UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

TAGARELA: MÓDULO JOGO DE LETRAS E NÚMEROS

ANDRÉ FELIPE FERREIRA

BLUMENAU
2016

ANDRÉ FELIPE FERREIRA

TAGARELA: MÓDULO JOGO DE LETRAS E NÚMEROS

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

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

BLUMENAU
2016

TAGARELA: MÓDULO JOGO DE LETRAS E NÚMEROS

Por

ANDRÉ FELIPE FERREIRA

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

Presidente:	Prof. Dalton Solano dos Reis, Orientador, M.Sc – Orientador, FURB
Membro:	Profa. Joyce Martins M.Sc – FURB
Membro:	Prof. Alexander Roberto Valdameri, M.Sc – FURB

Blumenau, 07 de dezembro de 2016

Dedico este trabalho a minha família e a todos os meus amigos, em especial aos que me ajudaram na realização do mesmo.

AGRADECIMENTOS

Aos usuários do Stack Overflow, sempre dispostos a ajudar os desesperados.

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

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

Conhecimento não é aquilo que você sabe, mas o que você faz com aquilo que você sabe.

Aldous Huxley

RESUMO

A motivação para este trabalho está na utilização de recursos tecnológicos para ajudar na educação de crianças portadoras de necessidades especiais. O uso da tecnologia permite uma maior interação e adaptação a necessidades especificas. Este trabalho é uma extensão ao projeto Tagarela (WIPPEL, 2015), com o principal objetivo de incluir o Jogo de Letras e Números desenvolvido por Reetz (2013). Tanto o Tagarela como o Jogo de Letras e Números foram migrados para a plataforma Ionic. Nesta migração também foram feitas melhorias no Tagarela, como a criação automática das miniaturas para as imagens da prancha e melhorias na interface. O Jogo de Letras e Números também teve de ser adaptado para trabalhar no mesmo contexto do Tagarela, já que este trabalha com planos, que por sua vez representam um conjunto de pranchas, aonde cada prancha representa um conjunto de símbolos (imagem e áudio). Desta forma, basicamente, o jogo consiste na escolha de letras de A a Z, ou números de 0 a 9, sempre representados por um símbolo por prancha. Estes símbolos também têm, além das letras e números, presas (representadas por imagens) que podem capturadas por um predador controlado pelo usuário. Foi possível concluir a migração do Tagarela e do Jogo de Letras e Números, concluindo o objetivo esperado para esse trabalho.

Palavras-chave: Jogo. Tagarela. Ionic. Comunicação assistiva. Motricidade.

ABSTRACT

This assignment is an extension of the project Tagarela (WIPPEL, 2015), as a inclusion of Game of Letters and Numbers (REETZ, 2013). Both application were migrated for Ionic and the Game of Letters and Numbers was integrated with Tagarela. Along with the migration for Ionic were done another improvements in Tagarela, as the automatic creation of the miniatures for the images of board and improvements in interface. The Tagarela mainly work with planes, which represent a group of boards. Besides the migration, the game needed to be adapted for the same operation. The game consist in the choose of letters between A and Z, or numbers between 0 and 9, always represented by a symbol in each board. Besides letters and numbers, these symbols have prey (represented by images) that can be captured by a predator controlled by user.

Keywords: Game. Tagarela. Ionic. Communication. Assistive. Motricity.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AJAX - Asynchronous Javascript and XML

APAE - Associações de Pais e Amigos de Excepcionais

CSS - Cascading Style Sheets

GNU - General Public License

HTML - HyperText Markup Language

JSON - JavaScript Object Notation

MER - Modelo Entidade Relacionamento

PNG - Portable Network Graphics

RF - Requisito Funcional

RGBA - Red Green Blue Alpha

RNF - Requisito Não Funcional

SDK - Software Development Kit

TCC - Trabalho de Conclusão de Curso

UML - Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

Com a expansão da internet e o aumento das tecnologias digitais na sociedade, é esperado um aumento da utilização de recursos tecnológicos para auxiliar na educação (SOUZA, 2013). Pesquisando por essas ferramentas na internet, encontrou-se pouco material que utilize recursos mais recentes, como Canvas e Ionic, sendo a sua maioria digitalização de material tradicionalmente impresso. Aulas modernizadas pelo uso de recursos tecnológicos têm vida longa e podem ser adaptadas para vários tipos de alunos, para diferentes faixas etárias e diversos níveis de aprendizado (SOUZA, 2013).

Muito se fala hoje em dia sobre a inclusão de portadores de necessidades especiais na sociedade e da garantia aos seus direitos como cidadão. A legislação brasileira abrange o direito à inclusão escolar e obriga que as escolas de ensino regular permitam a matrícula de qualquer aluno, independe da sua necessidade de cuidados especiais (INCLUSÃOJÁ, 2013). Porém, os professores de escolas públicas não recebem um treinamento para atender essas crianças e não recebem os recursos necessários para os cuidados adequados. É necessária a evolução na capacitação docente, pois a tecnologia é algo ainda a ser desmistificado para a maioria dos professores (SOUZA, 2013).

Entre as ferramentas tecnológicas disponíveis, existem dois aplicativos desenvolvidos como Trabalho de Conclusão de Curso pela Universidade Regional de Blumenau, que tem como objetivo ajudar a suprir a necessidade de ajudar os docentes a trabalhem com crianças com problemas de comunicação. O aplicativo Tagarela, que foi estendido e convertido para tecnologias web por Wippel (2015), tem como objetivo auxiliar na comunicação de pessoas com limitações fonoarticulosas e compartilhar a experiências que cada profissional teve no desenvolvimento de cada usuário. O outro aplicativo é o Jogo de Letra e Números, desenvolvido por Reetz (2013), com o objetivo principal de ensinar a escrever as letras do alfabeto e os números. Mesmo com projetos como esses, as funcionalidades necessárias para atender pessoas com necessidades especiais acabam fragmentadas em diversas aplicações e plataformas diferentes, dificultando o acesso e utilização dessas ferramentas por profissionais, professores e alunos.

Neste trabalho foi estendido o aplicativo Tagarela adicionando as funcionalidades do Jogo de Letras e Números. Com isso, acredita-se que o jogo possa ser utilizado em mais plataformas, aumentando a sua acessibilidade. Com a integração do jogo dentro do aplicativo Tagarela, é possível a utilização dos recursos que já existem nele, como o registro da evolução de cada usuário e o compartilhamento dessa evolução entre os educadores. O Tagarela

trabalha com pranchas de comunicação, sendo que uma prancha é um conjunto de até 9 símbolos. O Jogo de Letras e Números deverá utilizar a mesma estrutura de pranchas já existente para o Tagarela.

1.1 OBJETIVOS

O objetivo deste trabalho é estender o aplicativo Tagarela (WIPPEL, 2015) adicionando o Módulo de Jogo de Letras e Números.

Os objetivos específicos são:

a) migrar o Jogo de Letras/Números para Phonegap (REETZ, 2013);
b) criar miniaturas dos símbolos das pranchas do aplicativo Tagarela;
c) utilizar o *framework* Ionic para a construção da interface.

1.2 ESTRUTURA

A estrutura desde trabalho está apresentada em quatro capítulos. O primeiro é dedicado à introdução ao tema, aos objetivos e à estrutura do trabalho

O segundo capítulo é focado na fundamentação teórica necessária para o entendimento desde trabalho.

O capítulo três apresenta as etapas de desenvolvimento do trabalho, onde são apresentados os requisitos, os diagramas, a implementação, resultados e discussões.

O quarto capítulo refere-se às conclusões obtidas pelo trabalho e algumas sugestões de extensão ao mesmo.

2 FUNDAMENTAÇÃO TEÓRICA

A seção 2.1 trata de estudos a respeito do aprendizado da escrita. A seção 2.2 descreve o que é o aplicativo Tagarela atualmente. A seção 2.3 descreve o Jogo de Letras/Números (REETZ, 2013). Na seção 2.4 são descritas as tecnologias utilizadas, no caso, Cordova, Canvas HTML e Ionic. Finalmente, a seção 2.5 traz 3 trabalhos correlatos ao trabalho que foi desenvolvido.

2.1 APRENDIZAGEM DA ESCRITA

O alfabetizador deve ter consciência em adquirir conceitos relacionados à leitura e escrita, como um processo gradual, em que os estudantes irão alcançar maturidade cognitiva em relação à linguagem. Irão ocorrer alguns erros dos alunos, até que eles por si só (com o auxílio do educador) alcancem um nível de automatização da linguagem (UVA, 2007).

A leitura está estritamente relacionada à escrita, mas sua aprendizagem está tradicionalmente ligada aos atributos linguísticos, culturais, sociais e a formação do sujeito, sejam como meio de permitir ao indivíduo a aquisição do conhecimento, seja como meio de viabilizar sua atuação social. Em face disso, surge a necessidade de se discutir sobre o processo de leitura e escrita nos anos iniciais do Ensino Fundamental dada a sua relevância para o processo ensino-aprendizagem (MARLUCE, 2013).

Durante os jogos e brincadeiras, as crianças adquirem diversas experiências, interagem com outras pessoas, organizam seu pensamento, tomam decisões, desenvolvem o pensamento abstrato e criam maneiras diversificadas de jogar, brincar e produzir conhecimentos. Nesse sentido, os jogos e as brincadeiras são instrumentos pedagógicos importantes e determinantes para o desenvolvimento da criança, pois no jogar e no brincar as mesmas desenvolvem habilidades necessárias para o seu processo de alfabetização e letramento (VIEIRA, 2010).

O jogo é importante e necessário para o desenvolvimento intelectual e social da criança, estimulando sua criticidade, criatividade e habilidades sociais. Portanto, ao utilizar-se de atividades lúdicas o professor propicia ao aluno a oportunidade de interagir por meio da Língua Portuguesa de forma dinâmica, interpretando texto, expondo ideias ou mesmo extrapolando seus conhecimentos para outras áreas (SOARES, 2013).

Ao longo da história da Educação, as escolas trataram as crianças com deficiência como incapazes, necessitando de tratamento médico, não de ensino (GESTAOESCOLAR, 2016). Essa perspectiva começou a mudar a partir de 1948, com a Declaração Universal de Direitos Humanos, que garantiu o direito de todos à Educação. Demorou algumas décadas para, a partir dos anos 1990, a visão assistencialista ser deixada de lado e dar lugar ao

conceito de inclusão, que ganhou um papel central em documentos internacionais (GESTAOESCOLAR, 2016).

Mediante as situações oferecidas em sala de aula, para que o conteúdo se torne mais claro e acessível há a necessidade de alguns ajustes para auxiliar na compreensão dos assuntos, favorecendo a formação da imagem mental tão necessária para alunos com deficiência intelectual (REAB, 2015). Assim o uso da comunicação alternativa torna-se excelente ferramenta para viabilizar este acesso e compreensão das atividades, bem como das preferências e escolhas por determinados temas ou personagens (REAB, 2015).

2.2 TAGARELA – REDE DE COMUNICAÇÃO ALTERNATIVA

O Tagarela é um aplicativo para comunicação alternativa originalmente desenvolvimento por Fabeni (2012) para a plataforma iOS. O aplicativo tem como objetivo criar um ambiente aonde o fonoaudiólogo, o seu aluno e o tutor deste aluno possam interagir de forma a proporcionar evolução na capacidade de comunicação do aluno.

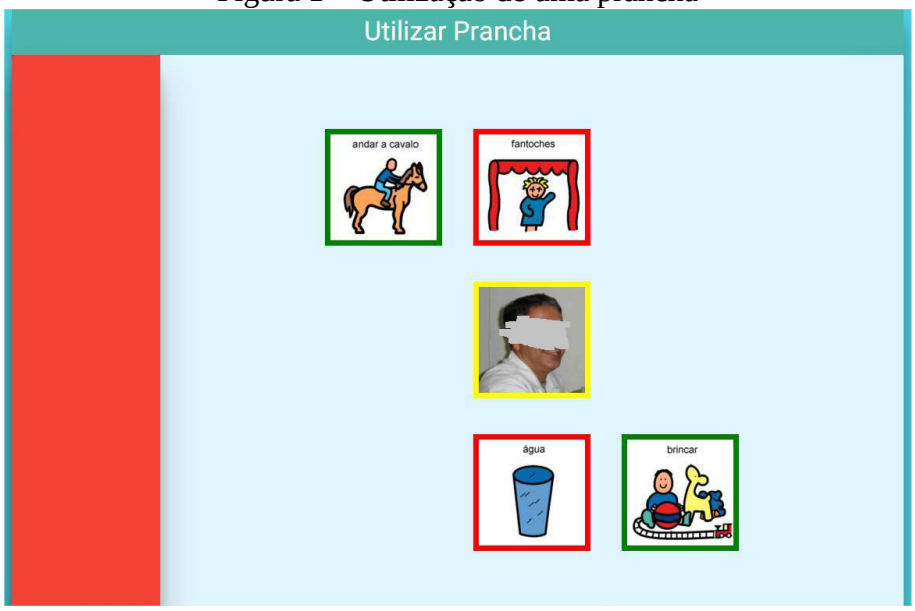
Após isso, Marco (2014) desenvolveu uma versão do Tagarela para o Android, aumentando o alcance do aplicativo. Nesta nova versão o aplicativo passou a utilizar pranchas de comunicação, as quais possuem símbolos gráficos. Esses símbolos são definidos de acordo com as necessidades de cada indivíduo e representam alguma ação ou objeto do mundo real (MARCO, 2014).

Um ano depois, Wippel (2015) reescreveu o aplicativo com o objetivo de torna-lo multiplataforma. A nova versão da aplicação foi implementada através da ferramenta PhoneGap, que permite reutilizar tecnologias de desenvolvimento web existentes para desenvolver rapidamente aplicações híbridas construídas com HyperText Markup Language (HTML), Cascading Style Sheets (CSS) e JavaScript, e estender a aplicação para várias plataformas com uma única base de código (PHONEGAP, 2016).

A principal funcionalidade do aplicativo Tagarela é a interação com as pranchas de comunicação. Ao criar uma prancha é mostrado ao usuário uma prancha 3x3 vazia. Ele então deve selecionar dentro de uma categoria, quais símbolos farão parte desta nova prancha. A seleção do símbolo é realizada a partir de quatro categorias específicas, verbos, substantivos, descritivos e pessoas (WIPPEL, 2015). Para visualizar uma prancha, primeiramente, deve-se selecionar um plano de atividades, objeto que agrupa pranchas de comunicação, para então selecionar a prancha desejada. A partir daí será apresentada uma tela para o usuário, com os símbolos da prancha 3x3, conforme a Figura 1. Ao pressionar um símbolo, o áudio vinculado a ele será reproduzido. Por não ser necessário o preenchimento de todas as 9 posições da

prancha, as posições que não possuírem símbolos ficarão em branco e não reproduzirão nenhum áudio ao serem pressionadas (WIPPEL, 2015).

Figura 1 – Utilização de uma prancha



Fonte: Wippel (2015).

O aplicativo permite criar diferentes perfis de usuários, como instrutor e aluno. As pranchas criadas por um perfil podem ser compartilhadas com outros perfis (preservando o anonimato do aluno), assim como as anotações feitas sobre usuários da ferramenta.

