UNIVERSIDADE FEDERAL DO CEARÁ

INSTITUTO UNIVERSIDADE VIRTUAL

CURSO DE GRADUAÇÃO EM SISTEMAS E MÍDIAS DIGITAIS

ANTONIA HARETA ALVES FORTE

DESENVOLVIMENTO DE SISTEMA PARA AUXILIAR O PROCESSO DE AVALIAÇÃO EM AULAS DE ROBÓTICA EDUCACIONAL

FORTALEZA

2018

ANTONIA HARETA ALVES FORTE

DESENVOLVIMENTO DE SISTEMA PARA AUXILIAR O PROCESSO DE AVALIAÇÃO EM AULAS DE ROBÓTICA EDUCACIONAL

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas e Mídias Digitais do Instituto Universidade Virtual da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de Bacharel em Sistemas e Mídias Digitais.

Orientador: Prof. Dr. Leonardo Oliveira Moreira

FORTALEZA

2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

F841d Forte, Antonia Hareta Alves.
Desenvolvimento de sistema para auxiliar o processo de avaliação em aulas de robótica educacional / Antonia Hareta Alves Forte. – 2018.
52 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto UFC Virtual, Curso de Sistemas e Mídias Digitais, Fortaleza, 2018.
Orientação: Prof. Dr. Leonardo Oliveira Moreira.

1. Robótica Educacional. 2. Aplicativo. 3. Android. 4. Ionic. 5. Aplicação web. I. Título.
CDD 302.23

ANTONIA HARETA ALVES FORTE

DESENVOLVIMENTO DE SISTEMA PARA AUXILIAR O PROCESSO DE AVALIAÇÃO EM AULAS DE ROBÓTICA EDUCACIONAL

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas e Mídias Digitais do Instituto Universidade Virtual da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de Bacharel em Sistemas e Mídias Digitais.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Leonardo Oliveira Moreira (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Emanuel Ferreira Coutinho
Universidade Federal do Ceará (UFC)

Prof[a]. Maria Joelma Pereira Peixoto
Universidade Federal do Ceará (UFC)

Prof. Andrei Bosco Bezerra Torres
Universidade Federal do Ceará (UFC)

AGRADECIMENTOS

Primeiramente, agradeço a Deus por ser meu porto seguro e me dar forças para concluir mais essa etapa na minha vida. Agradeço aos meus pais, Aldenir e Marcelo, por me concederem todo o suporte necessário no decorrer dessa longa caminhada. Não posso deixar de agradecer também aos meus filhos de coração, meu cachorro Bruce e meu gato Luh, que me fizeram companhia durante as várias madrugadas em que fiquei acordada.

Agradeço aos meus amigos que me acompanham desde o ensino médio, David, Everton e Jonathan, além dos amigos que conheci na faculdade e que pretendo levar para a vida toda, Helder, Ana Carla, Italo Oliveira, Taynara, Matheus Silva, Alexandre e Rômulo.

Agradeço ao meu amigo Railan, que foi meu braço direito durante todo o desenvolvimento do projeto, principalmente nessa reta final. E especialmente ao meu professor e orientador Leonardo Oliveira pela orientação excelente e por toda ajuda e paciência no decorrer do desenvolvimento deste trabalho. Agradeço a todos os professores do curso de Sistemas e Mídias Digitais.

Não posso deixar de agradecer aos meus amigos, Marshall, Cláudio, Mercez, Rejane e Assis Neto do setor de TI do NUTEC que foi meu local de estágio durante 2 bons anos. Agradeço também ao Instituto Atlantico, onde atualmente trabalho, que tem me proporcionado uma grande chance de exercer e aperfeiçoar minhas habilidades, em especial ao meu gerente técnico Luiz Alves, às minhas gerentes de projeto Danielly e Katryne, e aos meus amigos Laís, Rafaella, Italo Leitão, Bruno, Luna, Alano, Matheus Braga, João, Rudolf, Hendy e Raphael.

Enfim, quero agradecer a todos aqueles que participaram direta e indiretamente dessa trajetória em minha vida acadêmica, estejam mencionados aqui ou não. Muito obrigado a todos vocês.

RESUMO

Este trabalho realizou um estudo sobre uso da robótica no âmbito escolar como elemento obrigatório da estrutura curricular e a utilização de um sistema para suprir as dificuldades enfrentadas pelos professores durante o processo de avaliação dos alunos. Têm-se uma visão geral da arquitetura utilizada e das tecnologias utilizadas para o desenvolvimento do sistema *web* e do aplicativo Sistema de Avaliação em Robótica Educativo (SARE). Além disso, uma descrição dos principais *plugins* do *Ionic Framework* e do *Apache Cordova* que foram utilizados para o desenvolvimento de algumas funcionalidades do aplicativo. Bem como uma descrição das principais funcionalidades de ambos os produtos, como por exemplo a realização do processo de avaliação e a sincronização dos dados entre a aplicação *web* e o aplicativo. O trabalho está dividido quatro etapas, na primeira etapa de caráter exploratório são descritas as principais funcionalidades a serem implementadas no sistema e no aplicativo. Já em um segundo momento é apresentado o processo de implementação do aplicativo, seus aspectos funcionais e os principais *plugins* utilizados. Na terceira parte é apresentado o processo de implementação do sistema e seus aspectos funcionais e a quarta parte deste trabalho relatada o processo de desenvolvimento da integração entre os dois produtos. Por fim é relatada a análise de desempenho realizada para avaliar o uso do aplicativo em sala de aula e os aspectos de desempenho da sincronização de dados entre as duas aplicações.

Palavras-chave: Robótica Educacional. Ionic. Android. Aplicação Web.

ABSTRACT

This study carried out about on the use of robotics in the school environment as a mandatory element of the curricular structure and the use of a system to overcome the difficulties faced by the teachers during the evaluation process of the students. It has an overview of the technologies used for the development of web systems and the application of the architecture used. In addition, a description of the main plugins of the Ionic Framework and Apache Cordova that were used to develop some application features. As well as a description of the main functionalities of both products, for example the realization of the evaluation process and the synchronization of the data between the web application and the application. The work is divided into four stages, in the first stage of exploratory character are described the main functionalities to be implemented in the system and in the application, and in a second moment the system implementation process and its functional aspects are presented. The third part presents the implementation process of the application, its functional aspects and the main plugins used. Finally, the performance analysis performed to evaluate the use of the classroom application and the performance aspects of data synchronization between the two applications is described.

Keywords: Educational Robotics. Ionic. Android. Web Application.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

LISTA DE ABREVIATURAS E SIGLAS

APIs	*Application Programming Interface*
CRUD	*Create, Read, Update e Delete*
CSS	*Cascading Style Sheets*
CSV	*Comma-separated values*
DOM	*Document Object Model*
EF	*Entity Framework*
FLL	*First League LEGO*
GPS	*Global Positioning System*
HTML	*Hypertext Markup Language*
HTML5	*Hypertext Markup Language, versão 5*
MCV	*Model–View–Controller*
MVVM	*Model-View-View-Model*
PoC	*Proof of Concept*
SARE	Sistema de Avaliação em Robótica Educativo
SMS	*Short Message Service*
SQL	*Structured Query Linguage*
TICs	Tecnologias da Informação e Comunicação
URL	*Uniform Resource Locator*

SUMÁRIO

1 INTRODUÇÃO

Há algumas décadas já se vem discutindo em diversos artigos sobre a inserção das Tecnologias da Informação e Comunicação (TICs) no ambiente educacional. Através dessas tecnologias busca-se melhorar a qualidade de ensino e o enriquecimento dos ambientes de aprendizagem. É notório observar que muitas das propostas e ideias de artefatos computacionais referem-se aos *softwares*. No entanto, a demanda por propostas mais voltadas para a vertente de *hardware* vem crescendo cada vez mais através da inserção da robótica para fins educacionais.

A robótica educacional constitui uma ferramenta que se encontra à disposição do professor, por meio da qual é possível demonstrar na prática muitos dos conceitos teóricos, às vezes de difícil compreensão, motivando tanto o professor como principalmente o aluno (SCHONS *et al.*, 2004). Assim, a robótica educacional pode desenvolver competências como raciocínio lógico; formulação e teste de hipóteses; relações interpessoais; resolução de problemas por meio de erros e acertos; criatividade; e capacidade crítica (ZILLI, 2004).

Seguindo esses conceitos, já existem escolas especializadas no ensino de robótica para o público infantil, e outras escolas estão incluindo esse conteúdo como um elemento obrigatório da sua estrutura curricular. O número de aplicações voltadas para o ensino de robótica está se tornando cada vez maior, como por exemplo o *LEGO MindStorm Programmer* voltado para o ensino de programação de comandos que serão executados pelos robôs da *LEGO*, e a empresa *ZOOM* voltada para a criação de materiais didáticos sobre robótica.

Em entrevista com o grupo de professores de uma escola particular de Fortaleza, foi relatada uma grande dificuldade no processo de avaliação, pois esse processo ocorre de forma bem dinâmica durante a aula, e diante desse contexto há uma carência de aplicativos que proporcionem uma avaliação mais qualitativa das práticas realizadas em sala de aula. Os aplicativos existentes são voltados para as competições da *LEGO* e são bem específicos para as tais, tornando a sua utilização em sala de aula inviável.

Diante disso, é levantada a seguinte questão: como um software pode auxiliar os professores no processo de avaliação das aulas de robótica no âmbito do ensino fundamental?

1.1 Objetivo Geral

Desenvolver a aplicação SARE para auxiliar e simplificar o processo de avaliação dos alunos nas aulas de robótica.

1.2 Objetivos Específicos

a) Estudar o processo de avaliação que atualmente é utilizado.
b) Definir um conjunto de requisitos que atendam às necessidades dos professores.
c) Definir as tecnologias utilizadas para a implementação da aplicação.
d) Desenvolver aplicativo *Android* que permita o professor realizar o processo de avaliação dos seus alunos.
e) Desenvolver aplicação *web* a realização dos cadastros principais de forma mais rápida e sincronização de dados.
f) Avaliar o uso da aplicação em sala de aula com os professores.
g) Avaliar os aspectos de desempenho da sincronização de dados entre as duas aplicações.

2 REFERENCIAL TEÓRICO

A robótica está muito mais próxima da vida das pessoas do que é possível imaginar, seja através do uso de eletrodomésticos ou de aparelhos eletrônicos. Uma máquina de lavar, tão comum nos lares, é um robô que executa uma tarefa doméstica que costuma ser árdua para a maioria das pessoas – lavar roupas. Por esse motivo, pela proximidade na vida cotidiana, a robótica pode ser uma forte aliada no processo de aquisição do conhecimento, pois possibilita uma aprendizagem ativa, dialogal e participativa, onde o aluno é o sujeito do seu processo de construção do conhecimento. Permite a união de vários recursos tecnológicos em situações de ensino-aprendizagem de uma forma lúdica e interessante, dando oportunidade de estimular e desenvolver atividades altamente relevantes para o currículo escolar (EXPOENTE, 2004).

Quando se fala de tecnologia na educação, o computador é o artefato que mais se destaca contexto e deve ser objeto de discussão e reflexão por parte de educadores para que seja utilizado e incorporado ao ensino, orientando o seu trabalho pedagógico (LÉVY, 1993). O autor Almeida também concorda com essa ideia da utilização do computador como uma ferramenta a serviço de um projeto pedagógico, porém ressalta que *"a tecnologia será importante principalmente porque irá nos forçar a fazer coisas novas, e não porque irá permitir que façamos melhor as coisas velhas"* (ALMEIDA, 2000). O professor, diante desse contexto de transformação, terá que desempenhar papéis diferentes, o que torna necessário novos modos de formação que possa ele esteja preparado para o uso pedagógico do computador, assim como para refletir sobre a sua prática.

