UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ESPECIALIZAÇÃO EM DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS

JOÃO GUILHERME DYCK

SISTEMA JONGUI: TECNOLOGIAS INTEGRADAS PARA O ENSINO COLABORATIVO DE MÚSICA

CURITIBA

2018

JOÃO GUILHERME DYCK

SISTEMA JONGUI: TECNOLOGIAS INTEGRADAS PARA O ENSINO COLABORATIVO DE MÚSICA

Trabalho de Conclusão de Curso apresentado ao curso de Pós-Graduação em Desenvolvimento para Dispositivos Móveis, Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, como requisito parcial à obtenção do título de Especialista Desenvolvimento para Dispositivos Móveis.

Orientador: Prof. Dr. Leonelo Dell Anhol Almeida

Curitiba

2018

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Curitiba
Diretoria de Pesquisa e Pós-Graduação
Departamento Acadêmico de Informática
Coordenação do Curso de Especialização em Desenvolvimento para Dispositivos Móveis

TERMO DE APROVAÇÃO

“Sistema Jongui: Tecnologias Integradas para o Ensino Colaborativo de Música”

por

“João Guilherme Dyck”

Este Trabalho de Conclusão de Curso foi apresentado às 17:30 do dia 19 de fevereiro de 2018 na sala B201 como requisito parcial à obtenção do grau de Especialista em Desenvolvimento para Dispositivos Móveis na Universidade Tecnológica Federal do Paraná - UTFPR - Campus Curitiba. O(a) aluno(a) foi arguido pela Banca de Avaliação abaixo assinados. Após deliberação, a Banca de Avaliação considerou o trabalho aprovado.

_____ Prof. Leonelo Dell Anhol Almeida **(Presidente/Orientador – UTFPR/Curitiba)**	_____ Profa. Maria Claudia Figueiredo Pereira Emer **(Avaliador 1 – UTFPR/Curitiba)**
_____ Prof. Robson Ribeiro Linhares **(Avaliador 2 – UTFPR/Curitiba)**	

“A Ata de Aprovação assinada encontra-se na Coordenação do Curso.”

RESUMO

DYCK, João Guilherme. SISTEMA JONGUI: TECNOLOGIAS INTEGRADAS PARA O ENSINO COLABORATIVO DE MÚSICA. Trabalho de Conclusão de Curso - Especialização em Desenvolvimento para Dispositivos Móveis, Universidade Tecnológica Federal do Paraná. Curitiba, 2018

Este trabalho busca solucionar o problema do Ensino Musical em sala de aula utilizando dispositivos móveis. A relevância dessa pesquisa se dá porque com os avanços tecnológicos recentes abriu-se a possibilidade de integrar essas ao ensino musical. Partindo do princípio *Bring Your Own Device* (BYOD) e do desenvolvimento multi-plataforma para dispositivos móveis, integrando esses ao método Orff. Como resultado obteve-se o desenvolvimento do Sistema Jongui para ensino musical. Esse apresentou boa aceitação no teste feito, apesar das melhorias ainda necessárias.

Palavras-chave: Desenvolvimento Multi-Plataforma, *Orff Schulwerk*, *Bring Your Own Device*, Educação Musical.

ABSTRACT

DYCK, João Guilherme. JONGUI SYSTEM: INTEGRATED TECHNOLOGIES FOR COLLABORATIVE MUSIC TEACHING. Final Coursework – Specialization in Mobile Application Development, Universidade Tecnológica Federal do Paraná. Curitiba, 2018

This research tries to solve the problem of Music Education in the classroom employing mobile devices. With the latest technological advances many possibilities opened to integrate these with Mucis Education. Starting with the Bring Your Own Device (BYOD) principle and multi-platform development, integrating them with the Orff methods. As a result the Jongui System was developed. It presented a good acceptance by the test made, even with adjustments that are needed.

Keywords: Multi-platform development, *Orff Schulwerk*, Bring Your Own Device, Musical Education.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

No relatório divulgado pela Agência Nacional de Telecomunicações (ANATEL, 2018), em outubro de 2017 haviam 240.850.681 linhas de telefonia móvel ativas no Brasil. Segundo Meirelles (2017), também em outubro de 2017 havia no Brasil um smartphone por habitante. Portanto, em uma sociedade onde cada vez mais dispositivos móveis marcam presença a educação e a propagação do conhecimento também são por eles influenciados. Já em 2007 Trexler questionava os impactos e as mudanças que uma educação móvel traria aos ambientes de aprendizado. Desde então diversas tentativas foram feitas para integrar os avanços tecnológicos à sala de aula, mas também à vida dos estudantes. Por exemplo, a província canadense de Alberta lançou em 2012 um guia para a implementação do princípio *Bring Your Own Device* para suas escolas[1]. Segundo esse princípio, cada estudante deve trazer seu dispositivo eletrônico para a sala de aula e já existem diversos estudos e políticas empregadas para integração de dispositivos móveis ao ambiente escolar.

Com relação ao ensino de música propriamente dito, optou-se por empregar a metodologia conhecida como *Orff-Schulwerk*. Essa se desenvolveu durante o século XX a partir das ideias pedagógicas do Carl Orff[2]. Esse método parte de simples conceitos musicais ganhando complexidade ao longo do desenvolvimento. Outro ponto interessante para esse trabalho está em que nesse método a aprendizagem pode ocorrer de forma colaborativa.

Porém, para a metodologia *Schulwerk* requer-se um instrumentário muitas vezes difícil de adquirir. Para contornar esse problema, nesse trabalho busca-se desenvolver uma aplicação móvel que permita substituir alguns dos instrumentos Orff (xilofone, metalofone e glockenspiel) por ela. Para tanto, utilizou-se um Raspberry Pi para processar as interações dos estudantes com seus aplicativos.

Referente aos aparelhos nos quais a aplicação móvel executará, o presente trabalho pretende rodar em plataformas Android e iOS. Porém, para evitar a necessidade de se desenvolver uma aplicação específica para cada sistema

---

[1] O referido guia encontra-se no link: https://goo.gl/zivzTH

[2] Carl Orff foi um compositor musical do século XX. Sua grande obra foi Carmina Burana (KUGLER, 2017).

operacional, decidiu-se por utilizar tecnologia híbrida. Por tecnologia híbrida entende-se tecnologias de desenvolvimento de aplicações móveis que permitam o desenvolvimento da aplicação em uma linguagem, mas a compilação distinta para cada sistema operacional móvel. Com isso, atinge-se dois objetivos distintos. Primeiro, basta desenvolver a aplicação apenas uma vez, sem precisar escrever o aplicativo em uma plataforma para depois reescreve-lo em outra. O segundo objetivo atingido configura-se na integração dos estudantes com a mesma aplicação. Dessa forma, evita-se que durante a utilização da aplicação deva-se explicar como estudantes utilizando uma plataforma devem encontrar as atividades e depois fazer o mesmo com os estudantes de outra plataforma.