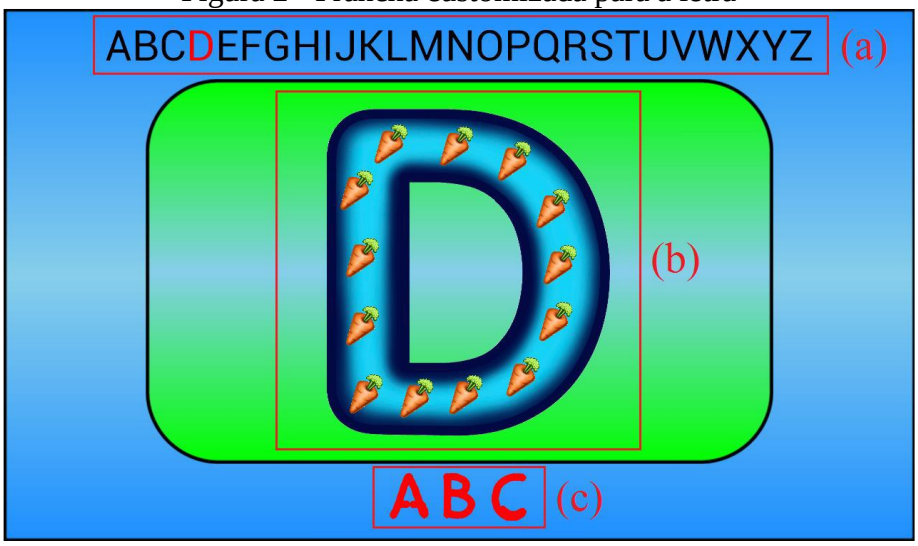
2.3 JOGO DE LETRAS E NÚMEROS VOLTADO PARA A TECNOLOGIA ASSISTIVA NO ANDROID

Reetz (2013) desenvolveu um protótipo de jogo para plataforma Android voltado para tecnologia assistiva, que explora o aspecto pedagógico/lúdico em computação aplicada, tendo como cenário um jogo 2D que manipula letras e números. O jogo foi desenvolvido com a utilização do *kit* de desenvolvimento de software Android e a linguagem de programação Java, limitando a sua utilização para dispositivos que executam o sistema operacional Android.

O jogo permite criar um plano (no caso, planos são conjuntos de pranchas no aplicativo Tagarela) customizado informando o nome do plano e um texto customizado para ser utilizado durante o jogo. A principal funcionalidade do jogo é a interação com os símbolos e os desenhos deles. Na Figura 2 é possível verificar a exibição de um plano onde no item (a) são mostradas todas as letras do plano, no item (b) é exibida a letra/número atual e no item (c) o progresso já realizado, nesse caso, os símbolos que já foram preenchidos. Na Figura 3 é

demonstrada a interação com o símbolo, onde ele é desenhado através do toque na tela e passa para o próximo plano após ser completado, salvando o resultado do plano anterior.

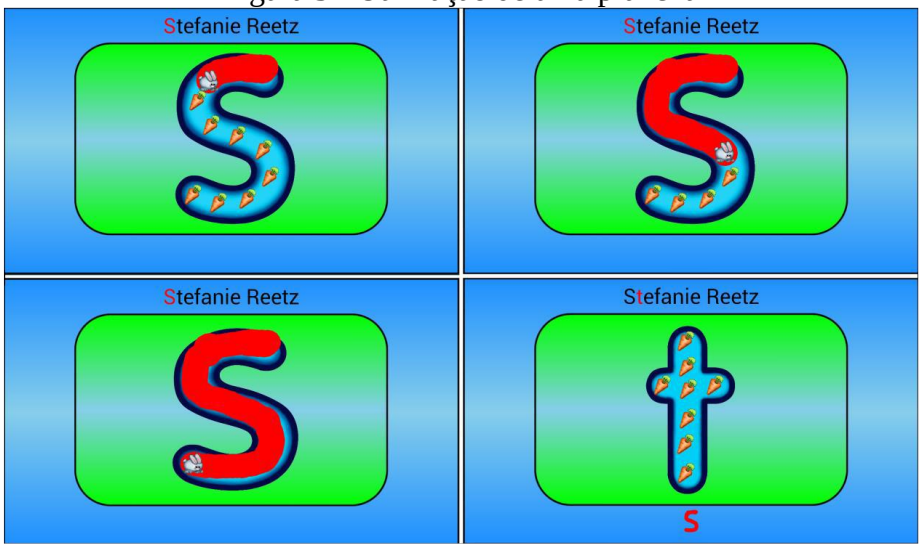
Figura 2 - Prancha customizada para a letra



Fonte: Wippel (2015).

Na Figura 3 é demonstrada a utilização de uma prancha com a letra s preenchida com desenhos de cenouras, nesse caso as presas. Ao tocar na letra, a criança consegue movimentar o desenho de um coelho, o predador, o qual deve ser arrastado pelo interior da letra para coletar todas as presas. O jogo possui 5 tipos de presas (Abelha, Coelho, Gato, Galinha e Sapo) e 5 tipos de predadores (Flor, Cenoura, Peixe, Minhoca e Mosca). Cada símbolo utiliza uma presa e um predador pré-definido.

Figura 3 - Utilização de uma prancha



Fonte: Wippel (2015).

2.4 TRABALHOS CORRELATOS

Nesta seção são apresentados três trabalhos utilizados para auxiliar na alfabetização de crianças. São comentadas as principais características dos trabalhos Participar, Livox e Scala.

2.4.1 Participar

É um software educacional de apoio a alfabetização e comunicação alternativa de jovens e adultos com deficiência intelectual. É um projeto no campo de alfabetização social, inclusão digital e cidadania, já que o foco é especializado para somente esse público-alvo. Seu objetivo é servir como ferramenta de apoio a professores atuantes no processo de alfabetização de jovens e adultos com deficiência intelectual. A meta final é que o educando passe a ser capaz de comunicar-se por meio de computadores (comunicação alternativa), bem como tenha maior inserção social (PARTICIPAR, 2014).

O software foi desenvolvido em 2011 pela Universidade de Brasília e foi testado em 2012 com casos reais, com professores especializados em alfabetização de adultos com deficiência e é utilizado em escolas públicas do Distrito Federal e em Associações de Pais e Amigos de Excepcionais (APAE) de todas as Unidades da Federação. Na Figura 4 pode ser vista a interface da aplicação, onde são exibidas palavras com a letra `Ç` e dois `players`, um para a leitura labial associada a letra e outro para a exibição da letra em Libras.

Figura 4 - Software Participar

Fonte: Participar (2014).

2.4.2 LIVOX

O Livox surgiu da necessidade de um pai (Carlos Edmar Pereira) e de uma mãe (Aline Costa Pereira) de se comunicarem melhor com sua filha (Clara Costa Pereira) que tem Paralisia Cerebral. Carlos Pereira tem formação em Análise de Sistemas e reuniu uma equipe

de colaboradores formada por profissionais de tecnologia, fonoaudiólogos e terapeutas ocupacionais para criação daquele que viria a ser o primeiro software de comunicação alternativa para tablets em Português: o Livox (LIVOX, 2015).

A Figura 5 apresenta a interface do Livox. A comunicação acontece a partir de toques nas imagens apresentadas na tela. A tela apresenta quadrinhos com informações sobre necessidades, emoções, o que comer, brincar ou até mesmo iniciar a exibição de vídeos ou filmes. De acordo com Carlos, "O Livox já vem com 12 mil imagens e ainda permite a inclusão de mais. Isso é possível através da câmera do próprio tablet ou da inserção de quaisquer outros itens" (BENGALALEGAL, 2012).

Figura 5 - Software Livox

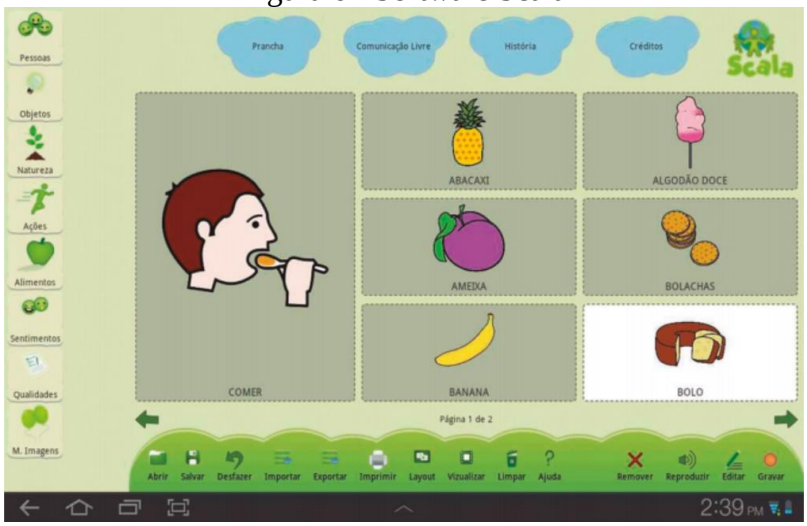
Fonte: Bengalalegal (2012).

2.4.3 SCALA

Scala é mais um software desenvolvido em ambiente acadêmico, na Universidade Federal do Rio Grande do Sul, sobre licença General Public License (GNU) e Creative Commons, garantindo um conteúdo aberto e livre, e a continuidade do projeto (PASSERINO, 2015). O programa tem como público alvo crianças com autismo, um transtorno de desenvolvimento que geralmente aparece nos três primeiros anos de vida e compromete as habilidades de comunicação e interação social (MINHAVIDA, 2016).

Existem duas versões do software disponíveis, uma para tablets com sistema operacional Android e outra versão Web. O funcionamento nas duas plataformas é igual. Na Figura 6 é apresentada a interface da aplicação onde a criança escolheu a opção COMER e pode escolher qual alimento ela deseja.

Figura 6 - Software Scala



Fonte: Passerino (2015).

.

3 DESENVOLVIMENTO

Neste capítulo são apresentadas as etapas de desenvolvimento do aplicativo. Na primeira seção estão descritos os requisitos funcionais e não–funcionais do trabalho. A segunda seção contém as especificações do aplicativo. Na terceira seção é detalhada a implementação do aplicativo, demonstrando trechos de códigos e imagens exemplificando a utilização da ferramenta. E, por último, a quarta seção apresenta os resultados obtidos com esse trabalho.

3.1 REQUISITOS

O trabalho proposta deverá:

a) ser implementado utilizando tecnologia web, como HTML, PHP e Javascript (Requisito Não Funcional - RNF);
b) ser implementado utilizando o ambiente PhoneGap (RNF);
c) apresentar uma interface que utilize o conceito de pranchas, seguindo o padrão já adotado pelo aplicativo Tagarela (RNF);
d) utilizar o *framework* Ionic (RNF);
e) ser um módulo integrado do Tagarela (RNF);
f) utilizar os planos presentes no Tagarela (RNF);
g) manter um histórico de cada aluno (Requisito Funcional - RF);
h) permitir trabalhar com todas as letras e números da língua portuguesa (RF).

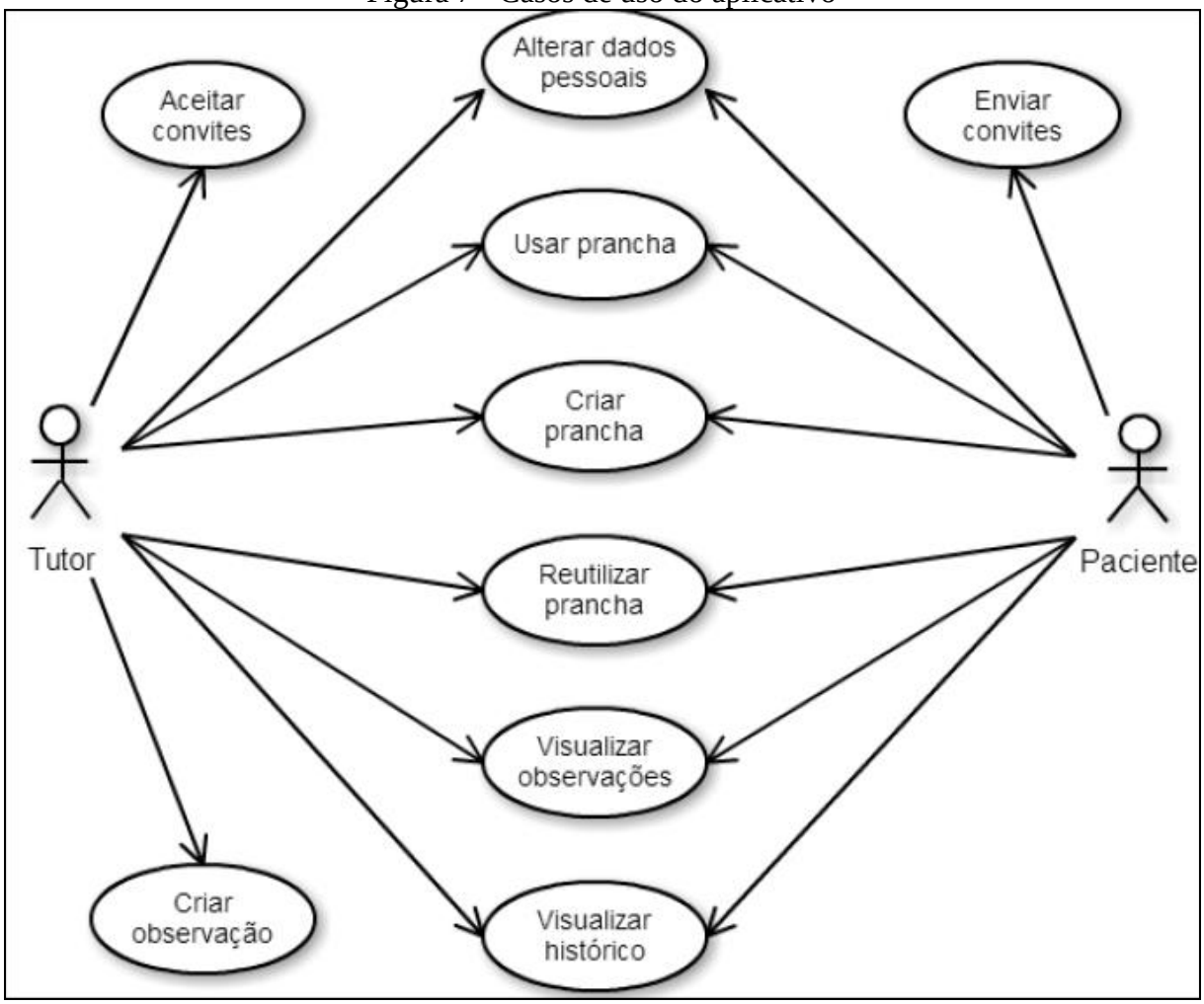
3.2 ESPECIFICAÇÃO

A especificação deste trabalho foi desenvolvida utilizando modelagem de diagrama de casos de uso, diagrama de atividades, Modelos de Entidade e Relacionamento (MER) e um diagrama de classes utilizando o padrão UML. A ferramenta utilizada foi o Enterprise Architect.