O estudo feito pela autora Zilli, realizado nas escolas particulares e públicas de Curitiba, teve como objetivo analisar o uso da robótica educacional como recurso pedagógico, apontando as diversas formas como essa tecnologia é utilizada através de uma avaliação das perspectivas em relação ao processo cognitivo (ZILLI, 2004). Ainda segundo Zilli, os professores de uma das escolas analisadas seguem uma proposta mais tradicional, que é o resultado de vários métodos pedagógicos, em destaque o conteudismo. A figura do professor é fundamental, pois é ele o principal transmissor dos conteúdos. O aluno é avaliado pela quantidade de informação que absorve (PLANETA EDUCAÇÃO, 2004). Em outra escola, por exemplo, o professor responsável definiu a sua proposta pedagógica tem como objetivo fazer com que o aluno possa dialogar, duvidar, discutir, compartilhar informações, elaborar hipóteses, de forma a que se possam visualizar as diferenças, as contradições e que se permita espaço para a colaboração mútua e para a criatividade.

Segundo Perrenoud, em relação ao processo de avaliação, o autor critica o fato de algumas escolas que seguem uma proposta pedagógica mais tradicional, estarem mais preocupadas com a "restituição dos saberes assimilados" – dessa maneira utilizam com mais frequência as formas clássicas de exames, provas escritas ou chamada oral – do que na sua mobilização para a ação (PERRENOUD, 2000). Segundo essa abordagem, é mais fácil avaliar os conhecimentos de um aluno do que suas competências, que exige do professor observar o aluno em situações complexas, exige mais tempo e dá margens para a contestação (SILVA *et al.*, 2004).

Autores como Papert (PAPERT, 1994) e Tajra (TAJRA, 2001) defendem o uso de tecnologias na escola como recurso com o objetivo de auxiliar na construção de novos conceitos, possibilitando que o processo de aprendizagem aconteça de forma mais prazerosa. Por outro lado a utilização desta ferramenta em sala de aula ainda é um grande desafio na escola, já que muitos educadores não cresceram dentro deste contexto e tem que se adaptar a esta nova realidade.

Esta situação, no entanto leva aos professores o receio em usufruir ferramentas computacionais dentro de sua prática pedagógica. Já na área computacional existem muitos desenvolvedores focados na criação de aplicações voltados para a educação. Entretanto muitos desconsideram os aspectos pedagógicos e colaborativos envolvidos nessa tarefa. Desta forma, isso acaba muitas vezes fazendo com que sejam criados aplicativos que não são utilizados da maneira correta pelas crianças e pelos educadores (SANTOS; ROLIM, 2011).

Ao usufruir da utilização de dispositivos móveis, o educando estará aprimorando diferentes habilidades e competências como: coordenação fina e ampla, lateralidade, percepção visual (tamanho, cor, forma) e auditiva (SANTOS; ROLIM, 2011). Também estimula o desenvolvimento do raciocínio lógico, assim como noções de planejamento e organização, e quando voltados para os educadores se torna uma alternativa para ajudá-los com a adaptação a essa nova realidade.

Em 2009 foi realizada uma tese de doutorado (RoboEduc: uma metodologia de aprendizado com Robótica Educacional) muito relacionada ao objetivo geral deste trabalho, sobre o RoboEduc, um software educacional destinado ao ensino de robótica. Os principais objetivos deste *software* são auxiliar crianças que cursam as séries do Ensino Fundamental em relação a montagem, controle e programação de protótipos robóticos, e auxiliar professores do Ensino Fundamental a desenvolvem atividades para o uso de robôs com elemento mediador no processo de ensino-aprendizagem. Nesta mesma tese, Alzira (SILVA, 2009) explica que a avaliação do processo educacional está ligada à concepção educacional que está subjacente à

prática pedagógica, portanto, avaliar é uma atividade que envolve um conhecimento sobre quem é avaliado. É constituída de um movimento cíclico que envolve a ação, reflexão e retorno à ação.

No âmbito mais técnico, o artigo de Wahlbrinck (WAHLBRINCK K. A; BONIATI, 2017)faz uma análise do desenvolvimento de aplicações *mobile* híbridas, ou seja, são aplicativos parcialmente nativos e parcialmente *web mobile*, que podem ser baseados em *Hypertext Markup Language, versão 5* (HTML5) e outros padrões web e exibidos através do navegador embutido no aplicativo. Aplicações híbridas são populares porque permitem desenvolvimento multiplataforma, utilizando o mesmo código para mais de um sistema operacional (GONÇALVES, 2017).

Nesse artigo, os autores (WAHLBRINCK K. A; BONIATI, 2017) citam algumas vantagens das aplicações híbridas, como por exemplo a necessidade de uma curva de aprendizagem baixa, já que na maioria dos casos os *frameworks* utilizam linguagens *web*, a portabilidade da mesma aplicação para vários sistemas operacionais, baixo custo das ferramentas de desenvolvimento, que em sua grande maioria são *free* e *open source* e dependem apenas de um editor de código para desenvolvimento, e desenvolvimento mais rápido, visto que se utiliza o mesmo código para gerar o *build* da aplicação para diversos sistemas operacionais. Além disso, eles citam um caso de sucesso, que foi o desenvolvimento do aplicativo *Moodle Mobile* (a versão móvel do *site* do Moodle), que é uma plataforma de aprendizagem projetada para educadores, administradores e alunos com um sistema robusto, seguro e integrado para criar ambientes de aprendizagem personalizados.

A decisão em utilizar ou não um *framework* de desenvolvimento multiplataforma *mobile* deve levar em consideração muitos requisitos, como por exemplo o tempo disponível para desenvolver a aplicação, número de desenvolvedores que compõe o time de programação, plataformas a serem atingidas, público alvo da aplicação e linguagem utilizada para desenvolvimento. Estes são fatores essenciais que precisam serem analisados no desenvolvimento de um aplicativo móvel (WAHLBRINCK, 2015), e que foram utilizados como base na metodologia deste trabalho.

3 METODOLOGIA

Em primeiro momento foi feita uma análise das principais funcionalidades que os professores precisavam no aplicativo. Essa análise foi feita através dos resultados de um questionário que foi aplicado aos professores de robótica da escola particular de Fortaleza, onde foi realizada a entrevista inicial. Por meio desse questionário inicial descobriu-se que as principais funcionalidades do aplicativo seriam a realização das avaliações e a criação de gráficos, nos quais os professores consigam ver tanto o desempenho do aluno quanto da turma no geral. Outro fator importante é a necessidade do aplicativo não depender de conexão com a internet, pois em muitas escolas esse recurso é muito instável, e isso poderia inviabilizar o uso do aplicativo.

Depois dessa análise inicial foi necessária uma análise mais técnica, que tem o objetivo de definir quais tecnologias seriam utilizadas para desenvolver o aplicativo. Para definir isto foram realizadas provas de conceito com os *frameworks Phonegap* e *Ionic*. Estes dois *frameworks* foram escolhidos pois ambos resultam em aplicativos híbridos, ou seja, são parcialmente nativos e parcialmente *web apps*, assim permite desenvolvimento multiplataforma, utilizando o mesmo *Hypertext Markup Language* (HTML) para diferentes sistemas operacionais. Essas características tornam o aprendizado e o custo de tempo para desenvolvimento mais rápidos comparados com o desenvolvimento de uma aplicação nativa.

O objetivo de cada *Proof of Concept* (PoC) era desenvolver um cadastro completo com conexão à banco de dados em ambos os *frameworks* e analisar as dificuldades durante o processo, o tempo que foi gasto, a presença ou não de uma boa documentação. Com base nisso o *Ionic* foi escolhido, pois em diversos momentos o *Phonegap* se tornou muito mais complexo do que o *Ionic* na realização de tarefas simples, como por exemplo, na criação de um simples cadastro. O *Ionic* também permitiu a criação de uma interface de forma mais rápida devido ao fato de que ele fornece vários componentes prontos e de fácil utilização. O *Ionic* é baseado no *Angular*, um *framework javascript* amplamente adotado e este fato traz um enorme ganho de produtividade para os desenvolvedores. E por último, a documentação do *Ionic* é bem mais atualizada e a comunidade de desenvolvedores bem participativa do que o *Phonegap*.

Depois de concluída essa definição das tecnologias, foi realizada uma priorização das funcionalidades com base na importância de cada uma delas para a aplicação, assim viabilizando o início do desenvolvimento. A primeira versão do aplicativo contém as seguintes funcionalidades:

a) Cadastro de Escolas, Turmas, Alunos.

   Essa funcionalidade permite que o professor consiga gerenciar todo o processo de avaliação de várias turmas, mesmo que sejam de escolas diferentes.

b) Criação de grupos de avaliação.

   O processo de avaliação utilizado na escola consiste em dividir a turma em grupos de quatro alunos, no qual cada aluno terá uma função diferente. Essas funções podem ser: construtor, organizador, programador e líder. Cada função é avaliada com critérios diferentes, e o grupo só será mudado depois que todos os alunos de cada grupo passarem por todas as funções. Isso permite que o professor perceba em quais aspectos os alunos têm mais dificuldades.

c) Realização da avaliação.

   Essa é uma das funcionalidades principais do aplicativo, no qual o professor responde várias perguntas referentes ao desempenho de cada aluno do grupo. No aplicativo esse processo é divido em etapas, na qual cada etapa representa a função que o aluno do grupo desempenhou naquela aula.

d) Gráficos.

   Ao final de cada avaliação é mostrado um gráfico geral dos resultados do grupo avaliado. Posteriormente o professor também pode visualizar novamente esse gráfico e gerar outros gráficos que podem mostrar todos os resultados de cada aluno ou com os resultados gerais da turma.

Para garantir que a aplicação realmente está correspondendo com as expectativas e as necessidades dos professores, ao final do desenvolvimento dessas funcionalidades foram realizados alguns testes que resultaram em novas funcionalidades que serão desenvolvidas. Uma delas é uma aplicação *web* que também permitirá de forma mais prática o cadastro das escolas, turmas, e alunos. Segundo os professores, é muito trabalhoso fazer todos esses cadastros através do celular, e seria mais rápido por meio do computador. Essa aplicação web também permitirá que o professor realize o cadastro desses registros por meio da importação de arquivos *Comma-separated values* (CSV). Para isso se tornar ainda mais prático para o usuário, a aplicação irá disponibilizar o modelo do arquivo que permitirá a importação desses dados.

Outra funcionalidade interessante é a possibilidade de adicionar fotos dos alunos, pois muitas vezes existem alunos com o mesmo nome em uma mesma turma, então o uso da foto torna bem mais fácil a identificação do aluno no aplicativo. Como explicado anteriormente, o

aplicativo não possui conexão com a *internet*, e percebeu-se que isso também poderia ser um problema, porque caso o dispositivo móvel fosse formatado ou o aplicativo desinstalado, os dados seriam perdidos. Para manter o requisito inicial de uma independência de *internet*, mas também garantir a segurança dos dados, o aplicativo possui uma opção de sincronização de dados, na qual permiti que quando o aparelho estiver conectado à rede, os dados sejam sincronizados com uma base de dados *online*. Este item será abordado com mais detalhes no capítulo 6.

Nos próximos capítulos deste trabalho será descrito mais tecnicamente como foram implementadas essas novas alterações no aplicativo, e sobre as principais funcionalidades que foram desenvolvidas para a aplicação *web*.

4 IMPLEMENTAÇÃO DO APLICATIVO

Para a codificação do aplicativo SARE foi utilizado *Ionic Framework*, na versão 3.20. Este *framework* permite o desenvolvimento de um aplicativo híbrido, ou seja, um aplicativo *web* incorporado em um aplicativo nativo que fornece uma ponte para o sistema operacional e os serviços nativos do dispositivo (Hartmann et al., 2011). O código produzido em uma aplicação híbrida é executado dentro de um recurso chamado *webview*, um tipo especial de *browser* que é executado quando a aplicação híbrida é requisitada pelo usuário. Essa forma de execução sobre o *webview* não fica explícita para o usuário, deixando a impressão de que o aplicativo se comporta da mesma forma que um aplicativo nativo. O nome híbrido é dado justamente pela característica da junção de código nativo para empacotamento e distribuição do aplicativo com o código não nativo, que é composto por HTML5, *Cascading Style Sheets* (CSS) e *JavaScript*, responsável pelo visual e funcionalidades da aplicação (GONÇALVES, 2017).

