</sup> Davi	1	1	<sup>4</sup> Inácio	5	2
			<sup>18</sup> Maria Luiza	1	1	<sup>5</sup> Olivia	2	2
			<sup>19</sup> Guilherme	1	1	<sup>6</sup> Manuela	2	2
			<sup>20</sup> Vitor	1	1	<sup>7</sup> Ana Julia	1	1
			<sup>21</sup> Eduarda	1	1	<sup>8</sup> Artur	1	1
			<sup>22</sup> Dora	1	1	<sup>9</sup> Isabela	1	1
			<sup>23</sup> Antonia	1	1	<sup>10</sup> Gabriel	2	1
						<sup>11</sup> Manoela	2	1

Mateus		Inácio	
Cenários	Situação	Cenários	Situação
Cenario 3	Não encontrou a solução	Cenario 1	Extrapolou Limite
Cenario 3	Não encontrou a solução	Cenario 1	Não encontrou a solução
Cenario 3	OK	Cenario 1	Não encontrou a solução
		Cenario 1	OK
		Cenario 2	OK

## REFERÊNCIAS

- ALVES, R. M. **Duinoblocks**: desenho e implementação de um ambiente de programação visual para robótica educacional. 2013. 112 p. Dissertação (Mestrado em Informática) - Universidade Federal do Rio de Janeiro, 2013.
- ALVES, R. M.; SAMPAIO, F. F. Duinoblocks: desenho e implementação de um ambiente de programação visual para robótica educacional baseado no hardware arduino. In: CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 3., 2014. **Anais...** [S.l.: s.n.], 2014. p. 11–20.
- COSTA, C.; ALVELOS, H.; TEIXEIRA, L. The use of moodle e-learning platform: a study in a portuguese university. **Procedia Technology**, [S.l.], v. 5, p. 334–343, 2012.
- DIAS, C. D. **Pensamento computacional**: uma relação de proximidade com a matemática e o raciocínio lógico. 2016. 25 p. Dissertação (Licenciatura em Computação) - Universidade Federal de Juiz de Fora, 2016.
- FIREBASE. **Firestore**. Disponível em: <<https://firebase.google.com/docs/?authuser=0>>.
- GOMES, A.; HENRIQUES, J.; MENDES, A. J. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. **Educação, Formação E Tecnologias**, [S.l.], v. 1, p. 93–103, 2008.
- LIMA, L. L. P. de. **Aplicativo educacional para inserção da lógica computacional para crianças**. 2017. 60 p. Dissertação (Bacharelado em em Ciência da Computação) - Universidade de Caxias do Sul, 2017.
- LIPECZ, A. **Codie - cute personal robot that makes coding fun**. Disponível em: <<https://www.indiegogo.com/projects/codie-cute-personal-robot-that-makes-coding-fun>>. Acesso em: 17 março 2017.
- MAKEBLOCK. **Codeybot**: new robot who teaches coding. Disponível em: <<https://www.kickstarter.com/projects/1818505613/codeybot-new-robot-who-teaches-coding>>. Acesso em: 20 setembro 2017.
- MATOS, M. A. E. de et al. Ensinando programação para crianças: um jogo. In: SBC – PROCEEDINGS OF SBGAMES 2016, 2016. **Anais...** [S.l.: s.n.], 2016. v. 15, p. 1210–1213. Disponível em: <<http://www.sbgames.org/sbgames2016/downloads/anais/157629>>. Acesso em: 17 outubro 2017.
- M.BALAJI et al. Robotic training to bridge school students with engineering. In: **PROCEDIA COMPUTER SCIENCE**, 2015. **Anais...** [S.l.: s.n.], 2015. v. 76, p. 27–33. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877050915037722>>. Acesso em: 15 setembro 2017.
- QUEIROZ, R. L.; SAMPAIO, F. F. Duinoblocks for kids: um ambiente de programação em blocos para o ensino de conceitos básicos de programação a crianças do ensino fundamental i por meio da robótica educacional. **XXXVI Congresso da Sociedade Brasileira de Computação**, [S.l.], v. 36, p. 2086–2095, 2015.

RIBAS, E.; BIANCO, G. D.; LAHM, R. A. Programação visual para introdução ao ensino de programação na educação superior: uma análise prática. **Novas Tecnologias na Educação**, [S.l.], v. 14, 2016.

SCAICO, P. D. et al. Ensino de programação no ensino médio: uma abordagem orientada ao design com a linguagem scratch. **Revista Brasileira de Informática na Educação**, [S.l.], v. 21, n. 2, 2013.

SCARADOZZI, D. et al. Teaching robotics at the primary school: an innovative approach. In: SOCIAL AND BEHAVIORAL SCIENCES - 174, 2015. **Anais...** [S.l.: s.n.], 2015. p. 3838–3846. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877042815011817>>. Acesso em: 15 setembro 2017.

SCRATCH. **Scratch**. Disponível em: <<https://scratch.mit.edu/>>. Acesso em: 06 novembro 2017.

TYNKER. **Tynker - coding for kids**. Disponível em: <<https://www.tynker.com/>>. Acesso em: 20 setembro 2017.

VALENTE, J. A. Integração do pensamento computacional no currículo da educação básica: diferentes estratégias usadas e questões de formação de professores e avaliação do aluno. **Revista e-Curriculum**, [S.l.], v. 14, n. 3, p. 864–897, jul/set 2016.

WING, J. Computational thinking: it represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. **“Communications of the ACM**, [S.l.], v. 49, n. 3, p. 33–35, mar 2006.

ZAHARIJA, G.; MLADENOVIC, S.; BOLJAT, I. Use of robots and tangible programming for informal computer science introduction. In: PROCEDIA - SOCIAL AND BEHAVIORAL SCIENCES, 2015. **Anais...** [S.l.: s.n.], 2015. v. 174, p. 3878–3884. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1877042815011878>>. Acesso em: 15 setembro 2017.