3.2.1 Casos de uso

A aplicação possui dois tipos de usuários, `TUTOR` e `PACIENTE`. Nessa seção são apresentados os casos de usos das funcionalidades dos mesmos. O `TUTOR` é o responsável pelo gerenciamento das pranchas e planos que serão utilizados pelos alunos. Já o usuário do tipo `PACIENTE` pode interagir com esses planos e pranchas. Desta forma, é possível definir o caso de uso da Figura 7.

Figura 7 - Casos de uso do aplicativo



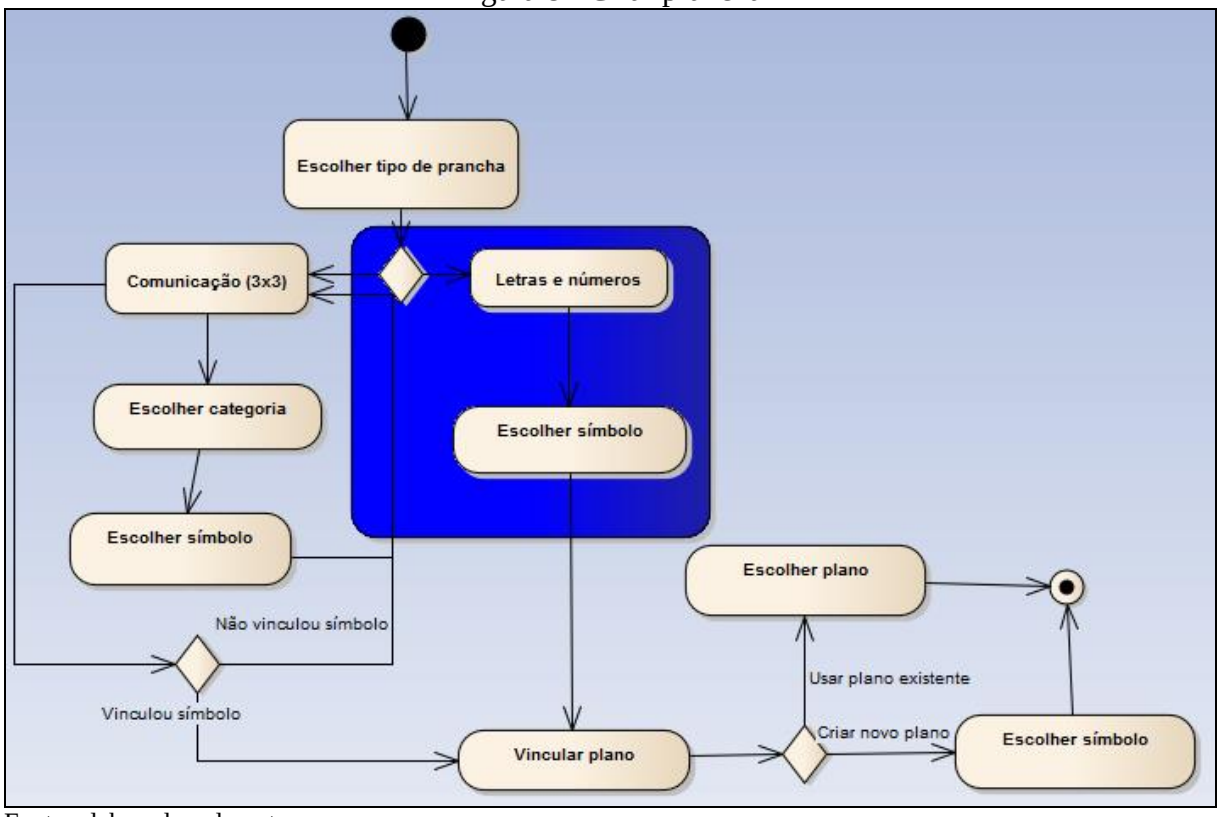
Fonte: Wippel (2015).

Os casos de uso pertencem ao aplicativo original e se mantiveram exatamente os mesmos. Eles foram desenvolvidos com o padrão UML de especificação. O caso de uso exclusivo do PACIENTE é o envio de um convite para outro usuário ser o seu TUTOR. Os casos de usos exclusivos do TUTOR são poder aceitar esses convites e criar observações sobre os PACIENTES que ele é tutor, para que possa ser utilizado por outro tutor em caso de troca. Os casos de uso compartilhados são a alteração dos dados pessoais do seu próprio usuário, criar uma nova prancha, utilizar uma prancha criada e reutilizar uma prancha criada, que é a possibilidade de um TUTOR criar uma nova prancha com base em uma já criada por outro TUTOR. Também é possível visualizar as observações criadas para um PACIENTE e visualizar o histórico, que é a visualização de todas as pranchas utilizadas por um PACIENTE, com data e hora.

3.2.2 Diagrama de atividades

O diagrama de atividades abaixo mostra a utilização dos recursos do Tagarela. Como algumas atividades se mantiveram da mesma forma do TCC original, será apresentada somente a atividade que teve alteração em relação a sua versão de origem. A Figura 8 demonstra o diagrama de atividades da criação de uma prancha na aplicação, com destaque em fundo azul dos novos passos da atividade.

Figura 8 - Criar prancha



Fonte: elaborado pelo autor.

Ao criar uma prancha, o primeiro passo do usuário é escolher entre uma prancha de comunicação ou uma prancha do Jogo de Letras e Números. Ambas seguem caminhos distintos, pois a prancha de comunicação irá criar uma prancha 3x3, para escolha de uma categoria e de um símbolo. Ao escolher esse tipo de prancha o usuário deve escolher os símbolos que irão compor a prancha. Para cada símbolo o usuário deve escolher uma das categorias, sendo elas pessoas (amarelo), verbos (verde), substantivos (vermelho) e descritivos (azuis). O usuário também possui a opção de reutilizar uma prancha existente, que irá carregar uma nova prancha com todos os símbolos, exceto os da categoria pessoas.

Caso o usuário escolha criar uma prancha do Jogo de Letras e Números ele será direcionado diretamente para a escolha do símbolo, composto por letras de A a Z (sem

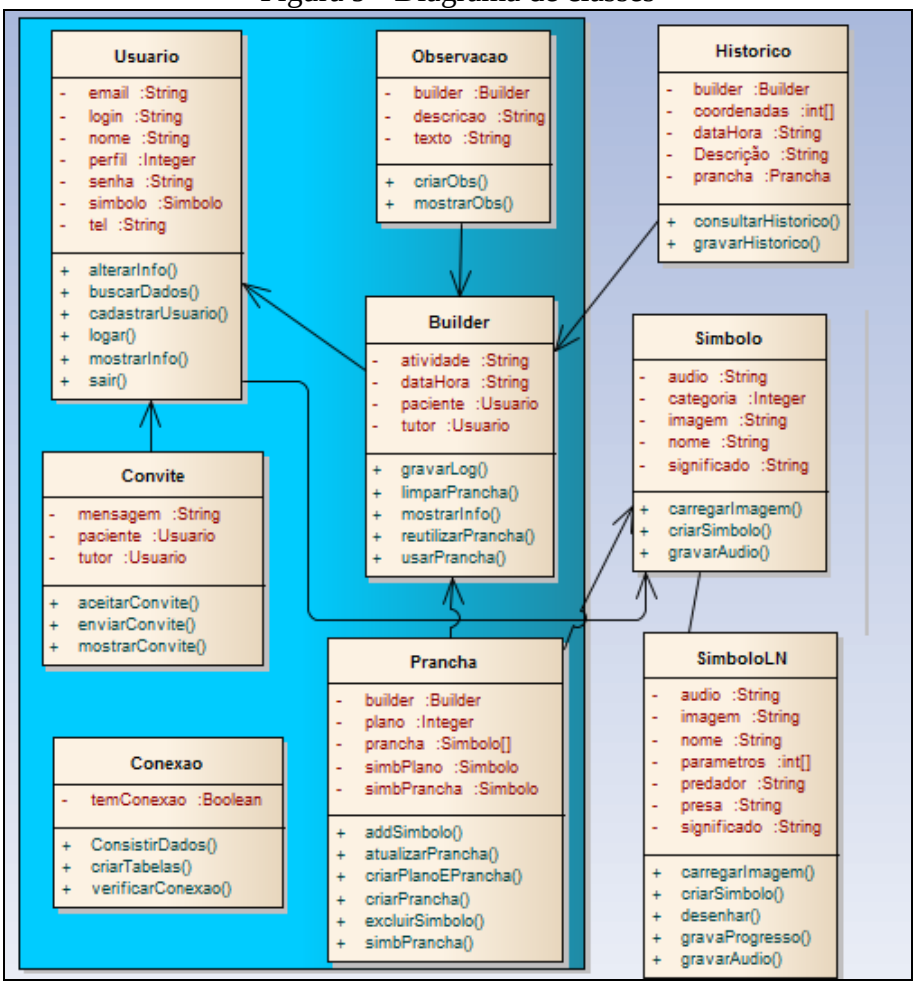
acentuação) e números de 0 a 9, não sendo necessário escolher previamente uma categoria. Ao contrário da prancha de comunicação, esta prancha é composta por somente um símbolo.

Após a escolha e criação de uma das suas categorias de prancha, o usuário é direcionado para uma página para a escolha do plano da prancha, onde ele poderá optar por utilizar uma prancha já existente ou criar uma nova.

3.2.3 Diagrama de classes

Não é possível a criação de classes através do Javascript, pois ele não permite a criação das mesmas. Devido a isso, o diagrama de classes abaixo é uma simulação dos objetos envolvidos na aplicação e suas funcionalidades. Esse diagrama manteve a mesma estrutura do aplicativo original, com a adição de novas propriedades e classes, conforme a Figura 9. As classes destacas dentro do quadrado azul são classes que se mantiveram sem alterações, as classes `Historico` e `Simbolo` tiveram as suas estruturas alteradas e a classe `SimboloLN` é nova.

Figura 9 - Diagrama de classes



Fonte: elaborado pelo autor.

Comparado com a diagrama de classes do aplicativo original, as alterações mais significativas foram a criação das classes `SimboloLN` e `Historico`, além de alterações nos métodos da classe `Prancha`.

A classe `Prancha` agora irá permitir a criação de pranchas para o Jogo de Letras e Números, armazenando o símbolo do jogo como o símbolo principal da prancha, não precisando mais fazer uso do vetor de símbolos.

A classe `Historico` possui um `array` de `int` destinado ao armazenamento das coordenadas utilizadas pelo usuário durante o jogo através do método `gravarHistorico`, para que possa ser exibido para o tutor ao visualizar o histórico através do método `consultarHistorico`. Esses dados ficam armazenados para poder visualizar como o usuário preencheu o símbolo ao consulta o histórico de utilização das pranchas do aluno.

A classe `SimboloLN` simboliza os dados que são gerados ao exibir uma prancha do Jogo de Letras e Números. O método `carregarSimbolo` faz a geração da imagem, do predador e das presas. Após o carregamento o método `desenhar` é o responsável por pintar o trajeto desenhardo pelo usuário no Canvas. O método `gravarProgresso` é acionado ao finalizar o desenho do símbolo, onde o método prepara os dados a serem enviados para a classe `Historico` efetuar o armazenamento do mesmo.

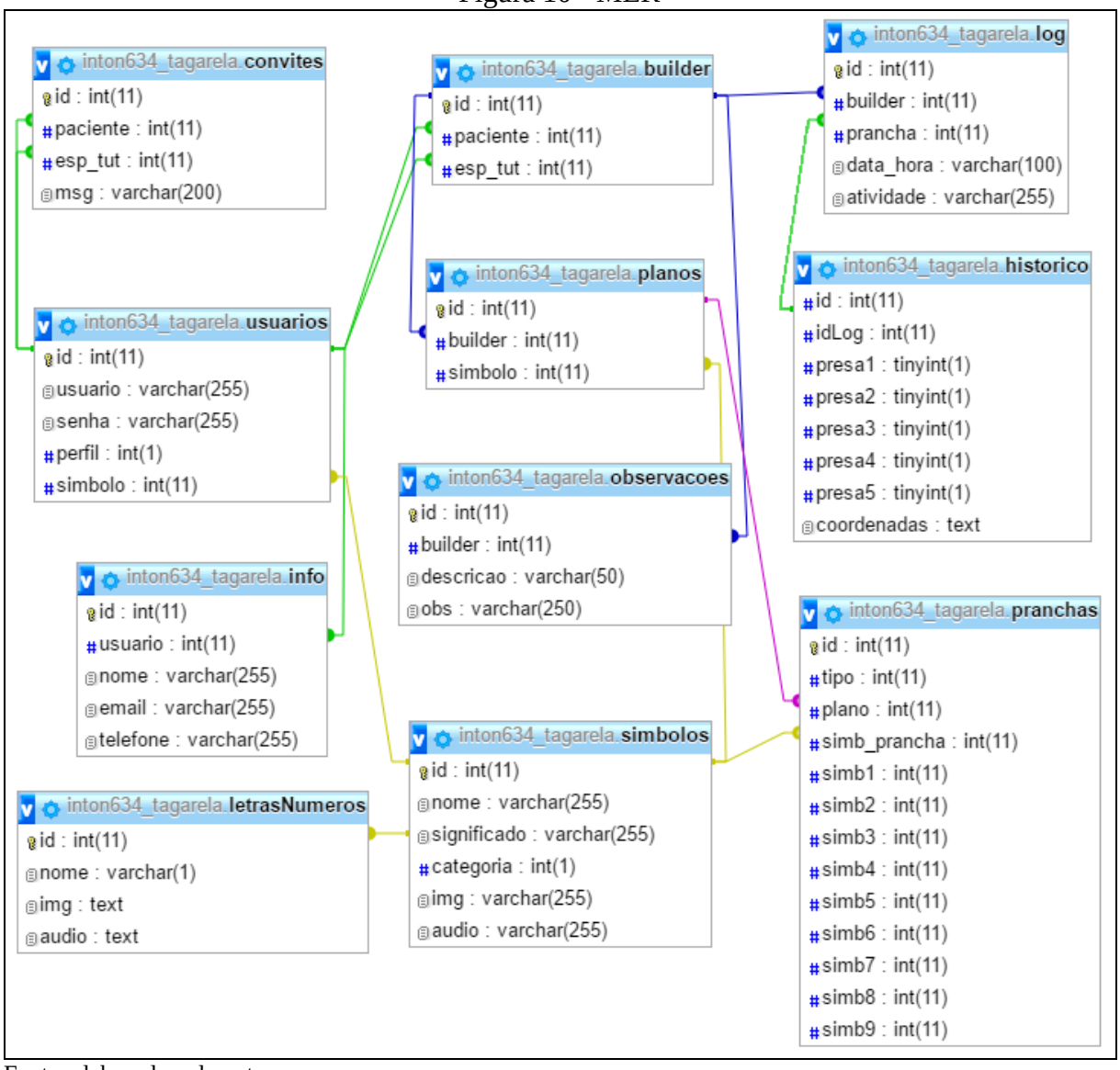
3.2.4 Diagrama MER

A Figura 10 é a representação do MER do Banco de Dados. Essa estrutura é mantida no servidor remoto e na base local, para a persistência dos dispositivos móveis. O banco de dados dessa aplicação é composto por 11 tabelas, conforme pode ser verificado na Figura 10. Os dados referentes aos usuários que utilizam a aplicação são armazenados nas tabelas `usuarios` e `info`.