Tem-se, portanto, uma aplicação móvel desenvolvida com tecnologia híbrida para conectar a um Raspberry Pi. Esse por sua vez processa os comandos dos estudantes para fazer música ou encontrar informações dos estudantes ou cursos. Para tanto, esse sistema é fundamentado no princípio de *Bring Your Own Device (BYOD)* e no método *Orff-Schulwerk*.

1.1 OBJETIVO GERAL

Propiciar o ensino e a aprendizagem de música de maneira colaborativa por meio de um aplicativo móvel.

1.2 OBJETIVOS ESPECÍFICOS

- Desenvolver um sistema de controle de atividades, a partir do Orff-Schulwerk, capaz de controlar as atividades propostas pelo professor bem como a sua realização por parte dos estudantes.

- Desenvolver um servidor de aplicação utilizando Raspberry Pi para gerenciar colaboração entre estudantes e a execução de comandos MIDI. Dessa forma, criar-se-ia um dispositivo portátil para uso do professor e estudantes.

- Desenvolver uma aplicação móvel multi-plataforma que propicie a interação dos estudantes com o sistema e o servidor descrito anteriormente.

1.3 MOTIVAÇÃO

A partir dos conhecimentos referentes à Educação Musical e interesse na execução coletiva de música utilizando tecnologias digitais surgiu a motivação de desenvolver algo para esse fim e também poder oferecer novas ferramentas para o ensino musical.

Essas novas ferramentas consistiriam de um sistema portátil e de fácil instalação pelo professor. Haja vista que em algumas escolas não há salas específicas para o ensino de música, isso diminuiria o tempo de trânsito do professor de uma sala para outra.

Da perspectiva do estudante também propicia-se utilizar dispositivos já disponíveis a ele. Para tanto, o interessante desse trabalho é o emprego por parte do estudante de seus próprios dispositivos móveis.

1.4 JUSTIFICATIVA

Com esse trabalho busca-se criar uma alternativa de baixo custo e que propicie aos estudantes das escolas despertarem em si o interesse pela produção musical. Também se aborda a problemática imposta dos avanços tecnológicos e seus impactos nos sistemas de ensino. Haja vista que cada vez mais smartphones e outros dispositivos inteligentes fazem parte da realidade da sociedade, essa realidade transborda para dentro do ambiente escolar. Citando Trexler(2007):

> Observando aprendizado móvel em um contexto mais amplo, temos que reconhecer que dispositivos móveis, pessoais e sem fio estão mudando radicalmente noções sociais de discurso e conhecimento. E são responsáveis por novas formas de arte, emprego, linguagem, comércio, depravação, crime e aprendizado também.

Portanto, busca-se uma discussão sobre algumas possibilidades de se fazer o aprendizado móvel de maneira proveitosa e mais eficiente para o ensino.

O presente trabalho busca, então, trazer uma nova possibilidade para o ensino de música, mas consciente que mesmo essa nova possibilidade possui limitações.

2. FUNDAMENTAÇÃO TEÓRICA

Para fundamentar o presente trabalhou buscou-se nos campos da Educação Musical e da Pedagogia conceitos norteadores dos desenvolvimentos realizados. Da Educação Musical recorreu-se aos princípios desenvolvidos por Carl Orff e da Pedagogia buscou-se o conceito de *Bring Your Own Device* (BYOD). Juntos a esses dois conceitos também serão utilizados Ambientes Virtuais Colaborativos e Aprendizagem Colaborativa com Suporte Computacional. Também buscou-se descrever melhor o Raspberry Pi e conceitos de desenvolvimento híbrido para aplicações móveis.

2.1 ORFF SCHULWERK

Carl Orff foi um compositor alemão que viveu de 1895 a 1981. Sua obra musical mais conhecida é a cantata *Carmina Burana* composta a partir de 24 poemas medievais intitulados *Carmina Burana*. Porém, além de compositor Orff desenvolveu um interesse grande pela educação musical e acabou por criar uma abordagem própria para o ensino musical. A essa abordagem por ele desenvolvida dá-se o nome de Orff-Schulwerk.

Referente ao ensino de música propriamente dito, essa abordagem se divide em cinco volumes onde apresenta-se conceitos básicos de música e, gradativamente, outros conceitos mais complexos são trazidos ao conhecimento (KUGLER, 2017).

Como forma de se transmitir esses conhecimentos utiliza-se o Instrumental Orff (IBIDEM, 2017). Esse instrumental Orff se configura em diversos instrumentos de percussão, podendo ser instrumentos de altura definida (xilofones, metalofones ou jogos de sino (*Glockenspiel*) ou de instrumentos de altura indefinida como tambores, maracas, clavas entre outros. Criando dessa forma um ambiente musical propício para o ensino, aprendizagem e experimentações musicais.

Essa abordagem de Orff incentiva a improvisação e produção sonora por parte dos estudantes. Com isso busca-se um envolvimento maior por parte dos estudantes

e oferece-se a eles maior liberdade para experimento dentro daquilo que eles já conhecem sobre música (IBIDEM, 2017). Porém, a partir desse conhecimento outros são apresentados conforme o tempo e isso leva os estudantes a aprofundar o conhecimento de maneira mais lúdica e intrigante para eles.

O trabalho pedagógico de Orff foi sistematizado em uma série de livros cujo primeiro volume recebeu o nome de "*Music for children – Volume 1*". Nesse livro se inicia com cantigas de roda e canções infantis (ORFF & KEETMAN, 1958). Este TCC emprega tais cantigas no primeiro exercício proposto no sistema desenvolvido. Ainda segundo ORFF & KEETMAN (1958), esse primeiro volume busca atingir três principais objetivos:

- Destina-se a todas as crianças, independente do conhecimento musical de cada uma;
- Não se configura uma instrução musical pura;
- O nível de exigência varia de acordo com a capacidade de cada criança.

Com esses três objetivos atinge-se uma liberdade criativa maior para as crianças além de mais cooperação entre elas. Isso porque enquanto uma criança mais talentosa pode improvisar uma melodia mais complexa em um instrumento, outra poderá acompanha-la batendo o ritmo ou com uma linha musical mais simples (ORFF & KEETMAN, 1958).