Para acessar os recursos nativos dos dispositivos em diferentes plataformas, o *Ionic* trabalha em conjunto com um outro *framework*, o *Apache Cordova*. *Cordova* é uma plataforma de desenvolvimento móvel com *Application Programming Interface* (APIs) que permitem que o desenvolvedor acesse funções nativas do dispositivo, como câmera ou *Global Positioning System* (GPS). Quando se cria uma aplicação com o *Ionic*, por padrão não são adicionados *plugins* do *Cordova* para que o aplicativo não fique sobrecarregado com *plugins* que não serão utilizados. No caso do aplicativo SARE, são utilizados *plugins* que fornecem acesso a câmera, envio de *Short Message Service* (SMS), e que permitem obter as informações se o celular está conectado com a *internet*.

Sendo assim, o *Cordova* é uma camada inferior, responsável pelo acesso aos recursos nativos dos dispositivos e pelo empacotamento da aplicação para as diferentes plataformas. O *Ionic* é a camada superior, focada na interface gráfica de usuário e regras de negócios da aplicação.

Além da utilização do *Cordova*, o *Ionic* também conta com o *Angular*, que se trata de um *framework open source* mantido pela *Google* e comunidade, que visa abstrair a manipulação do *Document Object Model* (DOM), separando a lógica da *view* do aplicativo. O desenvolvimento pode ser feito utilizando *Typescript* sob o padrão *Model-View-View-Model* (MVVM). Neste padrão, o modelo (*Model*) representa os dados e a lógica de negócio. A (*view*) representa a interface gráfica, geralmente definida de forma declarativa, e para cada *view*, existe um modelo de visão (*ViewModel*) associado.

4.1 Modelagem do banco de dados

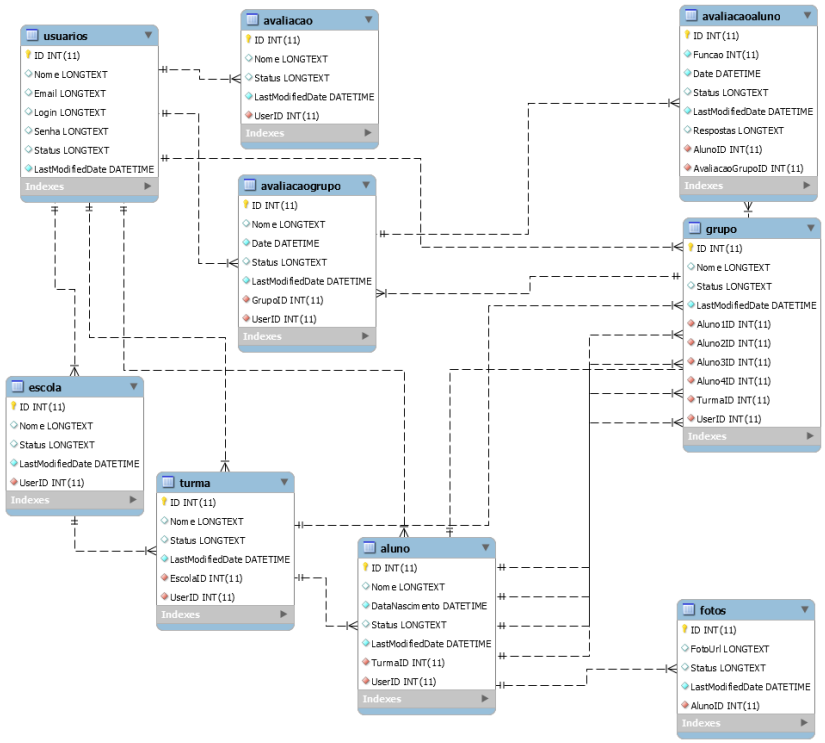
Os dados cadastrados no aplicativo são persistidos em um banco de dados interno e oficial da plataforma *Android*, o *SQLite*[1], no qual, com ele é possível modelar uma estrutura de tabelas relacionadas entre si. Para sua utilização em um projeto do *Ionic* é necessário adicionar os seguintes *plugins*:

ionic cordova plugin add cordova-sqlite-storage

npm install –save @ionic-native/sqlite

Na Figura 1 é apresentada a representação da modelagem utilizada para a base de dados utilizada no aplicativo.

Figura 1 – Modelagem do bando de dados do aplicativo



Fonte: Elaborada pelo autor.

O banco de dados do aplicativo tem o objetivo de ser um espelho do banco de dados da aplicação *web*, então devido a esse motivo a sua modelagem possui algumas particularidades que auxiliam na integração de dados entre as duas bases de dados. O *id* de cada registro salvo é um numero negativo composto por ano, mês, dia, hora, minuto e milissegundos, dessa maneira,

[1] SQLite. Disponível em <https://ionicframework.com/docs/native/sqlite/>. Acesso em: 08 junho de 2018

ao realizar a sincronização de dados é possível identificar os registros que ainda não foram sincronizados. O campo *status* serve para identificar uma ação realizada, se o registro foi adicionado, atualizado, deletado ou se já foi sincronizado. No Código-fonte 1 é apresentado um trecho do código utilizado para a definição dos *enums*, e na Figura 2 é apresentado um exemplo do registro salva no banco de dados.

Código-fonte 1 – Enum.ts: Classe que contém os enums utilizados.

```
export enum Status {
  added = 1,
  updated = 2,
  sync = 3
}
```

Figura 2 – Exemplo de registro salvo na tabela de turmas

Fonte: Elaborada pelo autor.

4.2 Aspectos funcionais

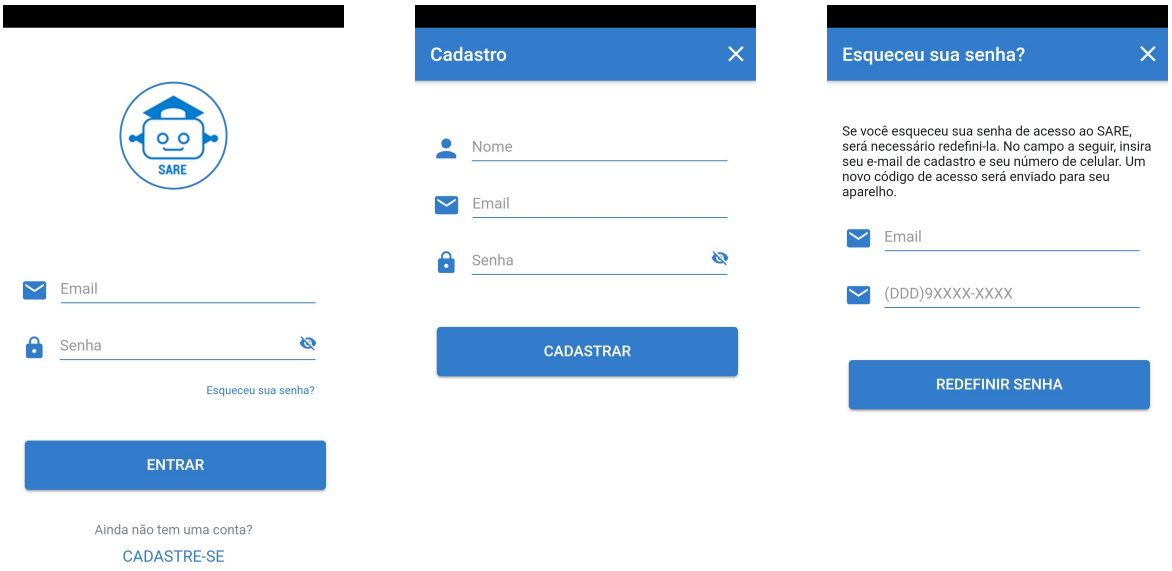
De maneira geral, o aplicativo SARE possui as seguintes funcionalidades: *login* de usuários no aplicativo, cadastros básicos para o gerenciamento das informações (cadastro de usuários, escolas, turmas, alunos e grupos), realização do processo de avaliação, e geração de gráficos para visualização do desempenho dos alunos.

4.2.1 Login e Cadastro de Usuários

Na tela inicial é possível realizar o acesso ao aplicativo informando os dados de um usuário já cadastrado, ou realizando o cadastro de um novo usuário. Para concluir o cadastro

de um novo usuário, o aplicativo realiza uma validação para verificar se o *e-mail* já existe na base de dados. Além disso, é possível gerar uma nova senha caso o usuário esqueça a senha que cadastrou inicialmente. A nova senha é gerada de forma aleatória e enviada via *SMS* para o número informado pelo usuário. Na Figura 3 estão as imagens referentes as telas iniciais do aplicativo, no qual a Imagem 3a representa a tela inicial de *login* do aplicativo, a Imagem 3b representa a tela de cadastro de um novo usuário, e a Imagem 3c representa a tela de recuperação de senha via *SMS*.

Figura 3 – Telas de *login* e cadastro de usuários



(a) Tela de *Login*

(b) Tela de cadastro de novo usuário

(c) Tela de redefinição de senha

Fonte: Elaborada pelo autor.

A implementação do envio de *SMS* foi feita utilizando o seguinte *plugin*: *cordova-sms-plugin*[2]. Para utilizar este *plugin* é necessário primeiro instala-lo ao projeto, por meio dos seguintes comandos:

ionic cordova plugin add cordova-sms-plugin

npm install –save @ionic-native/sms

No Código-fonte 2 é apresentado o código utilizado para realizar o envio de *SMS*.

[2] cordova-sms-plugin. Disponível em <https://ionicframework.com/docs/native/sms/>. Acesso em: 08 junho de 2018

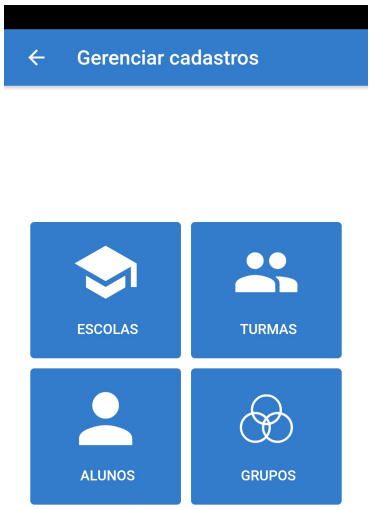
Código-fonte 2 – RedefinirSenha.ts: Código de envio de *SMS*.

```
redefinirSenha(){
    var options = {
      replaceLineBreaks: false,
      android: {
        intent: "INTENT"
      }
    };

    this.sms.send(this.telefone, "SARE: Uma nova senha foi
       gerada: " + novaSenha, options)
    .then( result =>
      console.log(result))
    .catch( error =>
      console.log(error));
}
```

4.2.2 Cadastros básicos

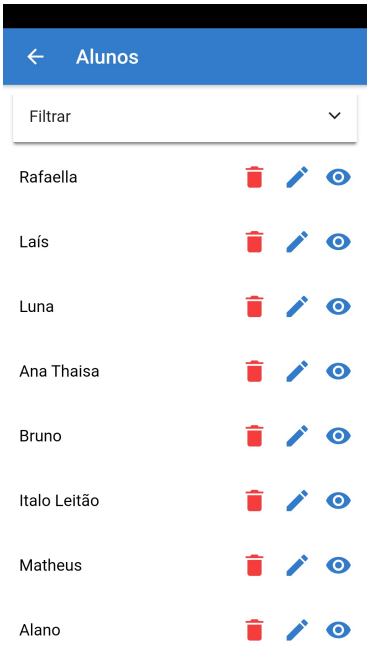
Após acessar aplicativo, o usuário pode realizar os cadastros das informações que podem ser utilizadas durante o processo da avaliação. Nessas telas de gerenciamento é possível realizar as funcionalidades de um *Create, Read, Update e Delete* (CRUD), ou seja, representa a criação, edição, visualização e exclusão um registro. No aplicativo é possível realizar o cadastro de escolas, turmas, alunos e grupos. Na Figura 4 é apresentada as principais telas de cadastros do aplicativo, na qual a Imagem 4a representa a tela inicial de gerenciamento dos cadastros, na Imagem 4b representa a tela de visualização das escolas cadastradas. A Imagem 4c representa a tela de visualização dos alunos cadastrados e na Imagem 4d é apresentada a tela de visualização dos grupos cadastrados.