A tabela `convites` armazena todos os convites enviados através da aplicação e a tabela `observacoes` todas as observações criadas para um `usuario`. As tabelas `log` e `historico` são utilizadas para armazenar o histórico de utilização das pranchas. Enquanto o `log` é responsável por armazenar qual usuário utilizou qual prancha, na tabela `historico` são armazenados dados específicos para as pranchas de letras e números, como quais presas foram capturados e as coordenadas utilizadas para isso. As tabelas `planos`, `simbolos` e `pranchas` armazenam respectivamente os planos, os símbolos e as pranchas. A tabela `letrasNumeros` armazena os simbolos para o Jogo de Letras e Números e a tabela `builder` armazena a ligação entre um tutor e um aluno.

A estrutura base do banco de dados se manteve a mesma do aplicativo original, com a inclusão de novas tabelas e campos necessários para o funcionamento do jogo. Assim como no original, as tabelas que são criadas localmente nos dispositivos móveis possuem uma coluna a mais, com o nome de `sync`, e é utilizada durante o processo de sincronização dos dados locais com o banco remoto. A representação do banco de dados não segue a mesma estrutura do diagrama de classes da Figura 9 pois o programa não possui classes reais, somente uma representação para o seu funcionamento.

Figura 10 - MER



Fonte: elaborado pelo autor.

3.3 IMPLEMENTAÇÃO

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

3.3.1 Técnicas e ferramentas utilizadas

Para o desenvolvimento da aplicação foi utilizado a ferramenta Phonegap em conjunto com o Ionic Framework. A utilização do PhoneGap foi devido a capacidade de portar um único código para mais de uma plataforma, permitindo a utilização do aplicativo na versão *web* e móvel. O Ionic foi utilizado devido aos recursos utilizados e para recriar o visual do aplicativo. Para isso foi criado um novo projeto com Phonegap e Ionic e reescrito o código do Tagarela original.

Para o desenvolvimento foi utilizada a ferramenta Brackets 1.7 e Cordova, com as linguagens HTML 5, Canvas, CSS 3 e Javascript para o aplicativo, e PHP 5.4 e MySQL para o servidor. Foi utilizada a biblioteca Javascript jQuery 1.7.2 3 jQuery Mobile 1.1.2. Para a emulação local do servidor foi utilizado o programa Easy PHP 14.1. Nos testes foram utilizados os navegadores Google Chrome 54.0 e Microsoft EDGE 34, e para a versão móvel foi utilizado o aparelho ASUS ZenPhone 2 modelo ASUS_Z00AD com Android versão 5.0.

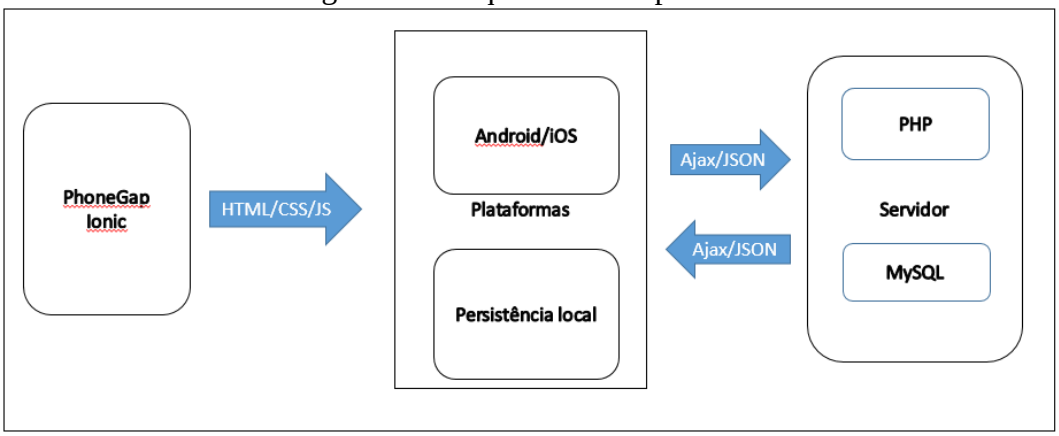
3.3.1.1 Canvas HTML

O principal elemento utilizado para o jogo é o Canvas. Canvas é um elemento HTML que pode ser utilizado para fazer desenhos na linguagem Javascript, permitindo criar gráficos, fotos ou animações. Ele possui métodos prontos para criação de linhas, caixas, círculos, textos e adicionar imagens. Ele foi primeiramente introduzido pela Apple no Mac OS X e mais tarde implementado nos browsers Sarafi e Google Chrome (MOZILLA, 2016).

3.3.1.2 Arquitetura

A arquitetura utilizada na aplicação seguiu a mesma ideia do aplicativo original, cliente/servidor. A alteração da estrutura original foi a introdução do Ionic na parte cliente e a alteração do tipo do servidor, que antes utilizava o OpenShift, um servidor limitado disponibilizado pela Oracle, e agora está em um servidor próprio. A estrutura no Tagarela é demonstrada na Figura 11.

Figura 11 - Arquitetura do aplicativo



Fonte: elaborado pelo autor.

O *deploy* da aplicação cliente é feito pelo Phonegap, que cria a aplicação para os dispositivos Androids e iOS com base no código HTML, CSS e Javascript. O aplicativo gerado verifica quando existe conexão com a internet e sincroniza os dados locais com o servidor, sendo que os dados locais são armazenados utilizando WebSQL. O acesso da aplicação web pelo navegador não necessita dessa sincronização pois faz o acesso diretamente no banco de dados do servidor. Foi optado por utilizar um servidor privado para ser possível o acesso a todos os recursos necessários. No servidor os dados são armazenados em um base de dados MySQL e o acesso a esses dados pelo aplicativo é feito via javascript utilizando a tecnologia Asynchronous Javascript and XML (AJAX) para acessar os scripts PHP, que fazem a consulta a esses dados no banco.

3.3.1.3 Utilização do Phonegap/Cordova

Cordova é uma ferramenta que envolve uma aplicação desenvolvida em HTML e Javascript em um *container* que pode acessar as funções nativas dos dispositivos. Essas funções são disponibilizadas através de funções Javascript, permitindo escrever uma aplicação que atinja quase todos os smartphones e tablets do mercado, bem como publicar em suas lojas (CORDOVA, 2015). Além das funcionalidades básicas, o Cordova possui uma biblioteca de *plugins*. *Plugins* são funções adicionais ao Cordova, fornecendo interface Javascript para componentes nativos. Eles permitem que os aplicativos possam usar as capacidades nativas dos aparelhos, que normalmente não estão disponíveis em aplicações web tradicionais (CORDOVA, 2015).

Foi optado por continuar utilizando a tecnologia Phonegap para o desenvolvimento do aplicativo, pelo motivo de ser uma ferramenta de desenvolvimento multiplataforma e do seu código poder ser reutilizado para acesso *web*. Devido ao Cordova não suportar a adição do Ionic a projetos já existentes, foi necessário a criação de um novo aplicativo com a adição das

plataformas desejadas. Para cada plataforma adicionada ao projeto é necessário obter o Software Development Kit (SDK) da mesma, para que o programa possa gerar o aplicativo em código nativo.

O primeiro passo foi a instalação do NodeJS, que é uma plataforma de desenvolvimento Web que utiliza somente código Javascript. Ela é necessária para a instalação do Phonegap e do Ionic, que é feita executando o comando `npm install -g cordova ionic` no terminal do NodeJS. Após a instalação do Phonegap o console do NodeJS passará a aceitar as linhas de comandos do Cordova. Para a criação do aplicativo, é necessária a execução das linhas do comando do Quadro 1.

Quadro 1 - Criação de um aplicativo do cordova

```
1  ionic start Tagarela blank
2
3  ionic platform android
4
5  cordova plugin add cordova-plugin-camera
6  cordova plugin add cordova-plugin-file-transfer
7  cordova plugin add cordova-plugin-media-capture
8  cordova plugin add cordova-plugin-media
```

Fonte: elaborado pelo autor.

O comando padrão para a criação de um aplicativo pelo Phonegap é o `phonegap create`, mas para que o aplicativo possa utilizar os recursos do Ionic o aplicativo precisa ser criado com o comando `ionic start` seguido pelo nome do aplicativo e em sequência qual modelo de aplicativo ele irá utilizar. Como o Tagarela não utiliza nenhum estilo padrão, foi criado um aplicativo em branco. Nesse momento são criados todos os arquivos necessários para iniciar o desenvolvimento do aplicativo, porém ele só poderá ser executado em dispositivos que foram adicionados. Para isso é utilizado o comando `ionic plataform`, seguido do nome da plataforma que se deseja adicionar, sendo que as plataformas aceitas são `android` e `ios`.

Os comandos utilizados entre as linhas 5 e 8 são opcionais, pois eles servem para adicionar *plugins* que permitem o acesso a ferramentas dos dispositivos em que serão executados. O comando a ser executado é `cordova plugin add` seguido pelo nome do *plugin*. Para esse projeto foram adicionados os *plugins* `camera`, responsável por permitir o acesso a câmera do dispositivo, `file-transfer` que permitem acessar e salvar fotos no dispositivo, e `media` e `media-capture` que permite gravar e executar áudios no dispositivo.

Para a simulação do aplicativo pode ser executado o comando `ionic run` seguido da plataforma desejada. Nesse caso o aplicativo será compilado, enviado e executado no dispositivo que estiver conectado e habilitado no computador. Se não for possível executar

diretamente em um dispositivo, é possível utilizar um emulador e executar com o comando `ionic emulate`.

3.3.1.4 Utilização do Ionic

Para facilitar o desenvolvimento utilizando HTML multiplataforma surgiu o Ionic. O *framework* Ionic é gratuito e de código aberto, criado em 2013 e oferece bibliotecas HTML, CSS e Javascript otimizadas para dispositivos móveis, gestos e criação de dispositivos interativos (IONIC, 2015). A vantagem em utilizar o Ionic é a grande quantidade de componentes que ele possui, com os recursos mais novos de CSS, HTML e Javascript, com a promessa de grande desempenho e a compatibilidade com Android 4.1, iOS 7 e BlackBerry 10. Esses recursos têm a intensão de facilitar o desenvolvimento dos aplicativos móveis, entregando vários recursos prontos, para que o desenvolvedor possa focar no objetivo da aplicação e posso abstrair os detalhes de implementação de cada plataforma. Esta ferramenta é gratuita e pode ser utilizada por qualquer desenvolvedor. O *site* do Ionic possui uma área para a compra e venda de *plugins*, que em sua maioria são bibliotecas visando facilitar a criação de funcionalidades mais comuns (IONIC, 2015).

Para a utilização do Ionic é necessário chamar via Javascript passando como referência o nome do *controller*. Conforme o Quadro 2 o nome do *controller* utilizado é `MyCtrl` e os *scripts* desejados para essa página devem ser adicionados dentro dessa função, na linha 25. Para que funcione, a *tag* `body` da página deve conter o atributo `nr-controller="MyCtrl"`.

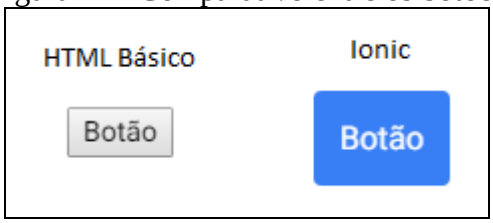
Quadro 2 - Chamado do Ionic

```
23      <script>
24      angular.module('ionicApp', ['ionic']).controller('MyCtrl', function($scope) {
25
26      });
27      </script>
```

Fonte: elaborado pelo autor.

A principal utilização do Ionic no projeto foi para refazer o visual das atividades, redesenhando suas páginas para serem responsivas e de fácil manutenção. O Ionic possui uma serie de classes CSS que podem ser utilizadas para criar mais rapidamente os objetos visuais da página. A Figura 12 mostra a diferença entre um botão gerado com HTML puro, utilizando o *tag* `<button>Botão</button>`, e de um botão gerado da mesma maneira mas utilizando uma classe do Ionic, com o comando `<button class="button button-positive">Botão</button>`.

Figura 12 - Comparativo entre os botões



Fonte: elaborado pelo autor.

3.3.1.5 Persistência dos dados

Os dados do aplicativo são armazenados de forma local nos dispositivos utilizando WebSQL e devido a isso se faz necessária a sincronização desses dados com o servidor do Tagarela. Essa sincronização é feita utilizando Asynchronous Javascript and XML (AJAX) para que os dados possam ser sincronizados sem interferir no funcionamento da aplicação. Quando o acesso é feito pelo navegador, a consulta é feita diretamente no servidor, como no Quadro 3 que demostra a consulta de uma prancha do Jogo de Letras e Números. Das linhas 30 a 37 são preparados os dados que serão enviados ao servidor, em formato JavaScript Object Notation (JSON). Na linha 40 está o endereço do servidor onde está o script PHP que irá carregar os dados da prancha. Entre as linhas 43 e 53 estão as ações a serem tomadas caso a consulta ao banco ocorra com sucesso, efetuando o carregamento da imagem da prancha na página. Linhas 54 a 60 fazem o tratamento do erro na consulta, a preparação da consulta, exibindo a imagem de carregamento da página durante a consulta. Entre as linhas 61 e 69 estão as ações a serem tomadas após a consulta, que nesse caso é a remoção da imagem de carregamento e adição de funções aos elementos da página.

Devido ao uso de um banco local nos aplicativos e a possibilidade de ser acessado *offline*, antes de efetuar cada transação, o aplicativo deve verificar se possui conexão com a internet, tentando fazer um acesso no *script* de verificação do servidor, conforme a linha 10 do Quadro 4. Se conseguir efetuar a conexão é gravado 1 na variável global `conectado` (linhas 15), indicando que a busca dos dados serão feitas diretamente no servidor e não no banco local, após isso é executado o metodo `consistirDados`, para efetuar a sincronização. Se não for possível efetuar a conexão, o acesso será feito pelo banco local WebSQL.

Quadro 3 - Consistência dos dados

```
    var dados = {
        "idBuilder" : paciente,
        "espTut" : tutor,
        "dataHora" : dataHora,
        "atv" : atv,
        "idPrancha" : localStorage.idPrancha,
        "gravarLog" : localStorage.gravarLog
    };
    $.ajax({
        type     : "post",
        url      : "http://tagarela.intonses.com.br/scripts/usar-pranchaLN.php",
        data     : dados,
        dataType : "json",
        success  : function gravarLog(ret) {
            $("body").removeClass("loading");
            if (ret.erro) {
                alert(ret.msg);
            }  else {
                var conteudo = "<img src='img/"+ret.simb_prancha+"' id='minhaImagem' style='width:500px;height:500px;'>";
                var conteudo2 = "<img hidden src='img/"+ret.simb_prancha+"' id='minhaImagemAux' style='width:1000px;height:1000px;'>";
                $("#imagemLN").append(conteudo);
                $("#divAuxiliar").append(conteudo2);
            }
        },
        error    : function(ret) {
            $("body").removeClass("loading");
            alert("Erro no servidor (TIMEOUT)!");
        },
        beforeSend: function() {
            $("body").addClass("loading");
        },
        complete: function() {
            $("body").removeClass("loading");

            $(".img-simbolo").click(function() {
                var src = "audio/"+$(this).attr("title");
                audioElement.setAttribute("src",src);
                audioElement.play();
            });
        },
        timeout: 5000
    });
```

Fonte: elaborado pelo autor.