Para implementar esse princípio e permitir que o sistema toque uma cantiga de roda decidiu-se por criar um arquivo JSON descrito em melhores detalhes no capítulo quatro. Buscou-se, através do uso desse formato de arquivo, encontrar algo de rápido processamento e que contivesse as informações necessárias para sua execução. Também colaborou para essa decisão a possibilidade de transmitir os comandos MIDI contidos do arquivo JSON para outros aparelhos. Essa praticidade propicia ao sistema analisar melhor as interações dos estudantes do que persistir um arquivo em formato MD4 apenas com os sons executados.

Referente ao instrumental Orff, procurou-se substituir os instrumentos de altura definida pelo aplicativo móvel desenvolvido. Nesse aplicativo não se contempla os instrumentos de percussão pura ou as palmas. Esses ainda seriam executados pelos estudantes com instrumentos tradicionais. Porém, mesmo contendo apenas os

instrumentos de altura definida, se possibilita aos estudantes experimentarem melodias e partirem de cantigas de roda para exercícios mais complexos musicalmente.

2.2 *BRING YOUR OWN DEVICE* (BYOD)

Como os dispositivos móveis têm transformado radicalmente as noções sociais de conhecimento e aprendizado (TREXLER, 2007), apresenta-se para a educação a questão de como trabalhar em um ambiente de aprendizado formal com as novas tecnologias surgidas nos últimos anos. “O uso de dispositivos móveis aumenta gradualmente e se diversifica através de todos os setores da educação, e através de países desenvolvidos e em desenvolvimento” (IBIDEM, 2007).

McLean (2016) argumenta em seu artigo que cada vez mais se busca uma abordagem um-para-um entre estudante e aparelho eletrônico. Isso tem levado as escolas a procurarem alternativas para fornecer os aparelhos para cada estudante. Segundo essa autora, o método de fornecer tecnologia para os estudantes iniciou-se com os laboratórios de informática, passando para *learning pods*. Desses últimos passou-se para notebooks pessoais e por fim para programas de um-para-um no fornecimento de tecnologia móvel. Esse último passo levou as escolas a procurarem alternativas para que ela forneça a tecnologia aos estudantes. Isso se deveu aos aumentos dos custos relacionados às equipes de TI exigidas para suportar os sistemas e dispositivos inseridos na realidade escolar. Como alternativa encontrou-se o princípio de BYOD no qual os pais dos estudantes se responsabilizam pela tecnologia da mesma forma como se responsabilizam pelos livros didáticos.

Cada estudante ser responsável pelo seu próprio dispositivo aliviou o orçamento das escolas, mas também gerou maior comprometimento e participação dos estudantes no decorrer das disciplinas (MCLEAN, 2016). Isso porque com o uso de seus próprios aparelhos o ensino e a interação dos estudantes com as disciplinas não se resumiram às salas de aula, transbordando para o tempo dos estudantes fora da escola.

Apesar dessas vantagens descritas, McLean (2016) também aponta limitações ao uso do BYOD em escolas primárias. Ela enumera duas dificuldades importantes de mencionar, a saber, a equiparidade entre os aparelhos dos estudantes e a pressão para os professores se familiarizarem com novas tecnologias trazidas para o ambiente escolar.

Apesar das possibilidades e dificuldades apresentadas pelo BYOD, ainda faltam pesquisas que mais precisamente descrevam os impactos de BYOD. Necessita-se de mais estudos e tentativas de implementar BYOD em ambiente escolar (MCLEAN, 2016). Cabe ressaltar que se considerar o contexto de países em desenvolvimento, tal conceito poderia ser entendido como discriminatório, pois demanda recursos que estudantes da rede pública de ensino poderiam não ter disponíveis.

Partindo desse conceito de BYOD descrito acima, desenvolveu-se nesse trabalho uma aplicação móvel desenvolvida com tecnologia híbrida. Essa aplicação apresentaria as aulas e atividades referentes ao estudante e também seu progresso. Cada estudante então teria, em seu dispositivo, o aplicativo que substituiria o instrumental Orff descrito acima. Esses diferentes estudantes se conectariam ao Raspberry Pi simultaneamente para o aprendizado, bem como realização, musical.

Como mencionado anteriormente, um dos problemas de encarregar os estudantes de trazerem seus próprios dispositivos móveis para a sala de aula está na desigualdade entre os aparelhos dos estudantes. Para mitigar esse possível problema e evitar que performances individuais de cada aparelho venham a atrapalhar o transcorrer dos estudos optou-se por empregar uma tecnologia híbrida no desenvolvimento da aplicação. Dessa forma todos os estudantes possuem o mesmo programa e não se distingue entre diferentes sistemas operacionais e aparelhos.

2.3 APRENDIZAGEM COLABORATIVA

Castro e Menezes (2010) discorrem sobre a importância do aprendizado coletivo e colaborativo. Para esses autores existem dois princípios norteadores do aprendizado colaborativo, a saber:

- Os estudantes trabalham juntos buscando aprender;
- Os estudantes são responsáveis também pela aprendizagem dos demais.

Porém, para esses autores, apesar dos avanços na área de Aprendizagem Colaborativa com Suporte Computacional (CSCL), as práticas realizadas ainda possuem caráter presencial. No entanto, com os recursos da Web 2.0 cada vez mais existe a demanda para a utilização de recursos computacionais na aprendizagem.

Nesse trabalho busca-se unir esses dois princípios descritos com os princípios de Orff para o ensino de música. Essa relação é possível porque com o aprendizado de música de Orff o caráter colaborativo é um dos seus princípios e, portanto, verifica-se um ponto de convergência entre Castro e Menezes (2010) e Carl Orff. Porém, com relação à crítica feita quanto ao caráter presencial do CSCL, esse trabalho ainda possui foco na sala de aula e presença dos estudantes. No entanto, quer se possibilitar a esses estudantes utilizarem o aplicativo móvel para estudos fora da sala e realização de tarefas estando os estudantes distantes uns dos outros.

Castro e Menezes (2010) também apresentam diversos métodos de aprendizagem coletiva, interessando para esse trabalho o método intitulado ‘Investigação em grupo’. Esse método possui quatro componentes básicos que devem ser integrados para ocorrer o aprendizado. Os quatro componentes são:

- Investigação: Estudantes são apresentados a um tema e o investigam sozinhos;
- Interação: Estudantes interagem entre si compartilhando as descobertas feitas pelas suas próprias investigações;
- Interpretação: Alunos interpretam os resultados alcançados coletivamente;
- Motivação intrínseca: Motivação de cada estudante para o estudo do tema proposto.