Figura 4 – Tela de cadastros básicos

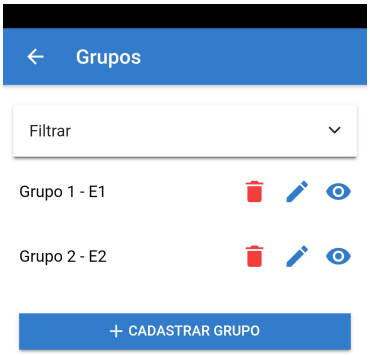


(a) Tela de cadastros básicos

(b) Tela de visualização das escolas cadastradas



(c) Tela de visualização dos alunos cadastrados básicos



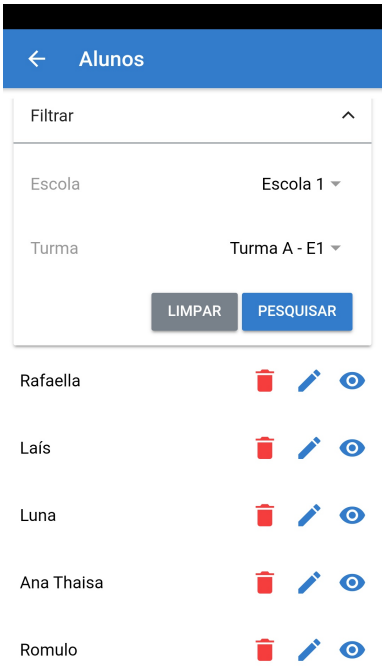
(d) Tela de visualização dos grupos cadastradas

Fonte: Elaborada pelo autor.

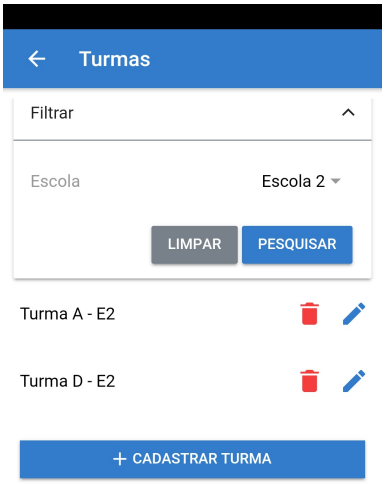
Além disso é possível listar os registros de acordo os filtros pré-definidos em cada tela. Por exemplo: no caso da listagem de alunos, é possível pesquisar os alunos de acordo com a escola e/ou a turma selecionadas. Na Figura 5 são apresentadas imagens das telas de visualização

dos alunos e turmas cadastrados.

Figura 5 – Telas de visualização de dados com filtro de buscas



(a) Tela de visualização dos alunos cadastrados básicos



(b) Tela de visualização das turmas cadastradas

Fonte: Elaborada pelo autor.

É válido ressaltar que o cadastro do grupo tem como base o modelo avaliativo que foi resultado do trabalho desenvolvido por uma professora de robótica de Fortaleza, baseado em métricas já existentes e utilizadas no processo de avaliação do torneio de robótica, *First League LEGO* (FLL). Nesse modelo, a avaliação é realizada com base em um grupo, que é composto por quatro alunos, no qual cada aluno desempenha um papel diferente durante o processo de realização da atividade. Os papéis podem ser: construtor, organizador, programador, e líder.

No caso do cadastro dos alunos é possível realizar também o *upload* ou a captura de uma foto do aluno, e para isso foi utilizado os seguintes *plugins*:

ionic cordova plugin add cordova-plugin-camera

npm install –save @ionic-native/camera

4.2.3 Processo de avaliação

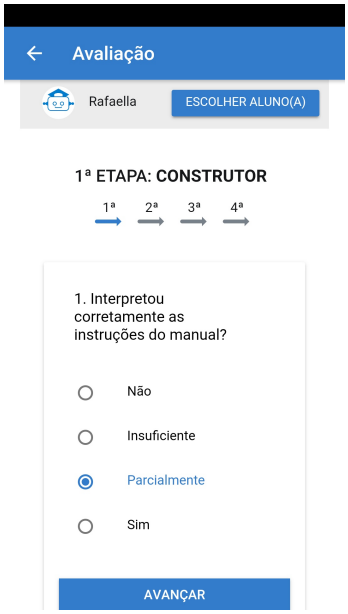
O fluxo de avaliação é uma das principais funcionalidades do aplicativo, no qual pode ser iniciado escolhendo dados já cadastrados, ou cadastrando as principais informações ao longo do processo, como por exemplo, os alunos que estão participando da avaliação.

4.2.3.1 Avaliação com dados cadastrados

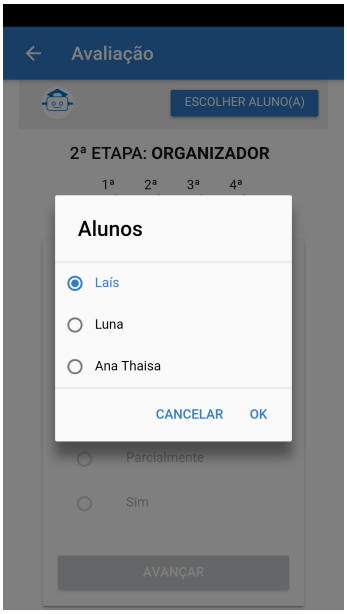
Na Figura 6 são apresentadas as principais telas referentes ao processo de avaliação com dados cadastrados, na qual a Imagem 6a mostra a tela para selecionar o grupo que será avaliado, a Imagem 6b mostra a tela que referente a primeira pergunta da avaliação do aluno selecionado como construtor, e na Imagem 6c é a tela que representa a seleção do aluno que será avaliado como organizador na segunda etapa do processo de avaliação.

Figura 6 – Principais telas do processo de avaliação com dados cadastrados

(a) Tela para selecionar o grupo avaliado



(b) Tela para selecionar o aluno que será avaliado como construtor



(c) Tela para selecionar o aluno que será avaliado como organizador

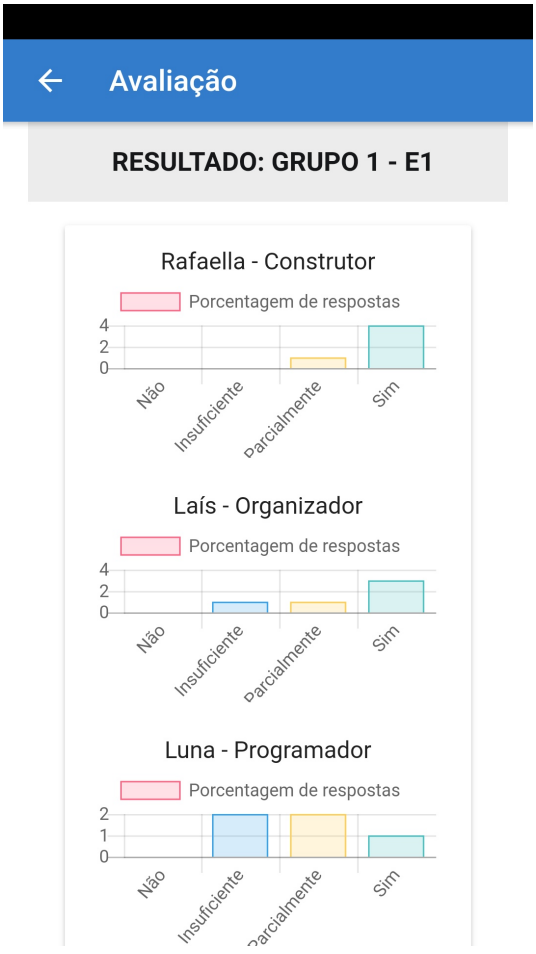
Fonte: Elaborada pelo autor.

No caso da avaliação por meio de dados pré cadastrados, o usuário deve iniciar o processo selecionando a escola e a turma em que a avaliação será aplicada, e depois selecionar o grupo de alunos. De acordo com a metodologia das aulas de robótica de uma escola de Fortaleza, para atingir o objetivo da aula é essencial o trabalho em equipe. Sendo assim, são atribuídos

papeis para os alunos de cada grupo, que são: construtor, organizador, programador e líder. A cada aula os alunos desempenham um papel diferente no grupo, para que assim o aluno desenvolva as diversas habilidades de cada função.

Após a seleção do grupo, o usuário avança para a primeira etapa da avaliação, no qual deve escolher o aluno que está desempenhando a função de construtor. A avaliação é composta por quatro etapas, na qual cada etapa corresponde a uma função. Ao final da avaliação são gerados gráficos que exibem de forma quantitativa o desempenho dos alunos naquela avaliação. Na Figura 7 é apresentada a tela do gráfico que é gerado pelo aplicativo após finalizar a avaliação.

Figura 7 – Tela do gráfico gerado após finalizar a avaliação



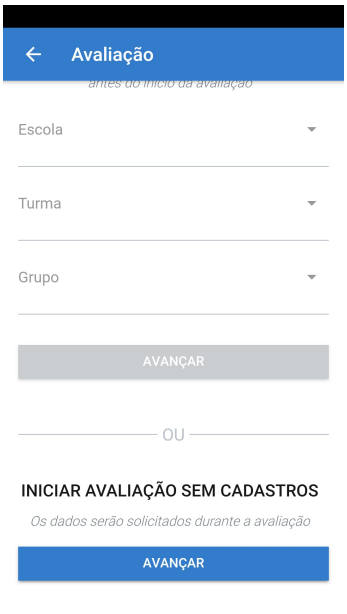
Fonte: Elaborada pelo autor.

4.2.3.2 Avaliação rápida

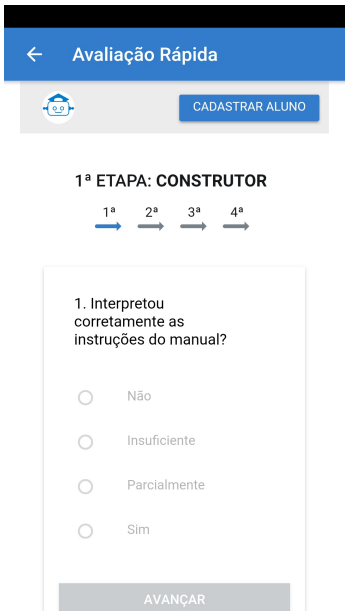
Já na avaliação rápida, o usuário não precisa, obrigatoriamente, já ter dados, como escola, turma e grupo, já pré cadastrados. Um grupo é criado automaticamente ao realizar a

avaliação e as informações de escola e turma podem ser inseridas posteriormente. Ao longo de cada etapa o usuário realiza o cadastro do aluno que está sendo avaliado. Na Figura 8 são apresentadas as principais telas referentes ao processo de avaliação rápida, na qual a Imagem 8a mostra a tela para iniciar a avaliação rápida, a Imagem 8b mostra a tela que referente a primeira pergunta da avaliação rápida, e na Imagem 8c é a tela que representa o cadastro rápido do aluno. Já na Figura 9 é apresentada a tela do gráfico que é gerado pelo aplicativo após finalizar a avaliação.

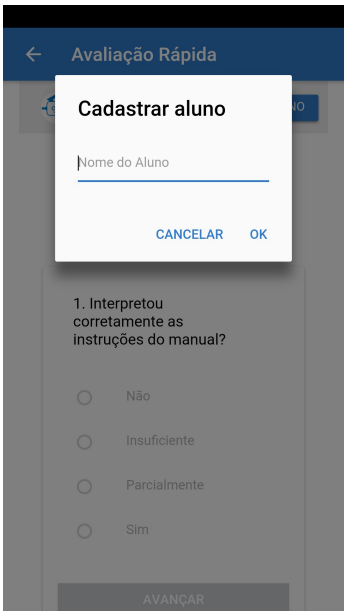
Figura 8 – Principais telas do processo de avaliação rápida



(a) Tela inicial da avaliação rápida



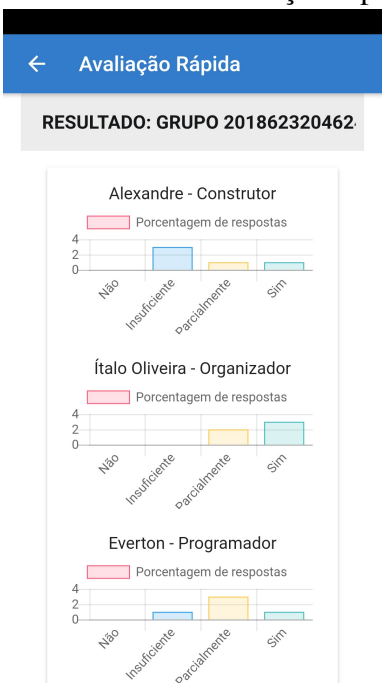
(b) Tela inicial da primeira etapa da avaliação



(c) Tela de cadastro rápido do aluno

Fonte: Elaborada pelo autor.