Quadro 4 - Script para verificar conexão

```
    // Verifica conexão com o servidor
    localStorage.conectado = 0;
    var dados = {
        "dadoEnv" : 1
    };
    $.ajax({
        type     : "post",
        url      : "http://tagarela.intonses.com.br/scripts/verifica-conexao.php",
        data     : dados,
        dataType : "json",
        success  : function(ret) {
            if (ret.dadoRet == 1) {
                localStorage.conectado = 1;
                consistirDados();
            }
        },
        error    : function(ret) {
            $("body").removeClass("loading");
        },
        beforeSend: function() {
            $("body").addClass("loading");
        },
        complete: function() {
            $("body").removeClass("loading");
        }
    });
```

Fonte: elaborado pelo autor.

3.3.1.6 Criação das miniaturas

Todas as pranchas possuem uma miniatura que é apresentada dentro do plano e que era escolhida pelo usuário durante a criação, porém nem sempre é possível indicar uma miniatura que represente corretamente a prancha. Foi criado um *script* em PHP que faz a criação de uma miniatura para as pranchas (Quadro 5).

Conforme o Quadro 5, é criado o *array* `$imagens` com todos os símbolos da pranchas, e setadas as variáveis como o tamanho das imagens, quantidade de linhas e colunas. É criada uma imagem com o comando `imagecreatetruecolor()` na variável `$background` setando o tamanho do canvas que será desenhado, de acordo com a quantidade e tamanho dos símbolos. Entre as linhas 88 e 93, é criado um novo *array* com o nome `$imagemsSimbolos`. É percorrido o *array* de símbolos da prancha e para cada símbolo é criada uma nova imagem com o comando `imagecreatefrompng()` e a imagem gerada é armazenada no *array* `imagemsSimbolos`.

Quadro 5 - Código para a criação de miniaturas

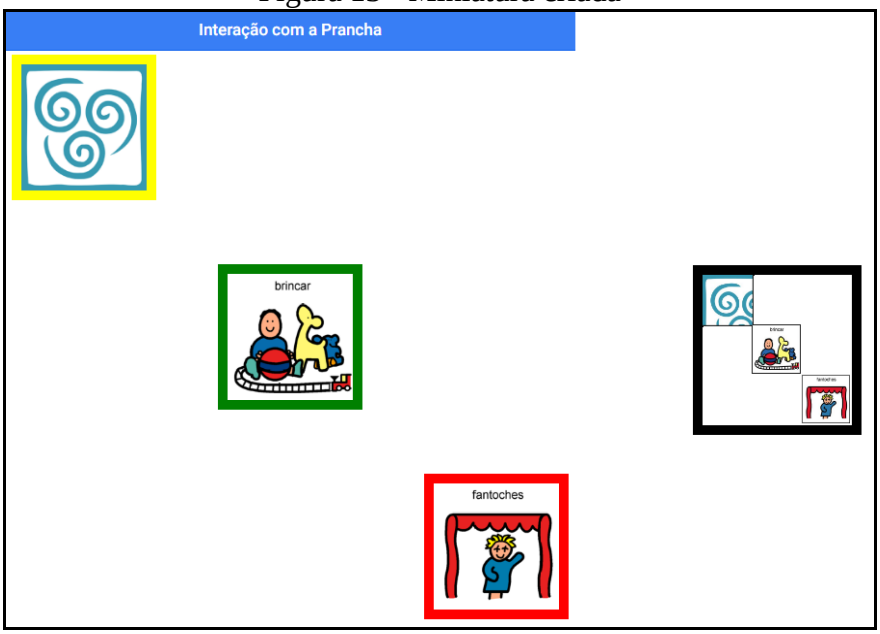
```
77      $imagens = array($simb1,$simb2,$simb3,$simb4,$simb5,$simb6,$simb7,$simb8,$simb9);
78      $numberoImages = count($imagens);
79      $colunas = 3;
80      $linhas = $numberoImages/$colunas;
81      $linhas = (int) $linhas;
82      $width = 200;
83      $height = 200;
84      $background = imagecreatetruecolor(($width*$colunas), ($height*$linhas)); // tamanho do canvas
85      $output_image = $background;
86
87      // Cria imagem
88      $imagensSibolos = array();
89      for($i = 0; $i < ($linhas * $colunas); $i++){
90          if (($imagens[$i] == '../img/no-symbol.png') == false){
91              $image_objects[$i] = imagecreatefrompng($imagens[$i]);
92          }
93      }
94
95      // Junta as imagens
96      $step = 0;
97      for($x = 0; $x < $colunas; $x++){
98        for($y = 0; $y < $linhas; $y++){
99          if(($imagensSibolos[$step] == null) == false){
100             imagecopymerge($output_image, $imagensSibolos[$step], ($width * $x), ($height * $y), 0, 0, $width, $height, 100);
101         }
102         $step++; // Incrementa o array de objetos
103       }
104     }
105
106     $imgdir = '../img/'.$pasta.'/'.$id.'.png';
107     $auddir = '../audio/no-sound.mp3';
108     imagejpeg($output_image, '../img/'.$pasta.'/'.$id.'.png');
109     imagedestroy($output_image);
```

Fonte: elaborado pelo autor.

Entre as linhas 96 e 104 está a criação de uma nova imagem composta pelas imagens da prancha e que será utilizada como símbolo da mesma. O código consiste em percorrer as linhas e colunas setadas no início no script e utilizar o comando `imagecopymerge()` para copiar as imagens armazenadas em `imagensSimbolos` para a imagem `output_image`. Nesse comando é feito o cálculo da posição e tamanho que a imagem deverá ocupar. Por fim, é criada uma imagem no diretório com base no `output_image`, e após isso o mesmo é

deletado, pois durante a criação ele é um arquivo fixo no diretório. Na Figura 13 é apresentado como fica a miniatura criada pelo script.

Figura 13 - Miniatura criada



Fonte: elaborado pelo autor.

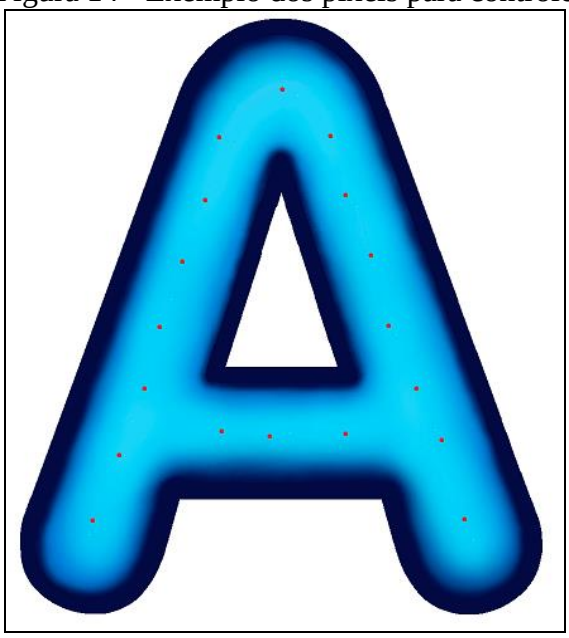
3.3.1.7 Exibição das pranchas de Letras e Números

As pranchas do Tagarela consistiam da exibição das imagens dos símbolos, mas para o jogo de Letras e Número as pranchas funcionam de forma diferente. Para tornar o jogo operacional e manter o controle do funcionamento das pranchas dentro de seus símbolos, foi utilizado os píxeis de cada imagem para efetuar o controle do jogo. Como cada pixel da imagem possui 4 canais, no caso o RGBA (*red*, *green*, *blue*, *alpha*), sendo que o *alpha* representa a transparência da imagem. Para implementar o controle de colisão definiu-se que o valor do canal *alpha* de cada pixel seria 255 para a parte interna e 0 para a externa da ilustração representada pela imagem.

O jogo possui símbolos que representam presas e predadores. Por possuir presas e predadores diferentes, era necessário definir quais seriam os símbolos que iriam representar cada presa e predador, bem como sua respectiva quantidade de vezes que deveriam aparecer na imagem. Para isso foram utilizados os 3 primeiros píxeis da primeira linha de cada imagem. O primeiro e o segundo pixel possuem valor *alpha* entre 1 e 5, que representam respectivamente qual o predador e qual a presa que irão aparecer para aquele símbolo. O terceiro pixel possui no valor de *alpha* a quantidade de presas que terão no jogo (no caso sempre se tem um único predador). O símbolo também possui alguns píxeis com *alpha* diferenciado, com valor de 253, para indicar em qual posição as imagens deverão aparecer,

conforme o exemplo da Figura 14, onde foram destacados alguns símbolos, pois não é possível diferenciar a olho nú essa variação de *alpha*.

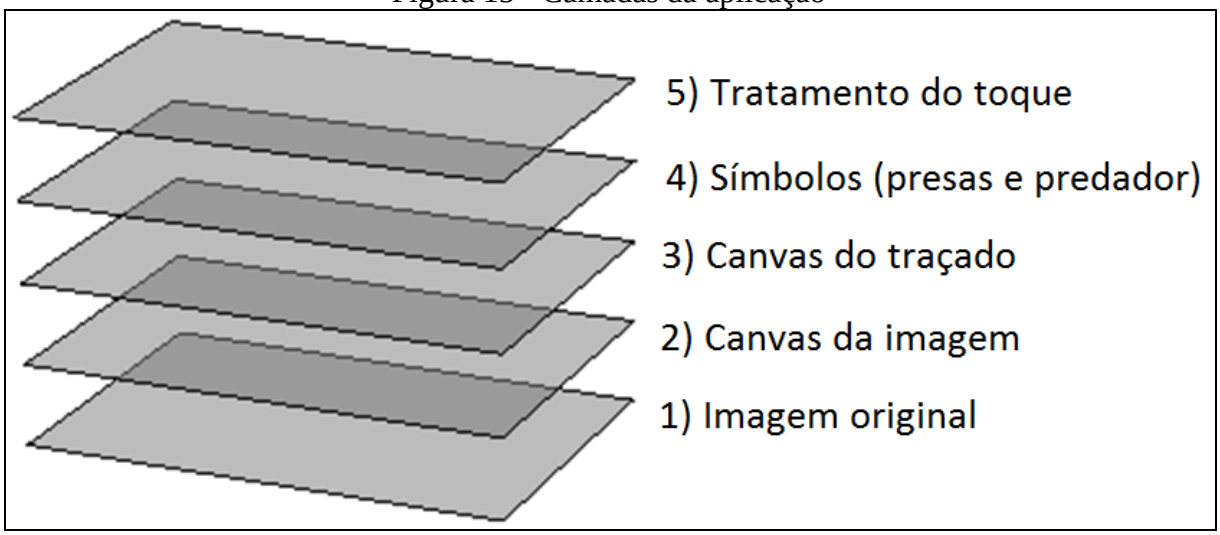
Figura 14 - Exemplo dos píxeis para controle



Fonte: elaborado pelo autor.

Com as imagens preparadas, já é possível a criação da prancha do jogo, que trabalha com 5 camadas distintas, conforme a Figura 15. A primeira camada é onde a imagem Portable Network Graphics (PNG) é carregada. Em seguida essa imagem é copiada para a segunda camada, redesenhada com Canvas. Esse desenho em canvas é necessário para se obter os dados de cada pixel via Javascript e poder processar em tempo real o jogo. Conforme o Quadro 6, após a leitura do script para recuperar a imagem, ela é desenhada com o identificador `minhaImagem`, na linha 45. Na linha 91 essa imagem é carregada na variável `image` e na linha 96 é desenhava em canvas com o nome `xtct`. Com essa variavel, é possível ler as informações de um determinado pixel com o comando `getImageData()`. Essa função retorna um *array* de 4 posições, sendo essas informações as propriedades *red*, *green*, *blue* e *alpha* do pixel.

Figura 15 - Camadas da aplicação



Fonte: elaborado pelo autor.

Quadro 6 - Javascript para leitura dos píxeis

```
35     $.ajax({
36         type     : "post",
37         url      : "http://tagarela.intonses.com.br/scripts/usar-pranchaLN.php",
38         data     : dados,
39         dataType : "json",
40         success  : function gravarLog(ret) {
41             $("body").removeClass("loading");
42             if (ret.erro) {
43                 alert(ret.msg);
44             }  else {
45                 var conteudo = "<img src='img/"+ret.simb_prancha+"' id='minhaImagem' style='width:400px;height:400px;'>";
46                 $("#imagemLN").append(conteudo);
47             }
48         },
```

```
91                  var image = document.getElementById("minhaImagem");
92                  var cnv = document.getElementById("canvas2");
93                  cnv.width = 400;
94                  cnv.height = 400;
95                  var ctxt = cnv.getContext('2d');
96                  ctxt.drawImage(image, 0, 0, 400, 400);
97                  var pixelData = ctxt.getImageData(event.offsetX, event.offsetY, 1, 1).data;
```

Fonte: elaborado pelo autor.

Após isso é criada a terceira camada. Essa camada é um Canvas e será utilizado durante o jogo, para desenhar o trajeto percorrido durante o jogo. A quarta camada é onde são apresentados os símbolos das presas e do predador. Com os dados lidos dos píxeis das imagens, é possível identificar quais são, quantos e suas posições. É necessário que eles fiquem em uma camada diferente do resto da aplicação para que os símbolos possam ficar acima do traçado realizado no jogo. A quinta e última camada é responsável por cobrir todas as demais e fazer o tratamento dos eventos de movimento na tela durante o jogo.

3.3.1.8 Eventos do jogo

Após fazer a leitura de todos os dados da imagem do símbolo e carregar essas informações, o jogo já pode ser executado. O controle do jogo é feito através dos eventos Javascript do *mouse*, conforme o Quadro 7. Entre as linhas 85 e 90 está o tratamento para o pressionar do botão do *mouse* (`onmousedown`), que define a variável `painting` como `true`, sendo essa variável utilizada para indicar que o usuário deseja percorrer o caminho, define o

`fillStyle` do Canvas para `#000000`, indicando que a cor ser utilizada será preta, e grava a posição no *mouse* nas variáveis `lastY` e `lastX`, sendo essas variáveis utilizadas para as coordenadas do desenho ao arrastar o *mouse*. Entre as linhas 92 e 94 está o tratamento ao soltar o *mouse*, alterando a variável `painting` para *false* e encerrando o controle do traçado. Após a linha 96 está o controle do traçado. Ela é acionada ao mover o *mouse*, com o evento `onmousemove`, verificando se a variável `painting` está como *true*. Nesse caso é criada a variável `image` com a imagem do símbolo e também é criada a variável `cnv` com o Canvas que será utilizada para recuperar as informações do pixel da imagem. A variável `pixelData` é carregada com as informações RGBA do pixel que o *mouse* está passando no momento. Já na linha 105 é verificado se o valor de `alpha` desse pixel é maior que 100 para definir o traçado do mouse.