2.4 RASPBERRY PI

O Raspberry Pi é um computador de placa única onde se pode instalar um sistema operacional e assim interagir com o mundo real(RASPBERRY PI, 2017). Para se poder utilizar esse computador necessita-se instalar um sistema operacional para tal. A fundação Raspberry oferece a possibilidade de baixar alguns sistemas operacionais compatíveis com ele. Até o momento da escrita desse trabalho os seguintes OS estavam disponíveis no site da organização Raspberry(IBIDEM PI, 2017):

- Raspian;
- LibreELEC;
- OSMC;
- RISC OS;
- Arch Linux.

Além da capacidade de processamento e de instalações de diversos softwares, outro ponto interessante para a presente atividade está em que o Raspberry Pi possui duas saídas de áudio, necessárias para a execução das notas tocadas pelos estudantes nos seus aplicativos(IBIDEM, 2017). Uma delas é a saída de áudio HDMI (do inglês *High-Definition Multimedia Interface*, Interface Multimídia de alta definição) e a outra um conector TRS (do inglês *Tip-Ring-Sleeve*, ponta-anel-capa). Essas duas saídas de áudio podem ser utilizadas com o projeto desenvolvido, bastando-se possuir os cabos e demais dispositivos necessários. Não se desenvolveu nada específico para uma opção ou outra, sendo assim permitido que se escolha futuramente entre elas. O próprio sistema operacional percebe qual saída está conectada e envia o som para ela.

2.5 *MUSICAL INSTRUMENT DIGITAL INTERFACE* (MIDI)

Tecnologia utilizada para comunicar e criar música entre dispositivos eletrônicos. Seu funcionamento divide-se em três etapas(MIDIORG, 2017):

- Mensagens MIDI: Também chamado de protocolo MIDI, é através dessas mensagens que o sintetizador saberá como executar a nota. Nessas mensagens encontram-se as informações sobre a dinâmica, instrumento, nota, tempo e canal para execução;
- Transporte físico para MIDI: Inicialmente idealizado para comunicação entre dois teclados, atualmente encontra-se em computadores e smartphones;
- Formato de arquivo MIDI: Arquivo criado para execução por um sintetizador MIDI[3].

A execução e todo o processamento acima necessita de um sintetizador MIDI que processa as mensagens e envia para execução. O sintetizador escolhido foi o FluidSynth que será abordado futuramente. Esse se encarrega também da implementação das especificações MIDI.

Para permitir a execução simultânea por parte dos estudantes, existe na tecnologia MIDI a possibilidade de designar diferentes canais para cada execução. Dessa forma, cada estudante recebe um número de canal que ficará persistida no banco de dados. Esse número não se alterará e cada vez que o estudante conectar ele será atribuído a ele.

A tecnologia MIDI existe há cerca de 30 anos e devido ao longo histórico desse formato, bem como sua fácil implementação no Raspberry Pi decidiu-se utilizar essa tecnologia no presente trabalho. Colaborou para essa decisão o fato de já existirem para Python bibliotecas para integrarem o MIDI com o servidor de aplicação. Nesse presente trabalho empregou-se a biblioteca MIDI desenvolvida pelo projeto pygame descrito na próxima seção.

2.6 PYTHON

Linguagem de programação de alto nível desenvolvida pela Python Software Foundation(PYTHON, 2017). Possui diversos módulos em C para funcionalidades de

[3] As especificações gerais MIDI não se ocupam em definir como o som dos instrumentos devem ser, dando assim liberdade para cada fabricante implementar suas próprias ideias e concepções estéticas(MIDIORG, 2017).

sistema bem como outros escritos em Python para prover soluções padronizadas para problemas recorrentes(IBIDEM, 2017). Dentre esses problemas encontram-se os de interface entre Python e sintetizador MIDI, interface entre Python e Sistema Gerenciador de Banco de Dados e disponibilização de serviços WEB escritos em Python. Para a primeira tarefa descrita recorreu-se à biblioteca pygame descrita a seguir.

Pygame – biblioteca gratuita e de código aberto utilizada para produção de aplicações multimídia(PYGAME, 2017). Essa biblioteca é desenvolvida por uma comunidade de desenvolvedores e sua implementação pode ser feita em sistemas operacionais Windows, MacOS ou Linux. Dentre das distribuições Linux suportadas está o Raspian, sistema operacional do Raspberry Pi utilizado neste TCC.

Dentre as bibliotecas desenvolvidas pelo projeto pygame encontra-se uma relacionada à execução de comandos MIDI, chamada pygame.midi(IBIDEM, 2017). Esse módulo MIDI pode enviar os comandos para diversos dispositivos (reais ou virtuais) e suporta as plataformas Windows, macOS e Linux. Empregou-se esse módulo MIDI como uma interface entre a aplicação Web desenvolvida em linguagem Python e o servidor MIDI FluidSynth descrito a seguir.

Além da interface com o FluidSynth, para permitir a criação de serviços WEB nesse projeto utilizou-se o *framework* web.py(WEBPY, 2017). Novamente, devido a simplicidade do módulo decidiu-se pelo seu uso.

A aplicação desenvolvida em Python destina-se a ser a interface do usuário com o FluidSynth, mas também do usuário com a instância de banco de dados criada. Para essa última interface utilizou-se a biblioteca MySQLdb. Essa biblioteca funciona como uma API para conectar a aplicação WEB com o servidor de banco de dados.

2.7 FLUIDSYNTH

O FluidSynth é um sintetizador MIDI destinado a executar comandos MIDI ou executar arquivos de áudio(FLUIDSYNTH, 2017). Possibilita-se instalar esse sintetizador para receber esses comandos no Raspian via entrada padrão e enviar para as saídas de áudio.

Nesse trabalho a sequência para a execução dos comandos MIDI pelo FluidSynth começa pelo usuário no aplicativo apertando o botão que representa uma nota. O aplicativo então envia para o servidor de aplicação uma String URL onde se encontra as informações necessárias para o MIDI. Esse servidor de aplicação processa a String buscando as informações desejadas e envia, via a biblioteca pygame.midi descrita anteriormente, um comando NOTE_ON para o FluidSynth. Ao soltar o botão pressionado, o usuário dispara uma nova String url para o servidor, que processa a cadeia de caracteres e passa o comando para o FluidSynth encerrar a execução da referida nota. Na documentação do sistema desenvolvido encontra-se maiores detalhes de como esses comandos são enviados e processados pelas partes integrantes.