Figura 9 – Tela do gráfico gerado após finalizar a avaliação rápida

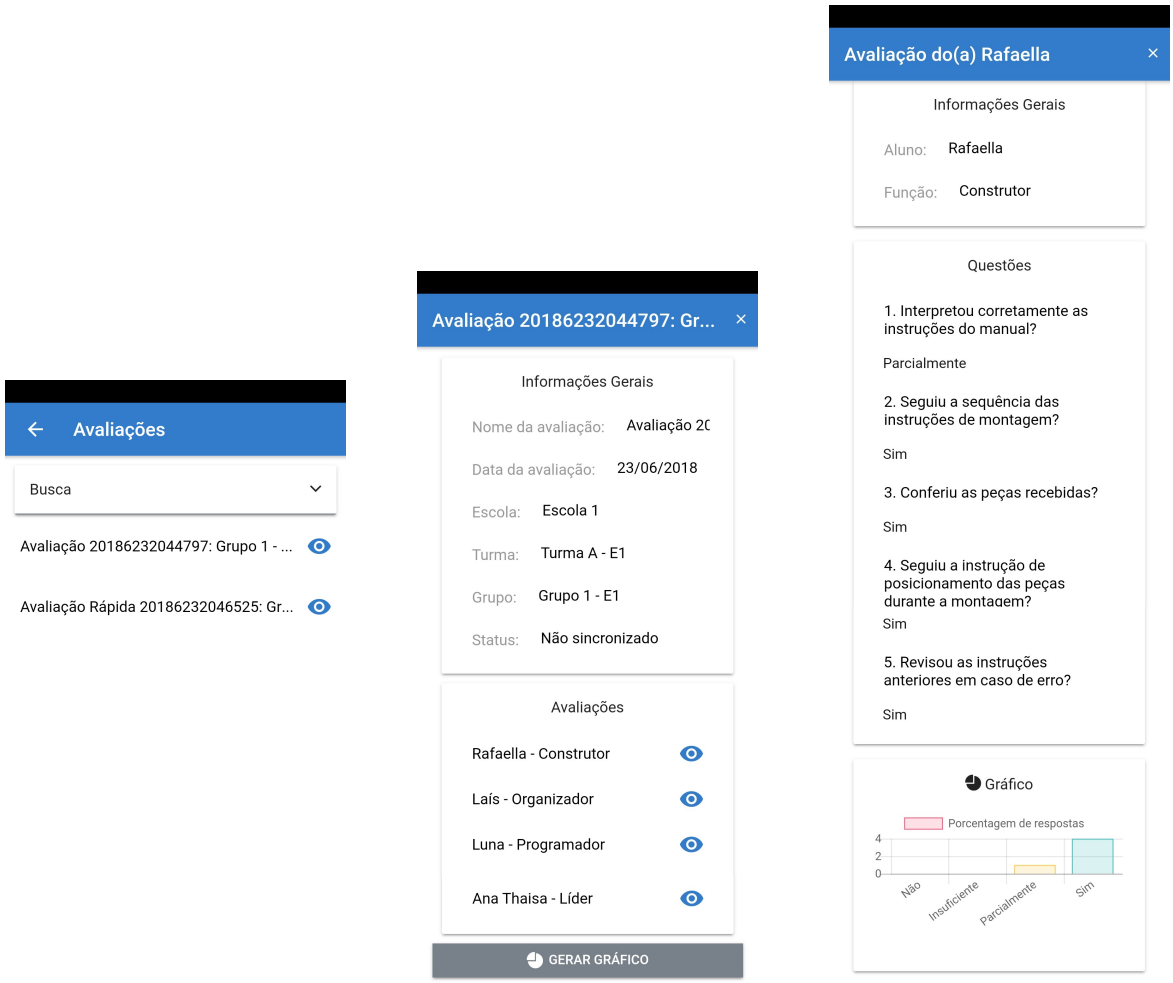


Fonte: Elaborada pelo autor.

4.2.4 Consulta das avaliações realizadas

O usuário tem a opção de consultar, posteriormente, os dados das avaliações realizadas. Nessa tela é possível visualizar informações gerais, como por exemplo, os membros do grupo e quais suas respectivas funções naquela avaliação, nome da turma e escola a qual pertencem, e gerar o gráfico geral do desempenho dos alunos do grupo. Também é possível visualizar verificar as perguntas e respostas da avaliação de cada aluno. Na Figura 10 são apresentadas as imagens referentes a visualização das avaliações que foram realizadas.

Figura 10 – Principais telas da visualização das avaliações realizadas



(a) Tela de visualização das avaliações realizadas

(b) Detalhes da avaliação selecionada

(c) Detalhes da avaliação do aluno selecionado

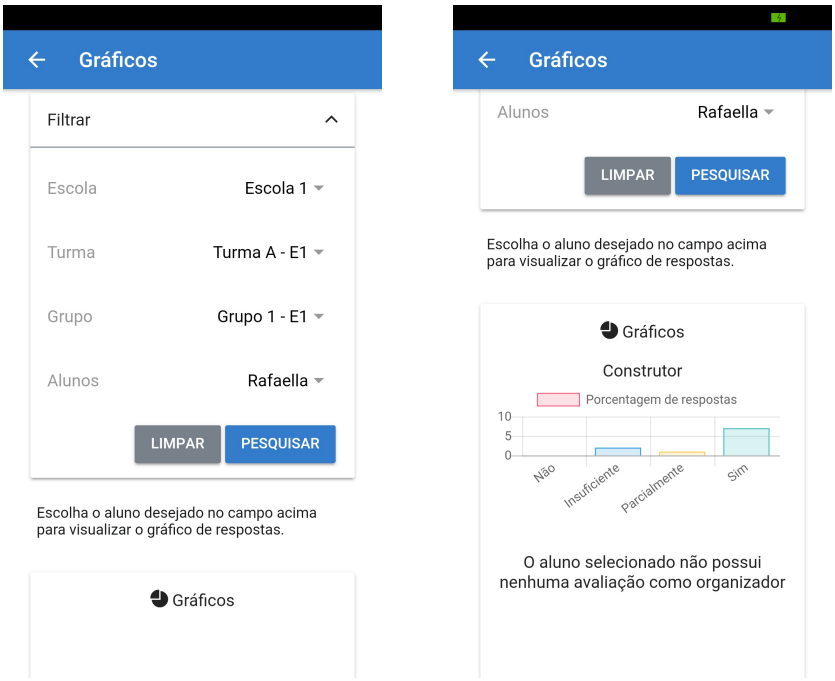
Fonte: Elaborada pelo autor.

4.2.5 Gerar gráficos

Além dos gráficos gerados ao final de cada avaliação e na tela de consulta das avaliações realizadas, o aplicativo também possui uma tela que permite gerar gráficos de desempenho por aluno. Ao pesquisar pelo aluno desejado, são gerados gráficos que mostram o desempenho daquele aluno em todas as funções nas quais ele foi avaliado.

Na Figura 11, o aluno em questão já foi avaliado em todos os papeis da equipe, sendo que o papel de construtor ele já desempenhou duas vezes. Isso é perceptível por meio do somatório dos valores do gráfico

Figura 11 – Telas do gráfico por aluno



(a) Busca por aluno para geração dos gráficos

(b) Gráfico do aluno

Fonte: Elaborada pelo autor.

Para o desenvolvimento dessa funcionalidade foi utilizada a biblioteca *chart.js*[3], que pode ser adicionada ao projeto por meio do seguinte comando:

npm install chart.js –save

[3] chart.js. Disponível em <https://www.chartjs.org/>. Acesso em: 20 junho de 2018

5 IMPLEMENTAÇÃO DO APLICAÇÃO WEB

A codificação da aplicação web do SARE foi feita utilizando o *framework ASP.NET MVC* na versão 5.2.3. Esta tecnologia faz uso de um padrão do padrão *Model–View–Controller* (MCV), implementado na forma de um *framework* pela *Microsoft*. Este padrão visa criar um código que utiliza a abordagem *loosely coupled*, que quer dizer que os componentes do sistema devem ter o mínimo de dependência possível uns com outros (GASPAROTTO, 2017). A utilização dessa abordagem auxilia o desenvolvimento do código pois facilita muito a manutenção e adição de funcionalidades ao código. Semelhante ao padrão *MVVM*, o padrão *MVC*, há três elementos principais: o *model*, que representa as entidades da lógica de negócios da aplicação; a *view*, responsável por apresentar uma interface para o usuário; e o *controller*, que realiza o controle dos outros elementos, fornecendo uma ligação entre eles.

Outra característica importante no desenvolvimento utilizando o *ASP.NET MVC* é porque esse *framework* utiliza *view engines*, ou motores de *view*. Essa utilização retira do *ASP.NET* a responsabilidade pelo controle das *views*. No *MVC 5*, esse motor é o *Razor*, que traz a possibilidade de inserir a lógica da aplicação diretamente na camada de visualização do projeto, por exemplo, é possível inserir a *sintax razor* junto com os códigos *HTML* dentro da mesma página. A sintaxe do *ASP.NET Razor* é extremamente simplificada, baseado na linguagem *C# .NET*.

Código-fonte 3 – GrupoLista.cshtml: Exemplo de *HTML* utilizando o *Razor*.

```
1 [...]
2 @if (Model.Grupos.Count() == 0) {
3     <tr>
4      <td colspan="4"><label>Nenhum registro encontrado</
          label></td>
5     </tr>
6 }
7 [...]
```

5.1 Modelagem do banco de dados

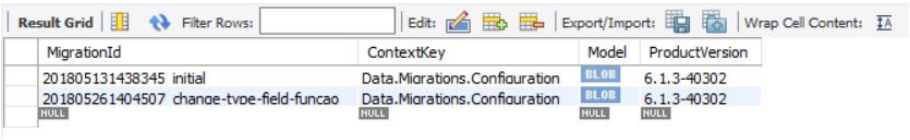
A aplicação desenvolvida utiliza como base da dados o *MySQL*, que é um sistema gerenciador de banco de dados relacional, que utiliza a linguagem *Structured Query Linguage* (SQL), que é a linguagem mais popular para inserir, acessar e gerenciar o conteúdo armazenado num banco de dados (PISA, 2014). Para realizar a persistência dos dados é utilizado o *Entity Framework*, que permite um mapeamento dos elementos de uma base de dados para os elementos de uma aplicação orientada a objetos. No desenvolvimento desta aplicação foi utilizada uma das três metodologias do *Entity Framework* (EF), o *Code First*. Essa abordagem permite que primeiro sejam descritas as classes de entidade antes de criar a base de dados, deixando assim que a responsabilidade de criar a base de dados fique para o *EF* (SATO, 2017).

Código-fonte 4 – TurmaModel.cs: Exemplo de classe de entidade.

```
1  namespace Domain.Models
2  {
3      [Table("Turma")]
4      public class TurmaModel
5      {
6          [Key, DatabaseGenerated(DatabaseGeneratedOption.
               Identity)]
7          public int ID { get; set; }
8          public string Nome { get; set; }
9          public string Status { get; set; }
10         public DateTime LastModifiedDate { get; set; }
11         public int EscolaID { get; set; }
12         [ForeignKey("EscolaID")]
13         public EscolaModel EscolaFK { get; set; }
14         public int UserID { get; set; }
15         [ForeignKey("UserID")]
16         public UsuarioModel UserFK { get; set; }
17     }
18 }
```

Além das tabelas mapeadas nas classes de entidades, também é criada uma tabela chamada *migrationhistory*, que serve para o controle de versão da base de dados, um recurso do *EF* chamado *Migrations* (Figura 12).