Quadro 7 - Eventos do mouse

```
85            canvas.onmousedown = function(e) {
86                painting = true;
87                ctx.fillStyle = "#000000";
88                lastX = e.pageX - this.offsetLeft - 310;
89                lastY = e.pageY - this.offsetTop - 200;
90            };
91
92            canvas.onmouseup = function(e){
93                painting = false;
94            }
95
96            canvas.onmousemove = function(e) {
97                if (painting) {
98                    var image = document.getElementById("minhaImagem");
99                    var cnv = document.getElementById("canvas2");
100                   cnv.width = 500;
101                   cnv.height = 500;
102                   var ctxt = cnv.getContext('2d');
103                   ctxt.drawImage(image, 0, 0, 500, 500);
104                   var pixelData = ctxt.getImageData(event.offsetX, event.offsetY, 1, 1).data;
105               if(pixelData[3] > 100){
```

Fonte: elaborado pelo autor.

No Quadro 8 são apresentados os principais trechos de códigos do desenho do traçado. Ao mover o mouse é carregada a posição atual em que ele se encontra, após isso essas posições, e as posições anteriores, são armazenadas nas variáveis `x1`, `x2`, `x3` e `x4`. Na sequência são feitas algumas validações para identificar para que lado o mouse foi movido. Na linha 147 está o cálculo utilizado para identificar a distância do movimento do mouse e o tamanho da linha a ser desenhada. Entre as linhas 152 e 164 está o desenho do traçado, que é feito pixel a pixel entre as variáveis `x1` e `x2`. É utilizado o comando `fillRect` do Canvas, pois o objetivo é sempre desenhar um quadrado, para que possa ser visível ao menor movimento. No fim do código, as posições atuais são armazenadas para que possa continuar o tratamento do movimento.

Quadro 8 - Desenho do traçado

```
106                         mouseX = e.pageX - this.offsetLeft - 310;
107                         mouseY = e.pageY - this.offsetTop - 200;
108
109                         // find all points between
110                         var x1 = mouseX,
111                             x2 = lastX,
112                             y1 = mouseY,
113                             y2 = lastY;
```

```
147                         lineThickness = 20 - Math.sqrt((x2 - x1) *(x2-x1) + (y2 - y1) * (y2-y1))/10;
148                         if(lineThickness < 1){
149                             lineThickness = 1;
150                         }
151
152                         for (var x = x1; x < x2; x++) {
153                             if (steep) {
154                                 ctx.fillRect(y, x, lineThickness , lineThickness );
155                             } else {
156                                 ctx.fillRect(x, y, lineThickness , lineThickness );
157                             }
158
159                             error += de;
160                             if (error >= 0.5) {
161                                 y += yStep;
162                                 error -= 1.0;
163                             }
164                         }
165                         lastX = mouseX;
166                         lastY = mouseY;
```

Fonte: elaborado pelo autor.

3.3.2 Operacionalidade da implementação

A base de funcionamento do Tagarela é a interação com as suas pranchas. Além disso ele possui outras funções, como o *login*, criação de usuários, envio de convites, criação e visualização de observações. Como essas funções estão descritas no TCC do aplicativo original, essa seção de dedica a demonstrar as funções que tiveram o seu funcionamento alterado em relação ao aplicativo original.

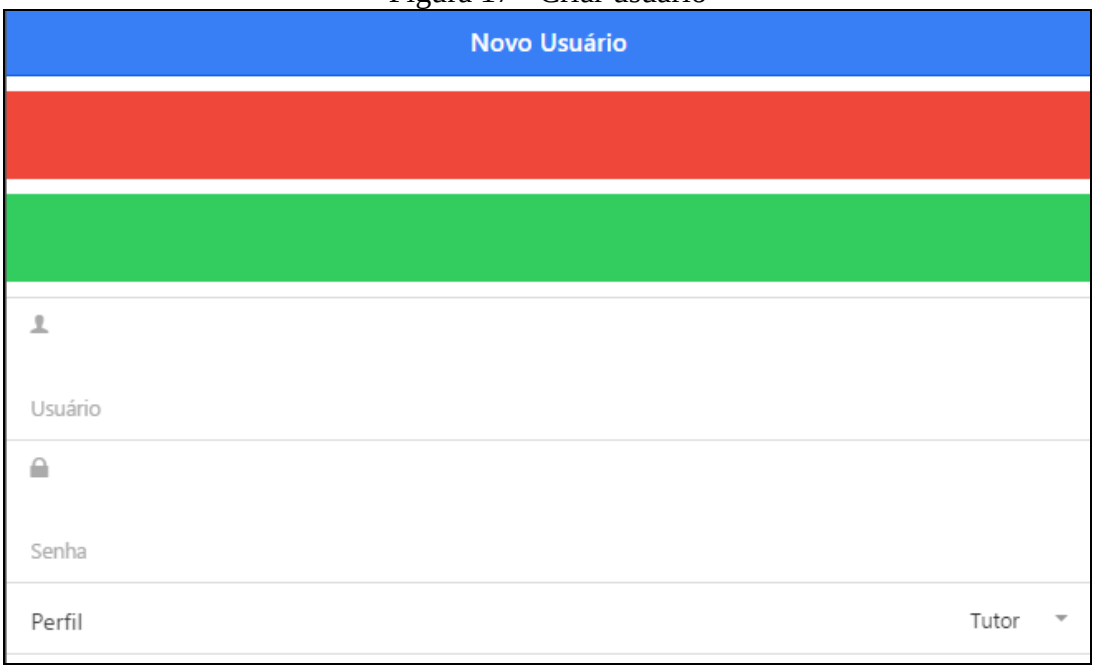
3.3.2.1 Criação de usuários

A tela principal do aplicativo possibilita o usuário realizar o *login* ou criar um novo usuário, conforme a Figura 16. Ao pressionar o símbolo `Adicionar Usuário`, será apresentada a tela de criação de usuários, conforme a Figura 17. Nessa tela o usuário deverá informar o nome de usuário, a senha e o tipo de perfil, que poderá ser `Tutor` ou `Paciente`. O usuário poderá confirmar a criação pressionando o botão verde, ou cancelar a ação e voltar para a tela de login pressionando o botão vermelho.

Figura 16 - Tela principal

Fonte: elaborado pelo autor.

Figura 17 - Criar usuário



Fonte: elaborado pelo autor.

3.3.2.2 Vínculo de usuários

Para que um TUTOR possa gerenciar as pranchas de um aluno, é necessário criar o vínculo entre eles. Esse vinculo é chamado de builder dentro da aplicação e todo aluno deve possuir um builder. Para criar esse vínculo, o PACIENTE deve enviar um convite para um TUTOR, podendo o TUTOR somente aceitar o convite. A Figura 18 abaixo mostra a tela de envio de convites. Ao criar um PACIENTE, já é criado automaticamente um builder para ele.

Figura 18 - Envio de convite

Fonte: elaborado pelo autor.

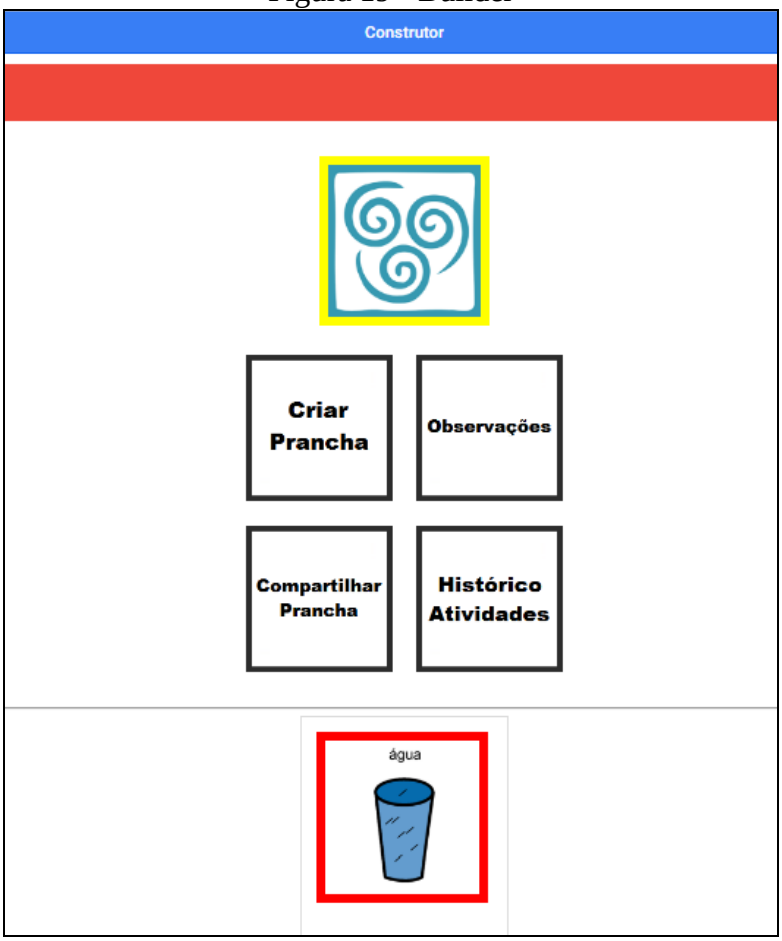
3.3.2.3 Criação das pranchas

A criação das pranchas é feita pela interface do builder, usuário responsável pelo gerenciamento das pranchas dos alunos. Na Figura 19 é apresentada a interface do builder, de onde ele pode criar uma nova prancha, gerenciar as observações, compartilhar pranchas criadas e verificar o histórico de atividades. Ao clicar em Criar Prancha o usuário é redirecionado para uma página onde deve escolher se irá criar uma prancha de comunicação, ou uma prancha do Jogo de Letras e Números.

Ao escolher uma prancha do Jogo de Letras e Número o usuário é direcionado para a seleção do símbolo, conforme a Figura 20. Nessa página o usuário poderá escolher entre as letras do alfabeto, maiúsculas e minúsculas, e números de 0 a 9. Ao clicar no símbolo desejado o usuário deve indicar a qual plano esse símbolo será vinculado, ou criar um novo

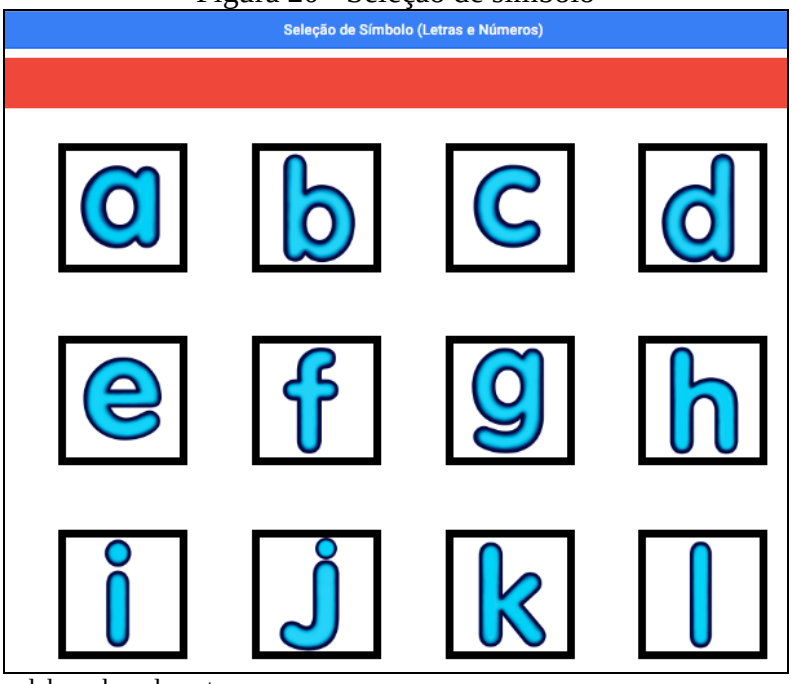
plano. Se optar por criar um novo, o usuário deverá escolher um símbolo para o plano e ele estará criado.

Figura 19 - Builder



Fonte: elaborado pelo autor.

Figura 20 - Seleção de símbolo

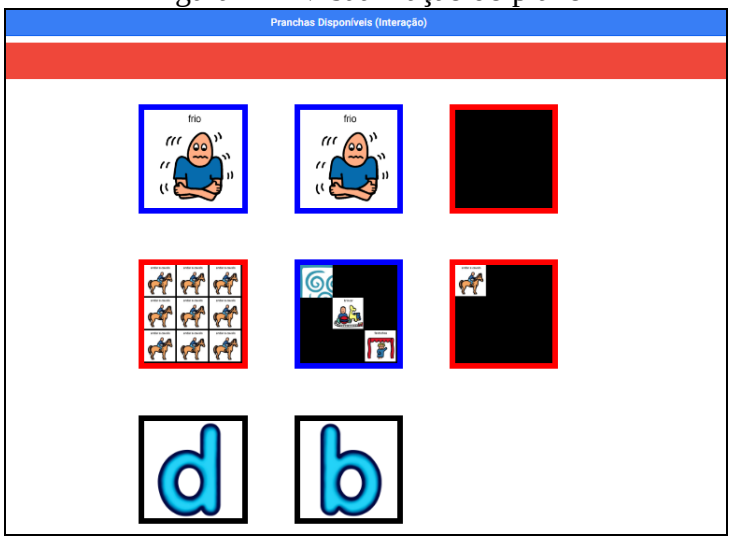


Fonte: elaborado pelo autor.

3.3.2.4 Utilização das pranchas

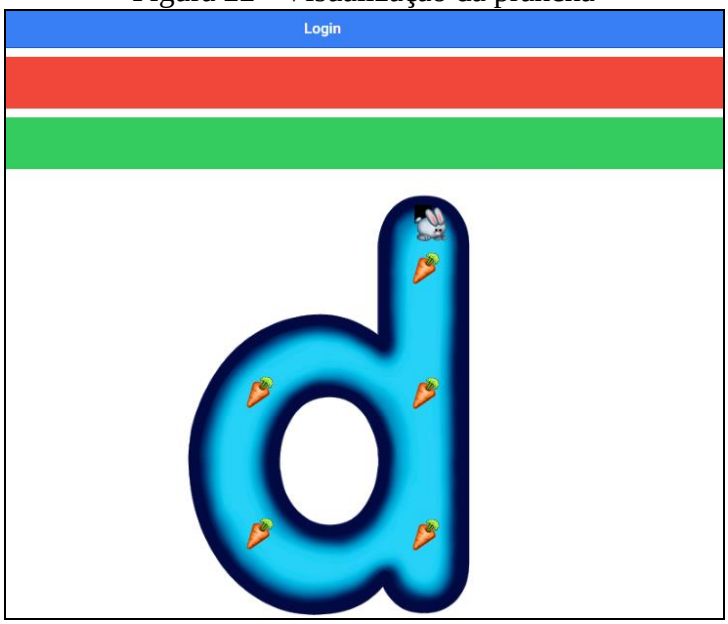
A utilização das pranchas é a principal atividade do Tagarela e é feita através da página do `builder`, visível na Figura 19. Nesta página o usuário tem, na parte inferior, uma lista de planos. O plano pode ser composto por pranchas de Comunicação e/ou pranchas do Jogo de Letras e Números, conforme a Figura 21. Ao clicar em uma prancha do Jogo de Letras e Números, o usuário inicia uma prancha do jogo, conforme a Figura 22, onde foram carregadas as presas (nesse caso, as cenouras) e o predador (o coelho).