2.8 MYSQL

O MySQL é um Sistema Gerenciador de Banco de Dados (SGBD) desenvolvido, distribuído e suportado pela Oracle(MYSQL, 2017). De acordo com a documentação oficial encontrada no site do MySQL(IBIDEM, 2017), esse sistema possui seis características principais listadas abaixo:

- MySQL é um SGBD: Um banco de dados é uma coleção estruturada de dados. Esses dados ou informações precisam ser acessadas e administradas para poderem ser utilizadas de maneira coerente. Para realizar essas tarefas utiliza-se um SGBD como o MySQL;
- Banco de dados MySQL são relacionais: Um banco de dados relacional guarda as informações em diferentes tabelas e não em apenas um grande repositório. Dessa forma possibilita-se salvar e organizar essas tabelas em arquivos físicos otimizados. Para otimizar e organizar melhor os arquivos criam-se regras que governam os relacionamentos entre as distintas tabelas do sistema. O SGBD então assegura que essas regras sejam seguidas e se evita informações duplicadas ou inconsistentes;

- MySQL é um software de código aberto: Com essa maneira de distribuição, qualquer pessoa pode instalar e até alterar o código do MySQL para atender suas necessidades;
- MySQL Database Server é rápido, confiável, escalonável e fácil de usar: O servidor de banco de dados do MySQL foi desenvolvido para lidar com grandes quantidades de dados e é possível utilizá-lo em um computador pessoal sem impactar os demais processos;
- MySQL Server funciona em cliente/servidor ou sistemas embarcados: MySQL é um sistema cliente/servidor que permite um gama grande de programas e bibliotecas utilizadas para se comunicar com ele;
- Existe uma grande gama de programas para contribuir com MySQL: MySQL possui diversas funcionalidades e isso permite afirmar que inúmeras linguagens de programação e aplicações possuem suporte ao MySQL.

Baseado nesses pontos descritos acima, decidiu-se criar e utilizar um SGBD MySQL. Contribuíram também para essa escolha o fato desse SGBD ser de fácil instalação e integração com a aplicação em Python desenvolvida. O SGBD encontra-se no Raspberry Pi e comunica-se com essa aplicação. Dessa forma, permite-se um controle das atividades desenvolvidas pelos estudantes, bem como o gerenciamento da sua evolução nas aulas de música feitas.

2.9 *JAVASCRIPT OBJECT NOTATION* (JSON)

Acrônimo para *JavaScript Object Notation*, JSON é um formato utilizado para transmissão de valores. Sua leitura e escrita por parte de humanos é de fácil execução e também o processamento por parte de máquinas(JSON, 2017). JSON constitui-se de duas estruturas:

- Coleção de pares atributo-valor;
- Uma lista ordenada de valores.

Esses valores mencionados são universais e praticamente todas as linguagens de programação modernas empregam algum tipo de biblioteca para sua interpretação ou geração(IBIDEM, 2017). Ao combinar essas estruturas, elas podem assumir os formatos de um objeto, *array*, valor ou número.

Empregou-se esse formato para gerar arquivos onde constam as atividades a serem desenvolvidas pelos estudantes. Dessa forma, no servidor de aplicação existem arquivos possíveis de serem lidos e processados pelo Raspberry Pi. E, além do processamento por parte desse, o formato JSON é legível por seres humanos.

Considerando a perspectiva do estudante, ao persistir suas repostas pelo aplicativo, este gera um arquivo JSON com a mesma estrutura da atividade inicial, possibilitando assim uma comparação entre o exercício proposto e o resultado atingido pelo estudante.

2.10 DESENVOLVIMENTO MULTI-PLATAFORMA

Xanthopoulos e Xinogalos (2013) descrevem que para o desenvolvimento de uma aplicação móvel as empresas precisam criar uma aplicação em uma plataforma para depois reescrever o programa para outra. Com isso, não se aproveita o código da primeira aplicação e ainda se requer conhecimentos aprofundados de cada plataforma. Nesse trabalho não pretende-se discutir profundamente as possibilidades de desenvolvimento multi-plataforma, porém abaixo lista-se quatro possibilidades apresentadas por esses autores.

- Aplicações Web: Desenvolvidas utilizando HTML5 e Javascript. São aplicações não instaladas no aparelho;
- Aplicações híbridas: Aplicações desenvolvidas também utilizando HTML5 e Javascript, porém instaladas e executadas localmente nos aparelhos;
- Aplicações interpretativas: Nessas aplicações o usuário interage com interfaces específicas da plataforma, porém a lógica da aplicação roda de maneira independente em outra linguagem;
- Aplicações geradas: Para cada plataforma gera-se uma aplicação específica.

Para esse trabalho implementou-se a tecnologia híbrida conhecida como Ionic(IONIC, 2017). Esse *framework* busca possibilitar o desenvolvimento de aplicações móveis utilizando tecnologias existentes como HTML, CSS e JavaScript(IBIDEM, 2017). Seu foco principal está na aparência dos aplicativos

desenvolvidos oferecendo diversos componentes visuais para a geração de uma aplicação móvel. Esses componentes nada mais são que peças para montagem permitindo a criação de aplicativos visualmente mais elegantes e amigáveis ao usuário(IBIDEM, 2017). A navegação entre telas funciona adicionando novas telas à pilha existente. Também se possibilita o uso de alertas para o usuário. Sob esse *framework* existe o Angular que se responsabiliza pelos componentes da API para implementação do Ionic.

Com a utilização de Ionic e o conceito de desenvolvimento multi-plataforma busca-se diminuir a desigualdade gerada pelos diferentes aparelhos trazidos pelos estudantes, além de fornecer apenas uma versão do aplicativo para os diferentes sistemas operacionais.

2.11 INSTRUMENTO MUSICAL DIGITAL AUTÔNOMO

Ao utilizar o Raspberry Pi como receptor de comandos do usuário, interpretador desses comandos e consequente execução musical surge a necessidade de se saber como realizar essas tarefas de maneira computacionalmente eficiente e musicalmente satisfatória. Franco e Wanderley (2016, p. 358-359) apresentam a discussão entre dispositivos dedicados e computadores de uso geral para instrumentos musicais digitais. Para esses autores, um dispositivo dedicado encontra-se pronto para o uso e não requer complexas conexões, sistemas operacionais ou ativar determinados processos computacionais para sua utilização. Por outro lado, um computador de uso geral pode processar um código escrito e gerar música a partir desses.

Porém, um problema apontado pelos autores reside no fato de atualizações de sistemas operacionais poderem quebrar funcionalidades das aplicações. Além disso, computadores normalmente desempenham diversas outras tarefas e isso pode resultar em perda de performance.

Tendo essas perspectivas em vista, Franco e Wanderley (2016) desenvolvem o *framework* ‘Prynth’ que segundo eles “é um *framework* de código aberto para a criação de instrumentos musicais digitais em dispositivos dedicados”.