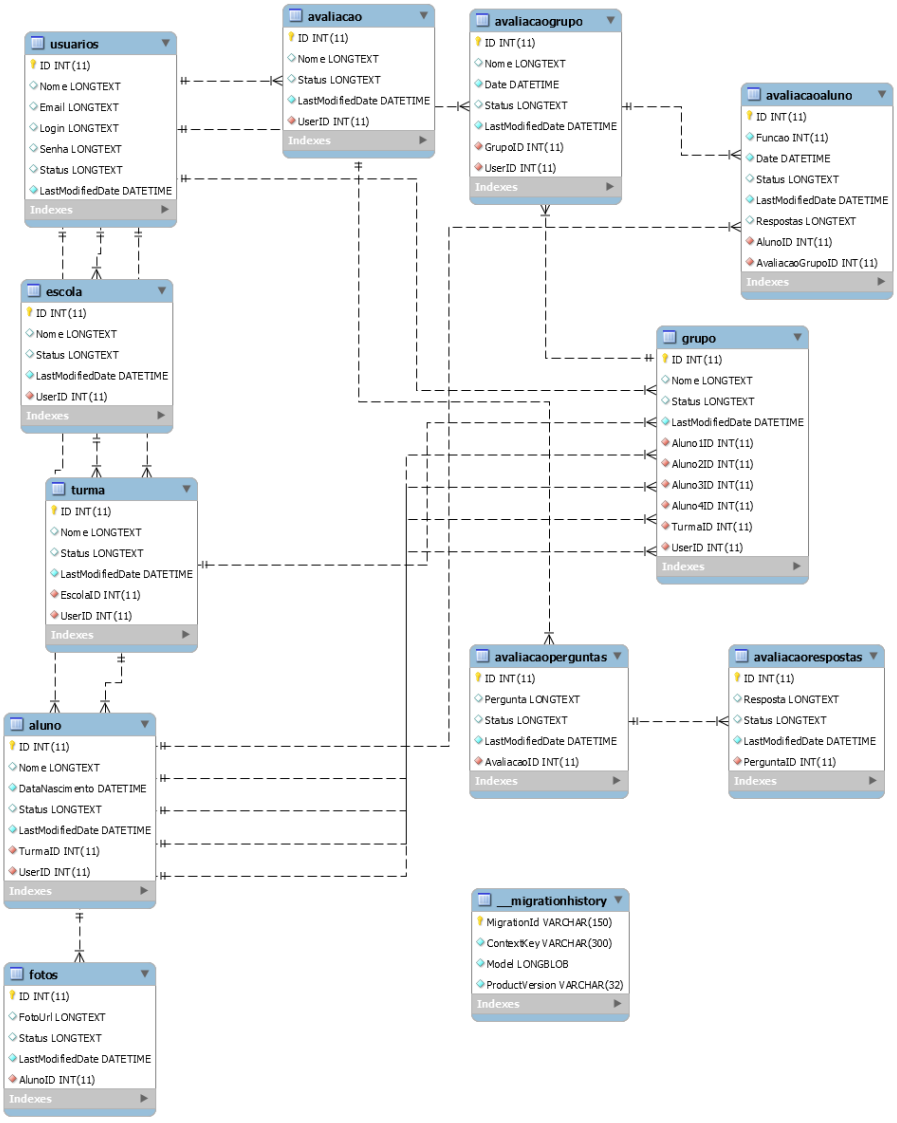
Figura 12 – Tabela *migrationhistory*



Fonte: Elaborada pelo autor.

A Figura 13 representa a modelagem utilizada para a base de dados utilizada na aplicação *web* desenvolvida.

Figura 13 – Modelagem do bando de dados da aplicação *web*



Fonte: Elaborada pelo autor.

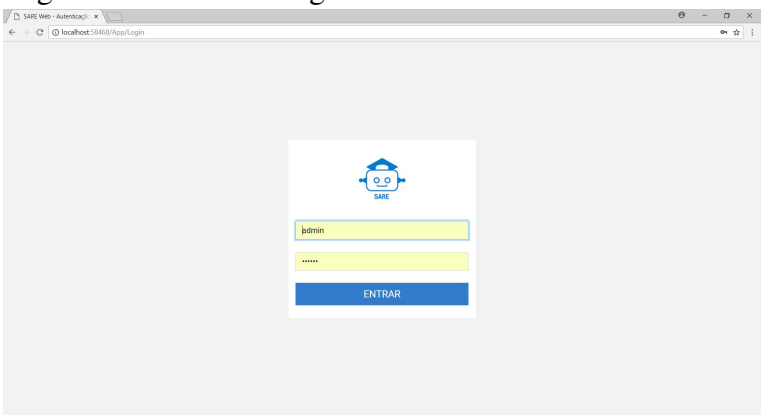
5.2 Aspectos Funcionais

Assim como o aplicativo, a aplicação *web* possui as seguintes funcionalidades: login de usuários, cadastros básicos para o gerenciamento das informações (cadastro de usuários, escolas, turmas, alunos e grupos), realização dos cadastros por meio da importação de arquivos CSV e visualização das avaliações realizadas.

5.2.1 Login de Usuários

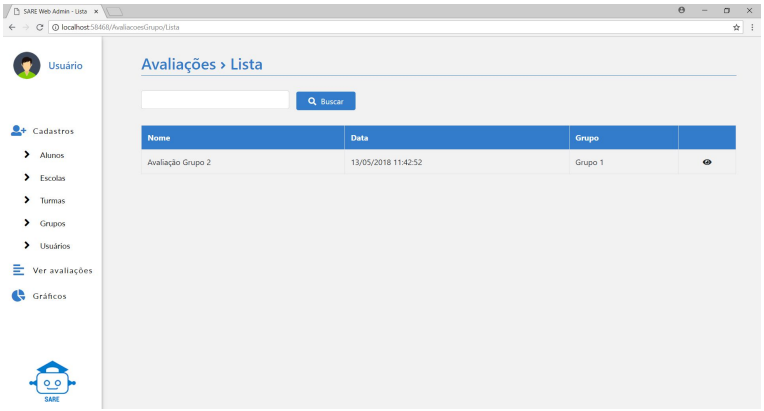
Na tela inicial é possível realizar o acesso à aplicação informando os dados de um usuário já cadastrado. É importante enfatizar que para isso o usuário precisa já ter feito a sincronização de dados por meio do aplicativo. Sobre isso será explicado com mais detalhes no capitulo 6 deste trabalho. Na Figura 14 é apresentada a tela de login da aplicação *web* e na Figura 15 é apresentada a tela de visualização das avaliações realizadas.

Figura 14 – Tela de Login



Fonte: Elaborada pelo autor.

Figura 15 – Página Inicial

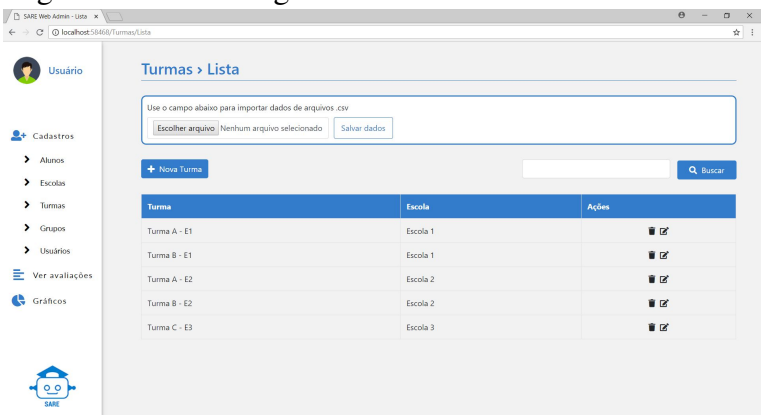


Fonte: Elaborada pelo autor.

5.2.2 Cadastros básicos e importação de arquivos CSV

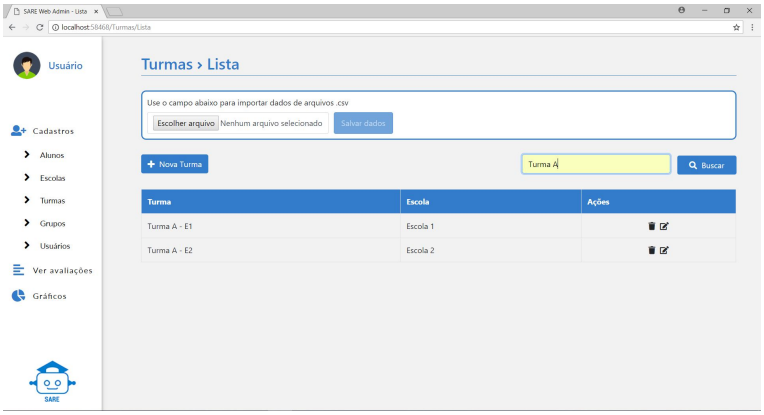
Após acessar a aplicação, o usuário pode realizar os cadastros das informações serão utilizadas durante o processo da avaliação no aplicativo após a sincronização dos dados. Nessas telas de gerenciamento é possível realizar a inserção de dados de forma manual e por meio da importação de arquivos CSV. Na aplicação é possível realizar o cadastro de escolas, turmas, alunos e grupos. A Figura 16 mostra a tela de gerenciamento das turmas e na Figura 17 mostra o resultado da realização da busca na tela de gerenciamento das turmas.

Figura 16 – Tela de gerenciamento das turmas



Fonte: Elaborada pelo autor.

Figura 17 – Resultado da busca



Fonte: Elaborada pelo autor.

Código-fonte 5 – TurmasController.cs: Exemplo do código utilizado para importação de dados.

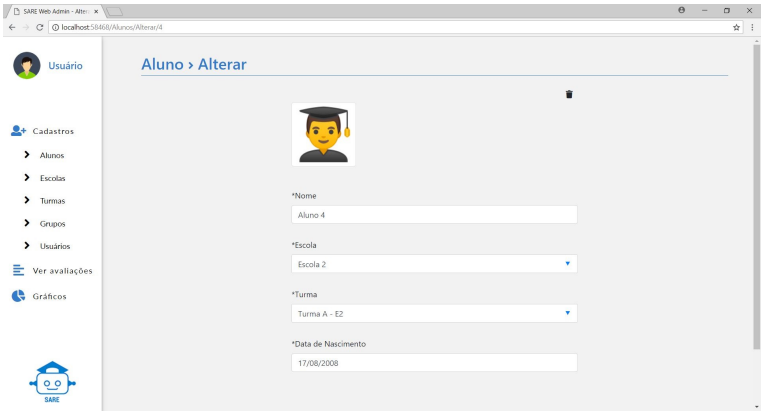
```
using (var reader = new StreamReader(pathReader)) {
    List<TurmaModel> listaTurmas = new List<TurmaModel>();

    while (!reader.EndOfStream) {
        var line = reader.ReadLine();

        if (!line.Contains("Cadastros de turmas") && !line.
            Contains("Nome;Escola")) {
            var item = new TurmaModel();
            var values = line.Split(";");
            item.Nome = values[0];
            item.EscolaID = this.escolaBusiness.GetByNome(
                values[1]).ID;
            item.LastModifiedDate = DateTime.Now;
            item.Status = "ADDED";
            item.UserID = SessionUtil.UserLogged.ID;

            listaTurmas.Add(item);
        }
    }

    this.turmaBusiness.AddList(listaTurmas);
}
```

Outra funcionalidade da aplicação *web* é o *upload* de fotos no cadastro dos aluno. O Código-fonte 6 mostra o código utilizado para a realização do upload de imagens no cadastro de alunos. Na Figura 18 é apresentada a tela de edição do cadastro de um aluno.