Figura 21 - Visualização do plano



Fonte: elaborado pelo autor.

Figura 22 - Visualização da prancha

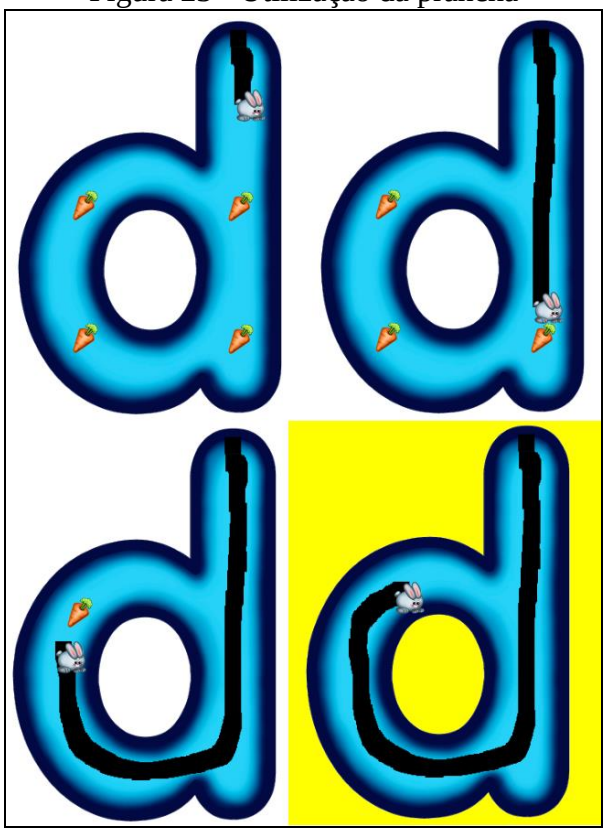


Fonte: elaborado pelo autor.

O coelho é controlado pelo usuário com o toque do mouse ou dedo. O objetivo é capturar todas as presas utilizando o predador, concluindo o desenho do símbolo. Sempre que o usuário passar com o predador por uma presa, ela irá sumir do símbolo. Ao capturar todas as

presas, o fundo do símbolo troca de cor, simbolizando a conclusão do jogo, conforme demostrado na Figura 23.

Figura 23 - Utilização da prancha



Fonte: elaborado pelo autor.

3.4 ANÁLISE DOS RESULTADOS

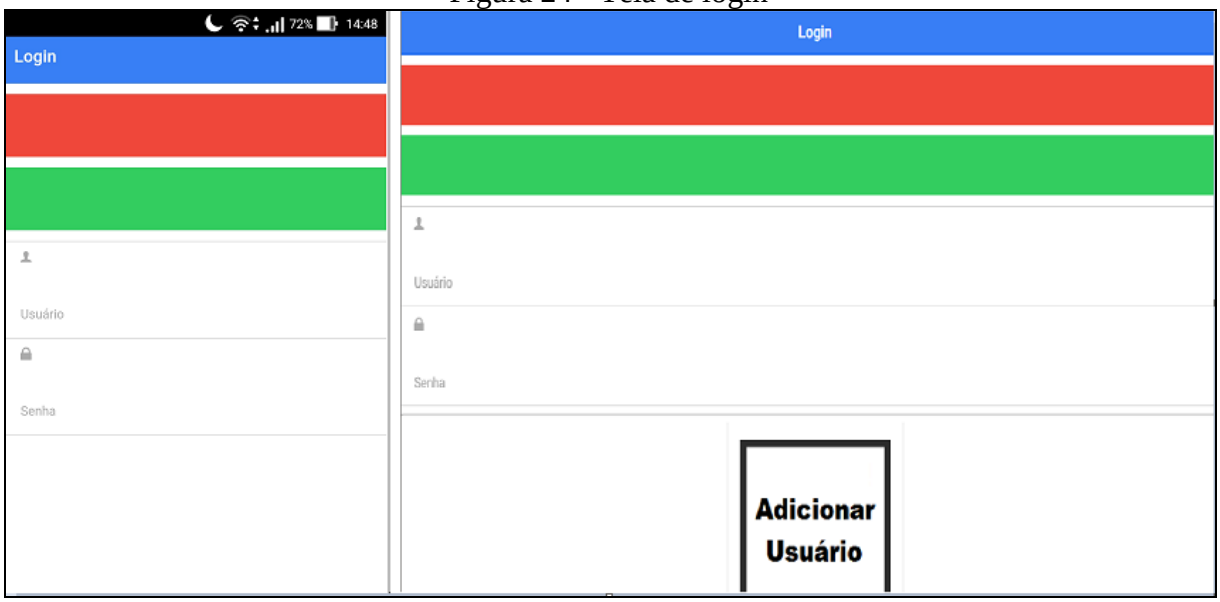
Os resultados deste trabalho foram obtidos com a análise de teste das funcionalidades do aplicativo, verificando o desempenho do aplicativo, principalmente em dispositivos móveis, e do correto funcionamento de todos os eventos do jogo. Os testes foram feitos utilizando navegadores para a versão *web* e um dispositivo Android para a versão móvel, garantindo que a interface se adapte corretamente a todas as plataformas sem comprometer o funcionamento do aplicativo, e garantir a compatibilidade dos plugins do Cordova.

3.4.1 Interface gráfica

O principal objetivo da utilização do Ionic foi criar uma padronização para a interface do sistema. Com a utilização desse *framework*, foi possível manter a aplicação *web* e a móvel com a mesma aparência. Da interação do usuário com a aplicação, o que se diferencia é a utilização de `plugins` para funcionalidades nativas do Javascript que não funcionam nos dispositivos móveis, como a `camera` e `media-capture`.

A Figura 24 apresenta a tela de login em smartphone Android e na plataforma *web*. Conforme pode ser verificado, a interface funciona da mesma maneira, com a mesma identidade visual. A principal diferença está no botão de Adicionar Usuário, que na versão *web* aparece logo abaixo do *login*, e na versão móvel é necessário rolar a página para cima.

Figura 24 - Tela de login

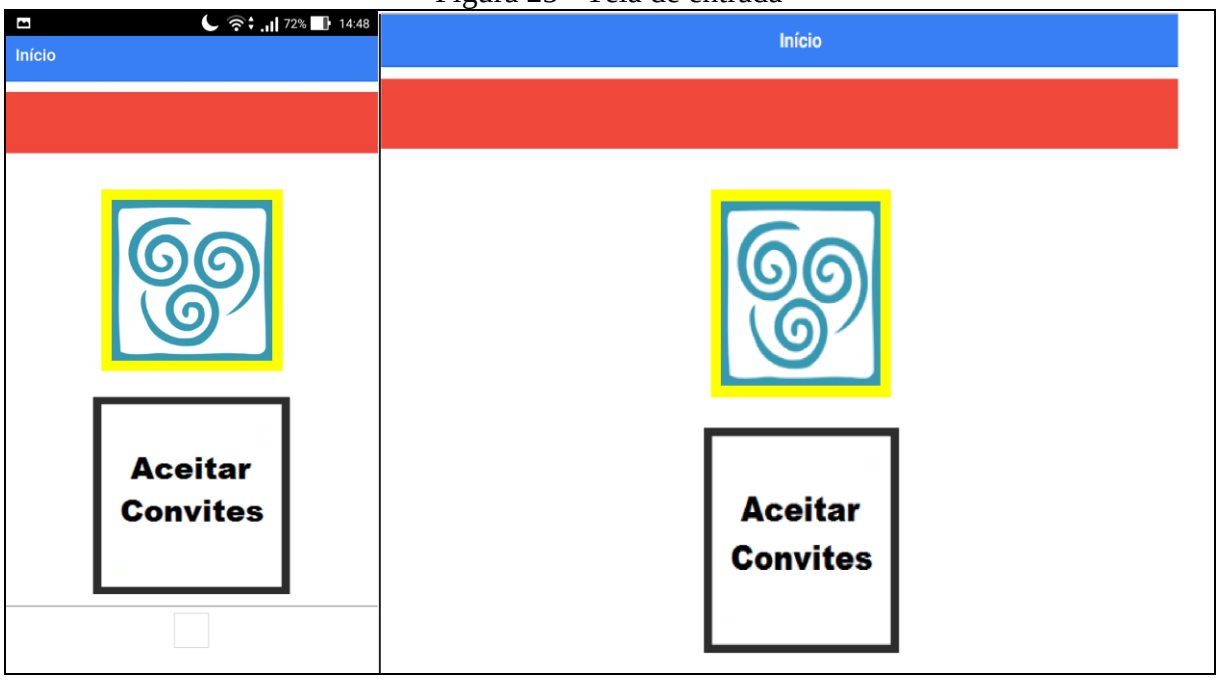


Fonte: elaborado pelo autor.

A Figura 25 apresenta a tela de entrada do aplicativo, que também é bem similar entre as versões, com diferença somente no final da página, onde é apresentado o símbolo que representa o `builder`. Na versão móvel é apresentado um quadrado vazio no local em que deveriam aparecer os `builders` quando não existe nenhum, enquanto na versão *web* não aparece nenhum símbolo nesses casos. Já a Figura 26 apresenta a tela do *builder* que são exatamente iguais em ambas as plataformas.

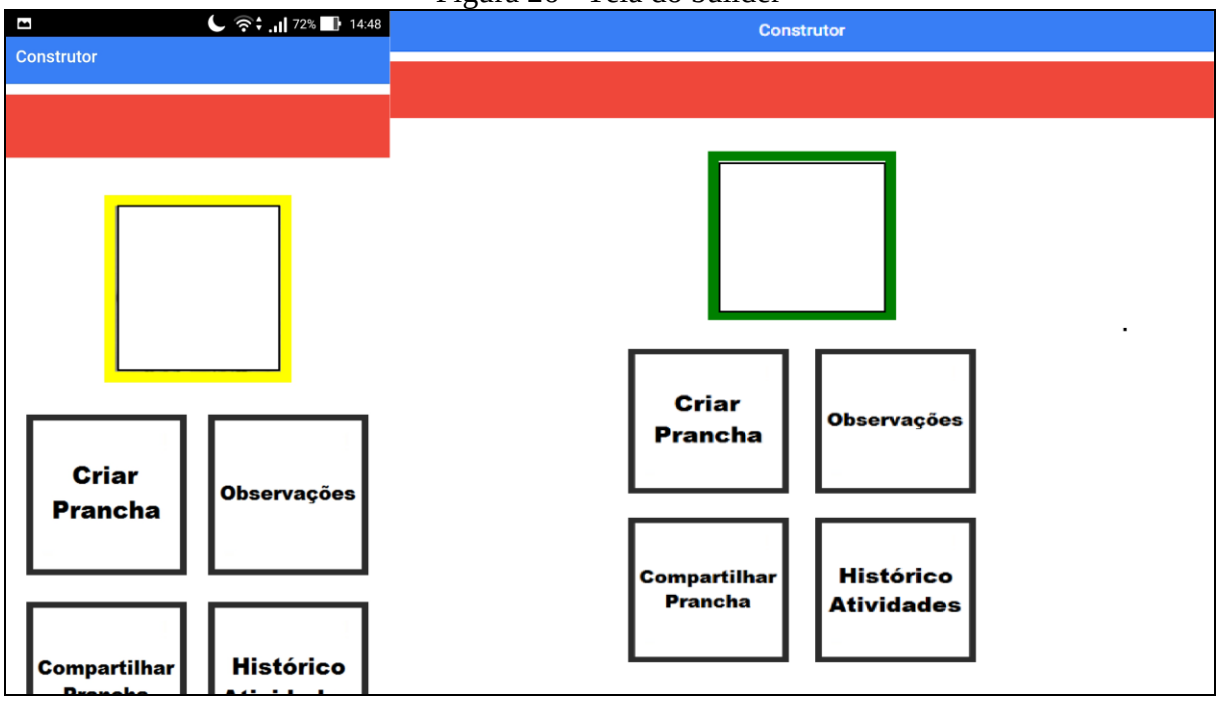
As telas do aplicativo ficaram bem similares em todas versões, tornando a experiência similar em ambas as plataformas, web e móvel. A principal diferença é o tamanho dos ícones nas telas, pois em um navegador se tem a mesma informação com tamanho maior.

Figura 25 - Tela de entrada



Fonte: elaborado pelo autor.

Figura 26 - Tela do builder



Fonte: elaborado pelo autor.

3.4.2 Teste de usabilidade

Foram feitos testes de usabilidade da aplicação pelo próprio autor, mas sugere-se fazer um teste de usabilidade do sistema com o público alvo do aplicativo, para verificar a facilidade e praticidade do jogo de Letras e Números no Tagarela. Para esse teste é indicado seguir os casos de teste descritos no Apêndice A. O primeiro passo é a identificação do usuário, através do questionário de perfil do usuário no Quadro 10.

Para os testes, é sugerida a utilização das instruções do Quadro 13, utilizando o roteiro das atividades descritas nos Quadros 11 e 12. Após os testes é sugerido ao usuário preencher o questionário usabilidade do Quadro 14.

3.4.3 Resultados obtidos

Este trabalho teve como objetivo a migração do aplicativo Tagarela para o Ionic, pensando em futuras implementações que poderão ser adaptadas e integradas a ele. Assim como a integração do Jogo de Letras e Números, adaptado de sua versão Android para a plataforma web e iOS, e assim permitindo ser distribuído para diversos smartphones e tablets.

A versão original do aplicativo fazia uma consistência dos dados para permitir a utilização do mesmo de forma *off-line*, atualizando os dados sempre que conectado. Para esse trabalho foi necessário tomar o cuidado de manter essa mesma forma de funcionamento, adicionando mais dados a serem sincronizados, sem haver perda de desempenho da aplicação. O grande agravante foi a adição do Jogo de Letras e Números, que deve gravar todos os dados do traçado desenhado pelo usuário, e que após muitas pranchas desenhadas *off-line* apresentava lentidão na sincronização. Uma sugestão para resolver esse problema seria alterar a estrutura e forma de gravação desses dados no banco.

A sincronização dos dados funciona muito bem quando um aplicativo novo é instalado sem nenhuma informação do banco de dados remoto, porém quando já existem dados no servidor essas informações se mostraram inconsistentes. Todos os dados inseridos *off-line* no aplicativo são marcados para serem sincronizados e quando o aplicativo ficar *on-line* esses dados são enviados ao servidor, apagados no aplicativo, e após isso todos os dados são baixados novamente. Esse processo gera inconsistências na instalação do aplicativo com dados remotos já inseridos no banco e é necessário ser revisto, pois se eu tenho dois aplicativos off-line e crio novos dados, eles irão incrementar o `id` desses elementos nas suas próprias tabelas e ao enviar esses dados para o servidor, esses `ids` serão alterados. Isso gera inconsistência nas chaves estrangeiras das outras tabelas que estavam associadas a eles. A sincronização das imagens do servidor com o aplicativo é feita com o *plugin* `file-transfer` junto com a sincronização do banco.