Para mitigar esses problemas apresentados, no *framework* ‘Prynth’ apresentado(PRYNTH, 2017) decidiu-se pelo Raspberry Pi como plataforma ‘principalmente pela sua popularidade, suporte e adoção por uma comunidade de usuários em diversos e amplos projetos de pesquisa, educação e entusiastas’ (FRANCO e WANDERLEY, 2016).

Esse trabalho empregou Raspberry Pi por causa da sua comunidade internacional, mas também pela capacidade computacional ofertada. Dessa forma possibilita-se criar um dispositivo central com autonomia para executar comandos musicais e armazenar as informações referentes às atividades e progressos dos estudantes.

2.12 APLICATIVO NA GOOGLE PLAY STORE

Ao se acessar o site da Google Play Store e entrar o verbete ‘Orff’ no campo de busca encontrou-se apenas um aplicativo que emprega deliberadamente o método Orff[4]. Esse aplicativo possui quatro funcionalidades principais[5]:

- Carregar e alterar peças musicais;

- Executar um ou mais instrumentos musicais;

- Salvar arranjos próprios;

- Compor e salvar linhas musicais próprias.

Dessas funcionalidades descritas empregou-se no presente trabalho a possibilidade de carregar uma peça musical e executar apenas determinadas faixas dela. Outra opção disponível é a de salvar as execuções próprias dos estudantes. A diferença entre essa aplicação encontrada e a desenvolvida está em que a produção dos sons musicais não ocorrem no dispositivo móvel, mas a partir do Raspberry Pi.

---

[4] https://play.google.com/store/search?q=orff&c=apps Acesso em 27/11/2017

[5] https://play.google.com/store/apps/details?id=com.schottmusic.playorff Acesso em 27/11/2017

Dessa forma facilita-se a colaboração entre os estudantes pois a fonte sonora é a mesma para todos os estudantes.

Uma vantagem apresentada pela aplicação encontrada é que ela possibilita a mudança de notas na tela de execução musical. Isso significa dizer, por exemplo, que na aplicação desenvolvida as notas estão fixas na tela e o estudante não pode alterar a nota F3 para F#3. Porém na aplicação encontrada requer-se o download de arquivos adicionais para a execução da mesma. Isso pode causar o uso excessivo de memória do dispositivo. No programa aqui desenvolvido não se busca o arquivo musical propriamente dito, mas apenas um arquivo JSON com as informações para formatar e executar as mensagens MIDI.

3. MÉTODO

3.1 FUNDAMENTAÇÃO TEÓRICA

O método empregado para o presente trabalho foi uma pesquisa bibliográfica a partir da base de dados Google Scholar. Nessa ferramenta buscou-se artigos que tratavam do tema BYOD. Encontrou-se diversos artigos relacionados a BYOD, porém muitos referentes ao meio empresarial. Filtrou-se então para se restringir os artigos referentes à implementação desse princípio em ambiente de aprendizado. A partir dos artigos encontrados pesquisou-se as referências desses para verificar se informações adicionais se encontravam.

Para os fundamentos de educação musical buscou-se no site oficial mantido pela Carl-ORFF-Stiftung(KUGLER, 2017) as informações referentes aos princípios de Orff no ensino musical. Além dessa fonte bibliográfica buscou-se na obra escrita de Orff (ORFF E KEETMAN, 1958) para o ensino de música para crianças os exemplos para atividades musicais do presente trabalho. Porém, apesar da base musical ser advinda desse material, para a orquestração dos instrumentos nos exemplos o autor desse trabalho compôs pequenos arranjos. Isso se justifica pois o Orff-Schulwerk permite ao professor implementar suas próprias ideias quando julgar necessário.

Também se buscou em livros sobre aprendizagem colaborativa para descrever quais os seus aspectos mais importantes. Com o artigo encontrado (Castro e Menezes, 2010) possibilitou-se uma análise teórica do tema para futura implementação no sistema criado.

Sobre o uso do Raspberry Pi para instrumento musical digital encontrou-se um artigo sobre um desenvolvimento atualmente em curso na Universidade McGill (FRANCO E WANDERLEY, 2016). Esse trabalho encontra-se em processo de desenvolvimento por Ivan Franco e possui convergências pertinentes para esse trabalho.

Com relação linguagens, bibliotecas e *framework*s para o desenvolvimento propriamente dito do sistema buscou-se nos sites oficiais de cada um sua descrição

e explicação. Dessa forma, possui-se uma fonte primária para descrever e basear o presente trabalho.

A partir dos estudos efetuados buscou-se então desenvolver o sistema de acordo com os seguintes princípios:

- Uma mesma aplicação para todos os estudantes envolvidos: conforme mencionado anteriormente, um problema do BYOD é a diferenciação entre os estudantes causados pelos diferentes dispositivos móveis. Esse problema pode até gerar a exclusão de alunos que não tenham condições de possuir um dispositivo. Para mitigar esse problema, uma aplicação única para todos os dispositivos ajuda nesse processo;
- Resposta imediata do sistema para as interações dos estudantes: como mencionado por Franco e Wanderley, um instrumento musical digital autônomo deve ter uma resposta rápida entre a ação do usuário e o som produzido;
- Informações sobre atividades e progresso dos estudantes devem ser persistidas: para possibilitar a aprendizagem colaborativa descrita no tópico 2.3 as atividades devem ser persistidas a fim de permitir aos alunos retornarem ao ponto em que encerraram a última atividade.

3.2 CONSTRUÇÃO DA FERRAMENTA

Para a construção do sistema empregou-se diferentes softwares e IDE's (do inglês *Integrated Development Environment*, Ambiente de Desenvolvimento Integrado). Relacionado ao hardware utilizou-se um computador MacBook Air com sistema operacional macOS Sierra pois assim pode-se testar o sistema utilizando dispositivo com sistema operacional iOS e Android. Abaixo segue a lista de softwares utilizados no desenvolvimento:

- Atom: Editor de texto que busca uma combinação entre adaptação e usabilidade(ATOM, 2017). Utilizado para o desenvolvimento da aplicação móvel, essa combinação possibilita a codificação utilizando Ionic. Porém, quando da compilação da aplicação, esse software não possui essa funcionalidade. Isso levou à utilização do software Ionic CLI.

- Ionic CLI: Interface de linha de comando utilizado para o desenvolvimento de aplicações Ionic(IONIC, 2017). Utilizado para a criação do projeto Ionic, as páginas da aplicação e os serviços do aplicativo.
- Xcode: Editor utilizado para o desenvolvimento de aplicações para plataformas da Apple(XCODE, 2017), possui também a possibilidade de desenvolver utilizando a linguagem Python. Com isso utilizou-se esse software para desenvolver o servidor de aplicação. Para tanto, acessava-se remotamente os arquivos presentes no Raspberry Pi e abria-se para edição no computador onde também se desenvolveu a aplicação móvel.
- MySQL Workbench: Ferramenta visual para modelar o banco de dados(MYSQL, 2017). Ela possibilita o acesso remoto a outro computador e com isso possibilitou-se desenhar e criar o banco de dados para o sistema a partir do computador MacBook Air.