Código-fonte 6 – AlunosController.cs: Exemplo do código utilizado para o *upload* de imagens.

```
var path = System.Web.HttpContext.Current.Server.MapPath("
   \\_uploads");
var file = Request.Files[i];
if (!Directory.Exists(path)) {
    Directory.CreateDirectory(path);
}
string filename = string.Format("{0}_{1}.{2}", DateTime.Now
   .ToString("dd-MM-yyyy"), nome, file.FileName.Split(".")
   [1]);
string fullname=string.Format("{0}\\{1}", path, filename);
file.SaveAs(fullname);
Bitmap img = new Bitmap(file.InputStream);
int w = img.Width;
int h = img.Height;
if (w > h) {
    h = (h * 120 / w);
    w = 120;
} else {
    w = (w * 120 / h);
    h = 120;
}
Image thumb = img.GetThumbnailImage(w, h, () => false,
   IntPtr.Zero);
thumb.Save(fullname.Insert(fullname.LastIndexOf("."), "
   _thumb"));
FotoModel foto = new FotoModel();
foto.FotoUrl = filename;
foto.Status = "ADDED";
foto.LastModifiedDate = DateTime.Now;
foto.AlunoID = id;
this.fotoBusiness.Add(foto);
```

Figura 18 – Tela de edição do aluno

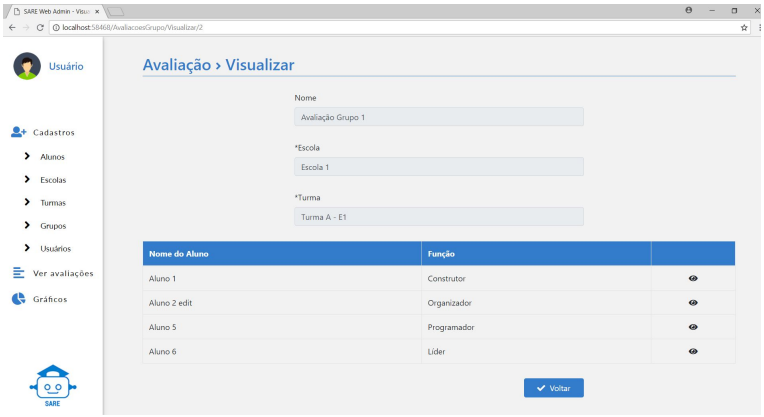


Fonte: Elaborada pelo autor.

5.2.3 Consulta das avaliações realizadas

O usuário tem a opção de consultar os dados das avaliações realizadas, após a realização da sincronização dos dados do aplicativo. Assim como no aplicativo, nesta tela é possível visualizar informações gerais, como por exemplo, os membros do grupo e quais suas respectivas funções naquela avaliação, nome da turma e escola a qual pertencem.

Figura 19 – Tela de visualização da avaliação



Fonte: Elaborada pelo autor.

6 INTEGRAÇÃO DO APLICATIVO E DA APLICAÇÃO *WEB*

A integração entre os dados do aplicativo e da aplicação *web* foi feita em duas partes principais. A primeira parte é realizada no aplicativo, no qual, antes de iniciar o processo de sincronização é feita uma validação para garantir que o dispositivo está conectado à *internet*. Essa verificação é realizada por meio do *plugin cordova-plugin-network-information*. Para adiciona-lo ao projeto é necessário executar os seguintes comandos:

ionic cordova plugin add cordova-plugin-network-information

npm install –save @ionic-native/network

Caso o dispositivo esteja conectado à *internet*, então são feitas chamadas HTTP para enviar os JSON com as informações que deverão ser inseridas ou atualizadas no banco de dados do servidor. Para realizar as chamadas HTTP são utilizados os módulos "HttpClient"e "HttpClientModule"que pertencem ao pacote *common* do Angular.

A segunda parte é feito do lado da aplicação web, que inicialmente foi inserida uma configuração para definir as rotas das URL's que serão chamadas. O Código-fonte 7 mostra o código utilizado para definir a rota da *WepAPI*.

Código-fonte 7 – RouteConfig.ts: Código para definir rota da WebAPI.

```
routes.MapRoute(
    name: "Default",
    url: "{controller}/{action}/{id}",
    defaults: new { controller = "App", action = "Index",
       id = UrlParameter.Optional }
);
```

A aplicação possui um *controller* responsável por realizar a sincronização de dados. Após receber o JSON, a aplicação cria uma lista de objetos de cada uma das classes de entidades, de acordo com o status de cada registro. Por exemplo, os registro com o *status* igual 1 são registros que devem ser adicionados ao banco. Ao termino do processo, são enviados como resposta a lista de registros que sofreram alteração ou foram adicionados, para dessa forma, manter a base de dados do aplicativo atualizada.

7 AVALIAÇÃO

7.1 Análise de desempenho da primeira versão do aplicativo

7.1.1 Experimento

Em 2017, durante a disciplina de Análise de Desempenho do curso de Sistemas e Mídias Digitais da UFC, foi realizada uma análise com a primeira versão do aplicativo que foi desenvolvida no mesmo ano. Essa analise teve como objetivo validar o uso do aplicativo em sala de aula.

7.1.1.1 Métricas

a) Tempo de realização de cada avaliação.
   Atualmente os professores realizam a avaliação dos alunos por meio de uma folha de papel, na qual é anotado o desempenho de cada aluno, e posteriormente ele revisa o que escreveu e assim define uma nota. Segundo eles, esse processo acaba sendo muito demorado considerando que cada turma tem em média 20 alunos. Sendo assim, essa métrica serve para avaliar se realmente a utilização do aplicativo para avaliar os alunos é mais rápido e vantajoso comparado com a forma que é realizada atualmente.
b) O esforço do professor para fazer a avaliação.
   Como dito no item anterior, os professores consideram o processo de avaliação atual muito lento e trabalhoso, e o aplicativo visa diminuir o esforço nesse processo. Um exemplo disso é que ao final da avaliação o professor já vê gráficos sobre o desempenho dos alunos. Desta forma essa métrica serve para verificar se o esforço de usar o aplicativo é realmente menor, tornando o processo de avaliação mais prático.

7.1.1.2 Parâmetros

Quando este experimento foi realizado o aplicativo ainda não tinha conexão com uma base de dados online, e os dados eram salvos somente no dispositivo do professor. Esse fato gerava uma restrição pois o aparelho não poderia ser formatado ou o aplicativo não poderia ser desinstalado pelo risco de perder as informações contidas nele. Outro fator que se tornou

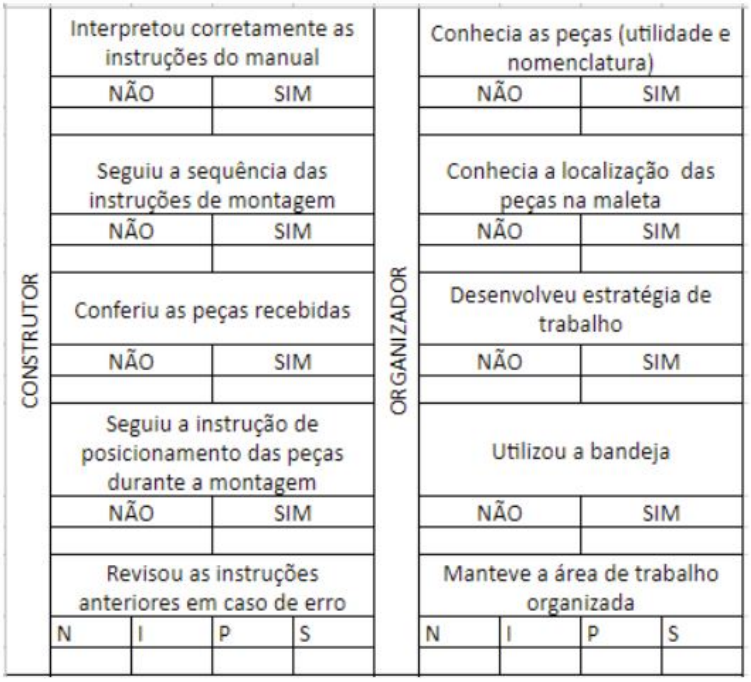
uma restrição neste experimento foi uma maior dificuldade de buscar professores/escolas que se disponibilizam a realizar esse experimento.

7.1.1.3 Fatores e níveis

a) Perfil de cada aluno.

Durante a aula, a turma é dividida em grupos com quatro alunos, onde cada aluno de cada grupo desempenha uma função diferente na atividade. As funções são:
Níveis: Líder, Organizador, Programador e Construtor.
Nas imagens abaixo pode-se perceber que os parâmetros para avaliação de cada perfil é diferente, devido a isso esse fator é importante pois os alunos podem ter desempenhos diferentes em cada tipo de função e é interessante para o professor identificar essas diferenças. As Figuras 21 e 22 mostram as perguntas que são utilizadas nas avaliações dos alunos, e esse material foi disponibilizado por uma professora de robótica de Fortaleza.

Figura 20 – Perguntas da avaliação do aluno construtor e organizador



Fonte: Elaborada pelo autor.

Figura 21 – Perguntas da avaliação do aluno programador e líder

Fonte: Elaborada pelo autor.

b) Turma.

   Na escola onde o experimento foi aplicado, as aulas de robóticas são fornecidas para as turmas do quarto ao nono do ensino fundamental, porém foram selecionadas somente algumas delas para a realização do experimento:

   Níveis: quarto ano, sexto ano, sétimo ano e oitavo ano

c) Tipo de processo avaliativo.

   Esse fator serve para representar o tipo de processo avaliativo usado no experimento. No caso, se a avaliação será feita pelo modelo tradicional ou pelo aplicativo.

   Níveis: Modelo tradicional e usando o aplicativo

7.1.2 Metodologia

No primeiro momento do experimento, foi feita a medição do tempo que leva para realizar cada avaliação em cada uma das turmas, considerando o modelo de avaliação que atualmente é utilizado pelos professores. Em um segundo momento, os professores avaliaram seus alunos utilizando o aplicativo e o tempo que eles leva para isso também foi cronometrado. Depois da realização dessas avaliações os professores responderam um formulário no qual foram

feitas perguntas sobre usabilidade e dificuldades que eles sentiram ao utilizar o aplicativo.

O experimento foi realizado nos dias 23/11/2017 e 28/11/2017, em uma escola particular de Fortaleza. Esses dias foram definidos de acordo com a disponibilidade dos professores que participaram do experimento. No primeiro dia, as avaliações foram realizadas utilizando o modelo convencional dos professores, para cronometrar o tempo gasto na avaliação de cada aluno durante a aula. Nesse mesmo dia, foi definido com os professores que depois das aulas, eles deveriam terminar todo o processo de avaliação que eles já realizam, que consiste em passar as respostas de cada avaliação para uma planilha, depois definir e lançar as notas dos alunos. No segundo dia de experimento, o objetivo foi cronometrar o tempo gasto para realizar o mesmo processo de avaliação utilizando o aplicativo durante e após a aula.

É válido ressalvar tanto no primeiro quanto no segundo dia cada professor avaliou a mesma turma. Exemplo: o Professor A avaliou a turma B no primeiro e no segundo dia de experimento. Nas tabelas da seção 7.1.3 Análise e apresentação dos resultados, estão os resultados dos tempos de avaliação de cada turma. Não serão identificados os nomes dos professores e alunos. Depois das avaliações do segundo dia de experimento os professores responderam um formulário sobre o uso do aplicativo em sala de aula. O objetivo desse formulário é identificar onde o aplicativo ainda está falhando, possíveis dificuldades e melhorias que devem ser feitas.

7.1.3 Análise e apresentação dos resultados

O Figura 23 representa os resultados da primeira turma avaliada é possível perceber uma irregularidade maior em comparação com os outros gráficos. No modelo tradicional é comum acontecer isso, pois existem alunos que têm mais dificuldades que outros, então demandam mais atenção, dessa forma a avaliação acaba se tornando mais demorada. Nessa mesma turma, a avaliação por meio do aplicativo também foi um pouco demorada, pois o professor ainda estava se adaptando ao uso do aplicativo e ele ainda não tinha nenhuma experiência prévia com ele.

O professor B já tinha testado as primeiras versões do aplicativo então ele conseguiu usá-lo sem muitas dificuldades e vemos que a linha referente ao uso do aplicativo ficou bem mais linear, mostrando assim esse domínio do professor. Assim como o professor B, o professor C também já tinha testado o aplicativo informalmente, e também obtive bons resultados de tempo. Outro fator que pode influenciar nessa irregularidade entre os tempos realizados em cada avaliação, é a turma, por exemplo a turma do quarto ano, tem menos experiência nas aulas de

Figura 22 – Gráfico com os resultados da turma do quarto ano

Fonte: Elaborada pelo autor.

Figura 23 – Gráfico com os resultados da turma do quinto ano

Fonte: Elaborada pelo autor.

Figura 24 – Gráfico com os resultados da turma do sétimo ano

Fonte: Elaborada pelo autor.

robótica, e muitas vezes demandam mais atenção do professor.