Também foi proposta a navegação entre as pranchas do Jogo de Letras e Números de dentro da própria prancha, mostrando a pranchas anterior e posterior, porem essa funcionalidade não conseguiu ser cumprido totalmente. Na versão web ele funcionou sem problemas, mas nos aplicativos móvel houveram alguns problemas, originados pelo problema de sincronização dos dados descrito acima.

3.4.4 Comparativo entre o trabalho desenvolvido e os trabalhos correlatos

O Quadro 9 apresenta a comparação entre os trabalhos correlatos apresentados e o trabalho proposto.

Quadro 9 - Trabalhos correlatos

Características	Participar (2014)	Livox (2015)	Scala (2015)	Trabalho Desenvolvido
Perfil de usuário	Não	Sim	Não	Sim
Plano de atividade	Não	Não	Sim	Sim
Histórico	Não	Sim	Sim	Sim
Gratuito	Sim	Não	Sim	Sim
Plataformas	Windows e Linux	Android	Android e Web	Android, iOS e Web

Fonte: elaborado pelo autor.

Dos trabalhos correlatos o que mais se assemelha ao desenvolvido é o Scala (2015), porém ele não possui perfil diferenciado para o tipo de usuário e não suporta a plataforma iOS, duas características que são atendidas pelo trabalho desenvolvido. Em relação ao perfil de usuário apenas o Livox (2015) possui essa funcionalidade, porém essa se limita ao perfil do usuário com deficiências, não sendo possível criar perfil para o profissional que irá atender esses usuários, um requisito que foi atendido pelo trabalho. O Plano de Atividades é atendido apenas pelo Scala (2015) e é bem semelhante ao Tagarela. O Livox (2015) é uma ferramenta paga, limitando a quantidade de pessoas que podem utilizar a ferramenta, e o Participar (2014) não possui histórico, não sendo possível acompanhar a evolução dos usuários.

4 CONCLUSÕES

Este trabalho apresentou a migração do aplicativo Tagarela para o *framework* Ionic e a integração do Jogo de Letras e Números para dentro dessa aplicação. O estudo mais aprofundado sobre os recursos do Phonegap permitiu a migração e integração dos dois aplicativos, mantendo o funcionamento original dos dois e seu funcionamento multiplataforma.

Outra característica do aplicativo era manter o visual de todas as plataformas o mais próximo possível, permitindo que o usuário se adapte rapidamente em caso de troca. Não foi possível manter o visual exatamente igual, pois existem particularidades de cada plataforma, inclusive do próprio *framework* utilizado, mas elas foram mantidas o mais próximo possível.

Também foi atingido o objetivo de utilizar somente os arquivos das imagens para armazenar os dados sobre as pranchas que as utilizam, como presas, suas posições e seu predador. Foi necessário obter o conhecimento sobre tratamento de imagens com HTML e Javascript, além de PHP que foi utilizado para fazer as alterações necessárias nas imagens, e no fim foi possível obter o resultado esperado. Esse mesmo conhecimento foi utilizado para automatizar a criação das miniaturas das pranchas.

As ferramentas utilizadas se mostraram adequadas para o objetivo de criar um aplicativo multiplataforma. O Phonegap é eficiente na geração de códigos nativos para os dispositivos móveis, e a combinação de HTML, CSS e Javascript facilitou o desenvolvimento e principalmente os testes, já que podia ser testado diretamente do navegador sem a necessidade de um dispositivo.

Durante o desenvolvimento foram encontrados alguns problemas, tais como a sincronização dos dados do aplicativo com o servidor, cujo objetivo era manter um único código para as duas plataformas, mas não foi possível identificar uma alternativa viável. Outro problema encontrado foi o tratamento dos eventos de toque para os dispositivos móveis, que são diferentes dos navegadores e precisam ser tratados de forma diferente. Também surgiram alguns problemas no redimensionamento das imagens, pois os píxeis delas são utilizados para o controle do jogo, e ao redimensionar a imagem o valor lido pelo Javascript não era o original. Foi possível resolver o problema criando camadas extras, ocultas para o usuário, mantendo as proporções originais.

A principal contribuição desse trabalho foi proporcionar a utilização de uma ferramenta multiplataforma, que permita a interação dos usuários com as pranchas, facilitando a sua comunicação e aprendizado da linguagem. Suas limitações estão na necessidade da

utilização de *plugins* para poder acessar os recursos nativos dos dispositivos, fazendo com que aplicativos que necessitam de vários recursos executem com baixo desempenho.

4.1 EXTENSÕES

São sugeridas as seguintes extensões para a continuidade do trabalho:

a) implementar uma persistência de dados integrada entre a versão móvel e web, permitindo a utilização de um único código fonte para as duas versões, ou carregar o Javascript dinamicamente, conforme a plataforma utilizada;
b) melhorar a navegação entre as páginas utilizando os recursos do AngularJS para o Ionic;
c) estender a aplicação para o Windows Phone;
d) melhorar a interface, com símbolos que possam substituir os textos existentes;
e) permitir ao usuário a criação de símbolos personalizados para o Jogo de Letras e Números, com um editor no aplicativo;
f) adaptar o aplicativo para se ajustar automaticamente a tela do dispositivo em que esta executando;
g) utilizar o *plugin crosswalk* para melhoria do desempenho da aplicação;
h) sincronizar utilizando *timestamp;*
i) criar um editor para novos símbolos para o Jogo de Letras e Números.

REFERÊNCIAS

BENGALALEGAL. **Livox**: software de cimunicação por imagens. [S.1.], 2012. Disponível em: < http://www.bengalalegal.com/livox>. Acesso em: 26 mar. 2016

CORDOVA. **Cordova**: Mobiles apps with HTML, CSS & JS. [S.1], 2015. Disponível em: < http:// cordova.apache.org>. Acesso em: 01 nov. 2016

FABENI, Alan Filipe C. **Tagarela**: Aplicativo para comunicação alternativa no iOS. 2012. 106 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

GESTAOESCOLAR. **A escola que ensina a todos** [S.1.], 2016. Disponível em: < http://gestaoescolar.org.br/aprendizagem/escola-ensina-todos-inclusao-necessidades-especiais-deficientes-politicas-publicas-flexibilizacao-508098.shtml>. Acesso em: 10 dez. 2016

INCLUSÃOJÁ. **Leis e documentos** [S.1.], 2013. Disponível em: < http://inclusaoja.com.br/legislacao/>. Acesso em: 02 abr. 2016

IONIC. **Ionic**: Advanced HTML5 Hybrid Mobile App Framework. [S.1], 2015. Disponível em: < http://ionicframework.com>. Acesso em: 26 mar. 2016

LIVOX. **Quem somos**. [S.1], 2015. Disponível em: < http://www.livox.com.br/quemsomos>. Acesso em: 26 mar. 2016

MARCO, Darlan Diego de. **Tagarela**: Aplicativo para comunicação alternativa na plataforma Android. 2014. 93 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

MARLUCE. **A leitura e a escrita na escola e os desafios atuais**. 2013. Disponível em: < http://pedagogiaaopedaletra.com/leitura-escrita-escola-desafios-atuais/>. Acesso em: 26 mar. 2016.

MINHAVIDA. **Autismo**. [S.1.], 2016. Disponível em: < http://www.minhavida.com.br/saude/temas/autismo>. Acesso em: 26 mar. 2016

MOZILLA. **Canvas tutorial**. [S.1.], 2016. Disponível em: < https://developer.mozilla.org/pt-BR/docs/Web/Guide/HTML/Canvas_tutorial>. Acesso em: 10 set. 2016

PARTICIPAR. **Software educacional "Participar".** [S.1.], 2014. Disponível em: < http://www.projetoparticipar.unb.br/images/material/descricao_do_Software_Participar.pdf>. Acesso em: 26 mar. 2016

PASSERINO, Liliane M.; BEZ, Maria R. **Comunicação alternativa**: Mediação para uma inclusão social a partir do Scala; Rio Grande do Sul: Editora UPF, 2015. 323 p.

PHONEGAP. **PhoneGap**. [S.1.], 2016. Disponível em: < http://phonegap.com>. Acesso em: 26 mar. 2016

REAB. **Atividades adaptadas para alunos com deficiência intelectual #janeiroreab**. [S.1.], 2015. Disponível em: < http://www.reab.me/atividades-adaptadas-para-alunos-com-deficiencia-intelectual-janeiroreab/>. Acesso em: 10 dez. 2016

REETZ, Wagner Jean. **Jogo de Letra/Números voltado para tecnologia assistiva no Android**. 2013. 63 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

SOARES, Jiane M. **A Importância do lúdico na educação**. 2013. Disponível em:< http://www.planetaeducacao.com.br/portal/imagens/artigos/diario/ARTIGO%20JIANE%20JOGO1.pdf>. Acesso em: 09 mai. 2016

SOUZA, Renata B. **O uso da tecnologia na educação**. 2013. Disponível em: < https://www.grupoa.com.br/revista-patio/artigo/5945/o-uso-das-tecnologias-na-educacao.aspx>. Acesso em: 03 abr. 2016.

UVA. **Como ocorre a aprendizagem da leitura na escrita**. [S.1.], 2007. Disponível em: <http://www.psicologia.pt/artigos/ver_artigo_licenciatura.php?codigo=TL0084>. Acesso em: 26 mar. 2016.

VIEIRA, Larissa de .S; OLIVEIRA, Valdiléia X. de **Como ocorre a aprendizagem da leitura na escrita**. 2010. Disponível em: < http://www.fecilcam.br/nupem/anais_v_epct/PDF/ciencias_humanas/21_VIEIRA_OLIVEIRA.pdf >. Acesso em: 05 mai. 2016.

WIPPEL, André Filipe. **Tagarela**: integração e melhorias no aplicativo de rede de comunicação alternativa. 2015. 64 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

APÊNDICE A – ROTEIRO DE TESTE SUGERIDO E AVALIAÇÃO DE USABILIDADE

Neste apêndice constam os questionários e o roteiro de testes sugeridos para os usuários. O questionário de perfil de usuário está no Quadro 10. No Quadro 11 esta o passo a passo para a criação de uma prancha e no Quadro 12 para a utilização dessa prancha. Já no Quadro 13 esta a lista de tarefas para o usuário poder testar as funcionalidades do aplicativo. No Quadro 14 consta o questionário de usabilidade do tutor.

Quadro 10 - Questionário sobre o perfil do usuário

PERFIL DE USUÁRIO

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

Sexo: ( ) Masculino ( ) Feminino

( ) Tutor **Área de atuação:** _____
( ) Aluno

Idade:	
( ) Tenho menos de 5 anos	( ) Tenho entre 20 e 25 anos
( ) Tenho entre 5 e 10 anos	( ) Tenho entre 25 e 30 anos
( ) Tenho entre 10 e 15 anos	( ) Tenho entre 30 e 35 anos
( ) Tenho entre 15 e 20 anos	( ) Tenho mais de 35 anos

Nível de escolaridade:
( ) Ensino fundamental incompleto
( ) Ensino fundamental completo – 1o grau
( ) Ensino médio incompleto
( ) Ensino médio completo – 2o grau
( ) Ensino superior incompleto
( ) Ensino superior completo
Observação: _____

Você utiliza o computador com qual frequência?
( ) Nunca utilizei
( ) Às vezes
( ) Frequentemente

Você utiliza dispositivos móveis com qual frequência?
( ) Nunca utilizei
( ) Às vezes
( ) Frequentemente

Fonte: elaborado pelo autor.

Quadro 11 - Roteiro para criação da prancha

Teste	Criação do plano
Ações	1 – Fazer `login` no aplicativo. 2 – Abrir um dos `builders` disponíveis. 3 – Pressionar o botão para criar uma nova prancha. 4 – Escolher a segunda opção, para criar uma prancha de Letras e Números. 5 – Escolher o símbolo desejado e adiciona-lo a um plano. Repetir os passos 3 a 5 até adicionar todos os símbolos desejados.

Fonte: elaborado pelo autor.

Quadro 12 - Roteiro para a utilização da prancha

Teste	Utilização do plano
Ações	1 – Na página do `builder`, abrir o plano desejado. 2 – Abrir a primeira prancha disponível. 3 – Arrastar o predador pelo símbolo até que todas as presas tenham sido removidas. 4 – Pressionar o símbolo para a próxima prancha. Repetir os passos 3 a 4 até completar todos os símbolos.

Fonte: elaborado pelo autor.

Quadro 13 - Instruções para o testa da aplicação

INSTRUÇÔES

Com este questionário buscamos avaliar o aplicativo Tagarela, qualificar a sua experiência de uso, assim como possíveis problemas e dificuldades que ele possa apresentar. Recomendados utilizar o aplicativo e fazer as atividades sugeridos, respondendo as perguntas relativas a cada teste efetuado.

Lista de tarefas a serem executadas pelo usuário:

1) Realize o cadastro de um novo tutor.
   A tarefa foi executada? ( ) Sim ( ) Não
   Observações_____
   _
   _____
   _
2) Realize o cadastro de um novo aluno e envie um convite para o tutor.
   A tarefa foi executada? ( ) Sim ( ) Não
   Observações_____
   _
   _____
   _
3) Faça o *login* com o tutor e aceite o convite do aluno.
   A tarefa foi executada? ( ) Sim ( ) Não
   Observações_____
   _
   _____
   _
4) Crie um no plano com os símbolos formando o nome do aluno, seguindo os passos do Quadro 11.
   A tarefa foi executada? ( ) Sim ( ) Não
   Observações_____
   _
   _____
   _
5) Faça o *login* com o aluno para a utilização do plano, seguindo os passos do Quadro 12.
   A tarefa foi executada? ( ) Sim ( ) Não
   Observações_____
   _
   _____
   _

Fonte: elaborado pelo autor.

Quadro 14 - Questionário de usabilidade

QUESTIONÁRIO DE USABILIDADE

1. Das tarefas solicitadas, quantas você conseguiu executar?
   ( ) Todas
   ( ) A maior parte delas
   ( ) Metade das tarefas
   ( ) Menos da metade das tarefas
   ( ) Nenhuma tarefa
2. De modo geral, você achou o "Módulo Jogo de Letras e Números" intuitivo e fácil de usar?
   ( ) Sim ( ) Não
3. Você achou fácil a forma que as atividades são realizados?
   ( ) Sim ( ) Não
4. Qual a sua avaliação do "Módulo Jogo de Letras e Números"?
   ( ) Muito bom
   ( ) Bom
   ( ) Regular
   ( ) Insatisfatório
   Observação: _____
5. Qual foi a sua dificuldade utilizando o "Módulo Jogo de Letras e Números"?
   _____
   _____
6. Você acha que os jogos, criados através do "Módulo Jogo de Letras e Números", juntamente com dispositivos móveis tornam o aprendizado mais atrativo?
   ( ) Sim ( ) Não
7. Você acha que o "Módulo Jogo de Letras e Números" pode ser utilizado para fins pedagógicos?
   ( ) Sim ( ) Não
8. Você acha que as atividades do "Módulo Jogo de Letras e Números", tornam o desenvolvimento do aluno mais atrativo?
   ( ) Sim ( ) Não
9. Você acha que as atividades do "Módulo Jogo de Letras e Números", podem auxiliar no desenvolvimento e evolução de pessoas com necessidades especiais?
   ( ) Sim ( ) Não

Fonte: elaborado pelo autor.