3.3 TESTES

Para testar o sistema desenvolvido optou-se pela implementação do *framework* DECIDE descrito por Preece, Rogers e Sharp (2005, p. 368) como contendo os seis passos abaixo:

1. Determinar as *metas* que a avaliação irá abordar.

2. Explorar as *questões* específicas a serem respondidas.

3. Escolher o *paradigma de avaliação* e as *técnicas* de respostas para as perguntas.

4. Identificar as *questões práticas* que devem ser abordadas, como a seleção dos participantes.

5. Decidir como lidar com as questões éticas.

6. Avaliar, interpretar e apresentar os dados.

Esse *framework* norteou a fase de testes do sistema para permitir a realização e interpretação dos dados coletados.

3.3.1 METAS

Para nortear os testes e as avaliações dos resultados definiu-se quatro metas principais para eles.

- Tempo de resposta do sistema: Como o sistema busca substituir instrumentos musicais reais, o tempo de resposta deve ser quase instantâneo, assim como ocorre com os instrumentos reais;
- Compreensão da atividade proposta aos estudantes: Embora a figura do professor seja importante para o funcionamento do sistema, as atividades propostas devem ser facilmente compreendidas pelos estudantes;
- Persistência das respostas enviadas pelos estudantes: As atividades devem ser salvas para avaliação dos professores ou para histórico do estudante na disciplina;
- Execução coletiva: O sistema deve permitir a execução de dois ou mais alunos simultaneamente.

3.3.2 QUESTÕES

Para cada uma das metas definidas acima formulou-se questões específicas a serem respondidas pelos testes. Para o tempo de resposta do sistema as questões encontram-se na tabela 1.

TABELA 1 – QUESTÕES PARA META 1

Questão	**Propósito**
Qual foi o maior tempo de resposta entre a interação do usuário e a resposta do sistema?	Verificar se o sistema responde instantaneamente à interação dos usários
Segundo os voluntários, como eles classificam o tempo de resposta do sistema às interações?	Validar se o sistema está com tempo de resposta boa e identificar anomalias.

FONTE: O autor (2017).

A compreensão da atividade proposta aos estudantes visa identificar o quanto o sistema é autoexplicativo sob a perspectiva do usuário. A tabela 2 traz as questões formuladas para essa meta.

TABELA 2 – QUESTÕES META 2

Questões	**Propósito**
Segundo os voluntários, quão intuitivo estão as Atividades?	Perceber o quanto o sistema é auto-explicativo.

FONTE: O autor (2017).

Um dos objetivos específicos desse trabalho é o de desenvolver um sistema para gerenciar as atividades dos alunos. Portanto, deve ele persistir as atividades do estudante. Para testar isso a tabela 3 traz as questões referentes a essa meta:

TABELA 3 – QUESTÕES META 3

Questões	**Propósito**
Quantas atividades foram enviadas pelos alunos e quantas encontram-se persistidas no servidor?	Validar que os arquivos se encontram persistidos

FONTE: O autor (2017).

O presente sistema deve permitir a execução coletiva por parte dos estudantes e para validar essa meta a tabela 4 traz as questões para ela formulada.

TABELA 4 – QUESTÕES META 4

Questões	**Propósito**
Houve queda do sistema ao serem realizadas várias interações simultâneas?	Um travamento causado pelo excesso de interação do usuário significa que o sistema ainda não comporta confiavelmente vários alunos simultâneos
Algum estudante não conseguiu executar a tarefa desejada por falha na aplicação móvel?	Se a aplicação móvel parar de funcionar significa que o estudante não está com um programa confiável

FONTE: O autor (2017).

3.3.3 MÉTODO DE AVALIAÇÃO E TÉCNICAS

O método de avaliação escolhido para esses testes é o de Testes de usabilidade. Conforme Preece, Rogers e Sharp (2005, p. 361)

> Os testes de usabilidade envolvem avaliar o desempenho dos usuários típicos na realização de tarefas cuidadosamente preparadas, por sua vez típicas daquelas para os quais o sistema foi projetado.

As tarefas preparadas para os testes do sistema descrevem-se na tabela 5.

TABELA 5 - ATIVIDADES PARA TESTES(CONTINUA)

TAREFA	**ATIVIDADES**	**RESULTADOS ESPERADOS**
Executar instrumentos individualmente	Cada estudante deve selecionar um instrumento e apertar o botão 'Exec. Instr.';	Todos os instrumentos devem ser executados ao menos uma vez
	Cada estudante deve executar o instrumento escolhido no teclado musical;	Alunos devem poder repetir o exemplo no teclado musical
	Após cada um executar seu instrumento individualmente, executá-los em conjunto;	Alunos devem executar o exemplo no teclado musical
	Todos devem executar suas partes coletivamente no teclado musical.	
Executar atividade	Um estudante deve executar a atividade apertando botão 'Play example'	Alunos devem ser capazes de compreender que o exemplo tocado é a atividade a ser desenvolvida
	Cada estudante deve procurar identificar seu próprio instrumento no exemplo (pode-se repetir várias vezes o passo anterior)	
Gravar resposta	Após treinar as atividades, cada estudante deve gravar sua resposta	Arquivos com as respostas individuais devem estar salvas no Raspberry Pi
	Ao terminar a gravação, enviar a resposta	
Ouvir a resposta	Voltar para a tela de selecionar a atividade e escolher a atividade que acabou de gravar	Respostas individuais devem ser carregadas para o dispositivo móvel
	Todos os alunos devem apertar o botão 'Play answer' ao mesmo tempo	Ao executar as respostas individuais coletivamente, o som resultante deve ser semelhante àquele executado pelos alunos
Livre execução	Na tela do teclado musical, os alunos estão livres para executar e tocar o que quiserem	Aos alunos deve ser possível tocar as notas musicais do teclado livremente

FONTE: O autor (2017).

Como técnica de avaliação, optou-se por observar usuários ao executar as tarefas listadas.

3.3.4 Questões práticas

Os testes acima serão realizados instalando-se a versão corrente do aplicativo em dispositivos móveis. Esses dispositivos então serão fornecidos aos voluntários que procurarão executar as tarefas. Essas execuções ocorrerão em conjunto com todos os voluntários reunidos em um mesmo ambiente. Elas também serão gravadas para futuras análises.