7.2 Análise de desempenho da aplicação *web*

A segunda avaliação proposta neste trabalho tem como objetivo verificar alguns aspectos de desempenho na comunicação entre o cliente e o servidor. Neste sentido, mediu-se, utilizando algumas métricas de desempenho, o comportamento do módulo de *web services* do servidor diante de algumas situações de acessos simultâneos aos *web services* no servidor.

A ferramenta *JMeter4*[1] foi utilizada para criar, gerar e simular as cargas de trabalho que serão enviadas ao módulo de *web services* do servidor. O *JMeter* é uma aplicação escrita em Java que foi desenvolvida para projetar e capturar resultados de experimentos orientados a métricas de desempenho (Ramakrishnan et al. 2017). Nos experimentos realizados neste trabalho, o *JMeter* atua simulando os diversos usuários, em seus dispositivos móveis, solicitando processamento no lado servidor.

A aplicação web foi implementado e hospedado em um servidor particular. A *Uniform Resource Locator* (URL) do serviço foi configurada no *JMeter* e um conjunto de cargas de trabalho imediatas foram especificadas com o objetivo de observar o comportamento do modulo de *web services* diante do aumento do numero de solicitaçoes simultaneas ao serviço de sincronização de dados. A Tabela 1 mostra a configuração experimental que foi projetada para a analise de desempenho do módulo de *web services* do servidor.

Configuração Experimental	Valor
Protocolo de Comunicação para a URL	HTTPS
Método de Comunicação com a URL	POST
Formato da Resposta da URL	text/plain
Tamanho da Resposta da URL	5,92 KB (6.063 bytes)
Série dos Números de Clientes Simultâneos	[10, 50, 100, 250, 500, 1000, 2000]
Métricas Avaliadas	Tempo Médio de Conexão

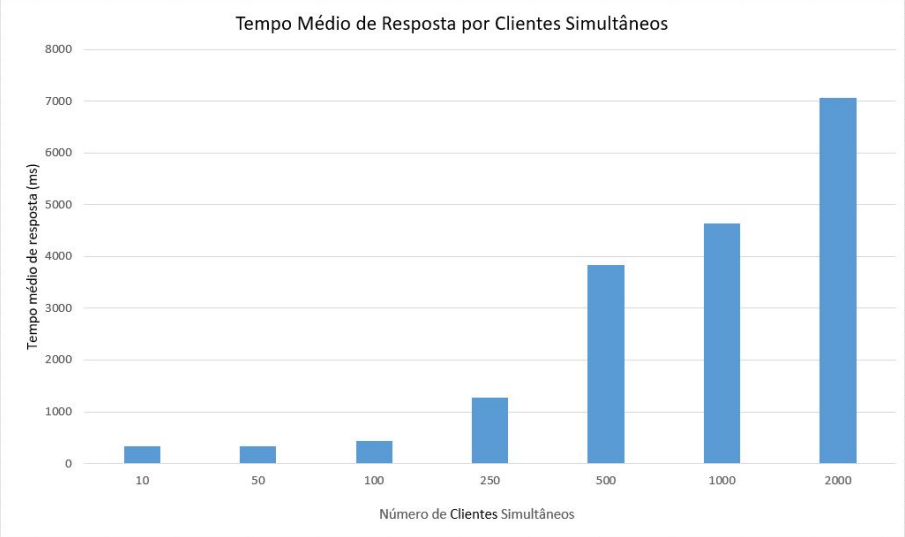
Tabela 1 – Configurações Experimentais e seus Valores

A primeira análise do experimento observou o comportamento do módulo de web services do servidor em relação ao aumento do número de clientes simultâneos e o respectivo tempo de resposta médio. Dessa forma, foi realizada uma média do tempo de resposta de todos os clientes, ou seja, o tempo médio de resposta levando em consideração o número de clientes

[1] Apache JMeter. Disponível em <http://jmeter.apache.org/>. Acesso em: 21 junho de 2018

simultâneos. A Figura 25 mostra os resultados obtidos para a métrica de tempo de resposta médio por clientes simultâneos.

Figura 25 – Tempo Médio de Resposta por Clientes Simultâneos



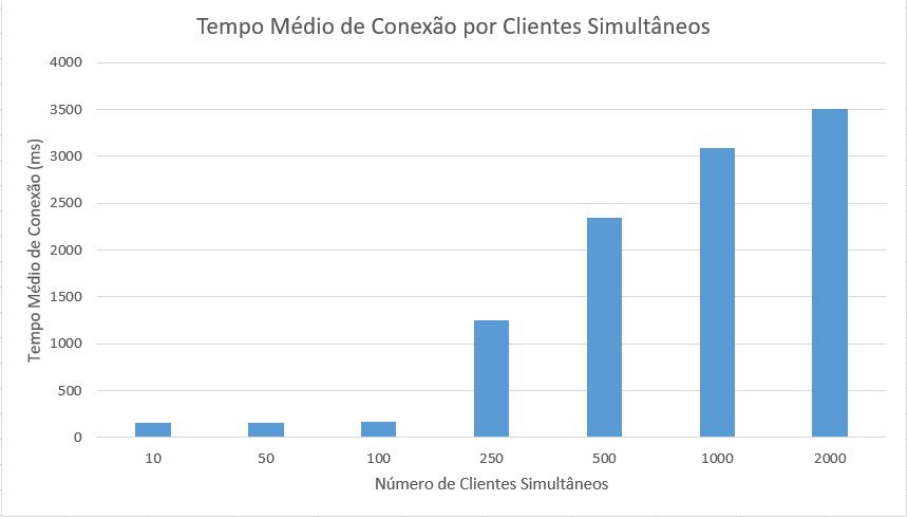
Fonte: Elaborada pelo autor.

Observando a Figura 25, pode-se perceber que houve um aumento no tempo de resposta médio diante do crescimento do número de clientes simultâneos. Entre 10 e 250 clientes simultâneos, a diferença no tempo médio foi pouca, no entanto, entre 250 e 2000 clientes simultâneos, houve uma diferença bem maior no tempo médio de resposta.

Outra análise realizada foi em relação ao tempo médio gasto por conexão ao servidor, à medida que o numero de clientes simultâneos vai aumentando. Este tempo de médio de conexão está relacionado ao tempo gasto, em milissegundos, para estabelecer uma conexão HTTPS com o servidor.

Pode-se perceber, na Figura 26, um comportamento semelhante ao que foi visualizado na Figura 25. Esse comportamento mostra que o servidor, onde esta implantado o serviço utilizado, é sobrecarregado quando o número de clientes ultrapassa 250. Uma alternativa inicial seria analisar a aplicação e identificar pontos de otimização, e no caso de isso não ser suficiente, outra alternativa seria adotar as técnicas de replicação do serviço em outros servidores e um balanceamento de carga.

Figura 26 – Tempo Médio de Conexão por Clientes Simultâneos



Fonte: Elaborada pelo autor.

8 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho mostrou o desenvolvimento de duas aplicações, um aplicativo *mobile* e uma aplicação *web*, que visa auxiliar no processo de avaliação dos alunos durante as aulas práticas de robótica educacional. Inicialmente foi definido um conjunto de funcionalidades que o aplicativo deveria ter. Após o desenvolvimento da primeira versão do aplicativo foi realizada uma avaliação com os professores de uma escola. Com os resultados dessa avaliação foram definidas outras funcionalidades que deveriam ser desenvolvidas e a necessidade do desenvolvimento de uma aplicação *web*. A implementação de ambas as aplicações foi descrita ao longo deste trabalho e o código-fonte está disponibilizado em repositórios no *Github* [1]. Por fim, uma avaliação foi realizada para observar o comportamento da arquitetura de sistemas e implementação de acordo com o crescimento de usuários simultâneos.

Como trabalhos futuros, pretende-se (1) entrar em contato com algumas escolas de Fortaleza para realizar testes de usabilidade tanto da aplicação web quanto do aplicativo em sala de aula; (2) gerar outros tipo de gráficos no aplicativo como por exemplo, gráfico dos resultados dos alunos por função e por turma; (3) concluir funcionalidade na aplicação web que permita o professor customizar uma avaliação, ou seja, em qual ele possa definir as perguntas e respostas que mais se adéquam ao seu modelo de avaliação já utilizado. Essa funcionalidade foi iniciado durante o primeiro semestre de 2018, toda a estrutura de *backend* já foi implementada, porém não foi possível conclui-lá em tempo hábil para entrar no grupo de funcionalidades descritas neste trabalho. Além disso pretende-se também (4) implementar gráficos que possuam as mesmas regras de negocio existentes no aplicativo; (5) implementar o envio de relatório por *e-mail* para o professor, (6) algumas melhorias de usabilidade em ambas as aplicações e (7) a investigação de possíveis melhorias na aplicação *web* para a otimização do tempo de resposta e do tempo médio de conexão.

[1] Repositório do aplicativo. Disponível em <https://github.com/HaretaAlves/smd-projetointegrado2-ionic>. Repositório da aplicação web. Disponível em <https://github.com/HaretaAlves/sare-web>. Acesso em: 08 junho de 2018

REFERÊNCIAS

ALMEIDA, M. E. **The texbook**. Brasília: Editora Parma, 2000. v. 1. 15 p.

GASPAROTTO, H. M. **ASP.NET MVC: Desenvolvendo uma aplicação na prática**. 2017. Disponível em: <https://www.devmedia.com.br/asp-net-mvc-desenvolvendo-uma-aplicacao-na-pratica/31529>. Acesso em: 20 jun. 2018.

GONÇALVES, A. J. R. Desenvolvimento de aplicativos híbridos com o ionic framework. III Escola Regional de Informática do Piauí, 2017.

LÉVY, P. **As tecnologias da inteligência: o futuro do pensamento na era da informática**. São Paulo: Ed. 34, 1993. v. 1. 15 p.

PAPERT, S. **A máquina das crianças: repensando a escola na era da informática**. Porto Alegre: [s.n.], 1994.

PERRENOUD, P. **10 novas competências para Ensinar**. Porto Alegre: Artmed, 2000.

PISA, P. **O que é e como usar o MySQL?** 2014. Disponível em: <http://www.techtudo.com.br/artigos/noticia/2012/04/o-que-e-e-como-usar-o-mysql.htm>. Acesso em: 20 jun. 2018.

SANTOS, T. F. M.; ROLIM, C. O. A utilização de dispositivos móveis como ferramenta pedagógica colaborativa na educação infantil. II Simpósio de Tecnologia da Informação da Região Noroeste do Rio Grande do Sul, 2011, Santo Angelo, 2011.

SATO, P. M. **Entity Framework Tutorial**. 2017. Disponível em: <https://www.devmedia.com.br/entity-framework-tutorial/27764>. Acesso em: 20 jun. 2018.

SCHONS, C.; PRIMAZ, E.; WIRTH, G. d. A. P. Introdução a robótica educativa na instituição escolar para alunos do ensino fundamental da disciplina de língua espanhola através das novas tecnologias de aprendizagem. I WORKSHOP DE COMPUTAÇÃO DA REGIÃO SUL, 2004, 1, Florianópolis, UNISUL, 2004.

SILVA, A. F. D. Roboeduc: uma metodologia de aprendizado com robótica educacional. Tese (Doutorado) – Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Norte, Natal, 2009.

SILVA, N. B. d.; YOSHIDA, R. R.; GUERRA, R. G. **Formação profissional baseada em competências: um estudo de caso brasileiro**. 2004. Disponível em: <https://docgo.net/philosophy-of-money.html?utm_source=formacao-profissional-baseada-em-competencias-um-estudo-de-caso-brasileiro>. Acesso em: 15 março. 2018.

TAJRA, S. F. **Informática na Educação Novas Ferramentas Pedagógicas para o Professor da Atualidade**. São Paulo: Érica, 2001.

WAHLBRINCK, K. A. Análise de performance de frameworks para desenvolvimento multiplataforma mobile. Universidade Federal de Santa Maria, 2015.

WAHLBRINCK K. A; BONIATI, B. B. Aplicações mobile híbridas: Um estudo de caso do framework ionic para construção de um diário de classe. Anais do EATI - Encontro Anual de Tecnologia da Informação, 2017.

ZILLI, S. d. R. A robótica educacional no ensino fundamental: Perspectivas e prática. Dissertação (Mestrado). Programa de Pós-Graduação em Engenharia de Produção. Universidade Federal de Santa Catarina, 2004.