3.3.5 Questões éticas

Para cada um dos voluntários se disponibilizará um Termo de Consentimento Livre e Esclarecido onde as atividades estarão descritas. Esse se encontra no Apêndice I do presente trabalho.

3.3.6 Avaliar, interpretar e apresentar os dados.

Procurou-se interpretar as questões abertas do questionário na busca por padrões entre as respostas dos voluntários. Para as questões de múltipla escolha apresentou-se um quadro com as respostas.

4. DESCRIÇÃO DO SISTEMA

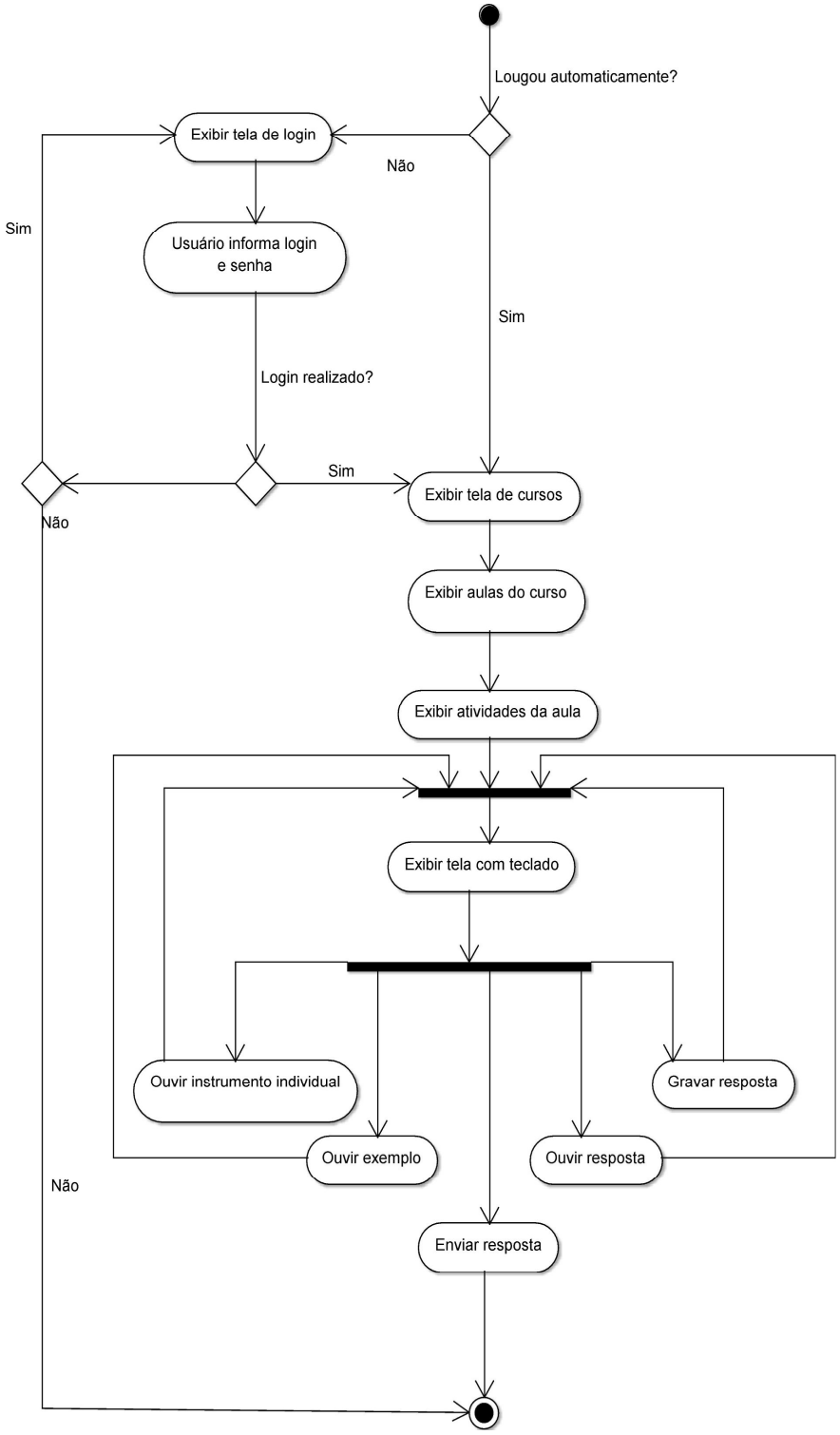
4.1 VISÃO GERAL

A partir dos conceitos descritos no capítulo 2 e das metas enumeradas no capítulo 3, buscou-se criar um aplicativo que possibilite ao estudante localizar suas tarefas a serem realizadas em um curso de música como também executar e fazer música coletivamente utilizando smartphones. Esse aplicativo deve ser o mesmo para os sistemas operacionais Android e iOS. Por isso optou-se pelo *framework* Ionic já descrito nesse trabalho.

Juntamente ao aplicativo desenvolveu-se um servidor de aplicação em Python e configurado em um dispositivo Raspberry Pi. Nesse disposivito optou-se por utilizar Raspian como Sistema operacional pois ele possui mais de 35.000 pacotes num formato de fácil e rápida instalação(RASPBIAN, 2017), além de ele ser o sistema operacional recomendado pela Fundação Raspberry Pi(RASPBERRY PI, 2017). Dentre esses pacotes mencionados estão contidos todos aqueles necessários para a execução e programação das atividades aqui propostas. Dessa maneira, possibilitou-se a instalação do servidor MIDI (FluidSynth), sistema gerenciador de banco de dados (MySQL) e dos respectivos módulos Python necessários para integração desses dois com o aplicativo móvel. Esse servidor desenvolvido no Raspberry Pi então interage com o servidor de banco de dados MySQL, também configurado nesse mesmo computador ou envia mensagens MIDI para o servidor MIDI presente no Raspberry. Para saber com qual servidor a aplicação deve se comunicar o servidor de aplicação avalia o comando enviado pelo usuário do seu dispositivo.

O protótipo desenvolvido possui como atividade básica permitir ao estudante se conectar ao sistema e acessar os materiais a ele destinados. Após essa conexão ao estudante se apresenta uma tela onde ele poderá executar diversas atividades referentes ao conteúdo da aula. A seguir segue um diagrama de atividades (Figura 1) que se inicia com o estudante iniciando a aplicação e ela tentando logar automaticamente com o usuário e senha persistido no celular e se encerra com ele enviando a sua resposta para a atividade proposta:

FIGURA 1 – DIAGRAMA DE ATIVIDADES

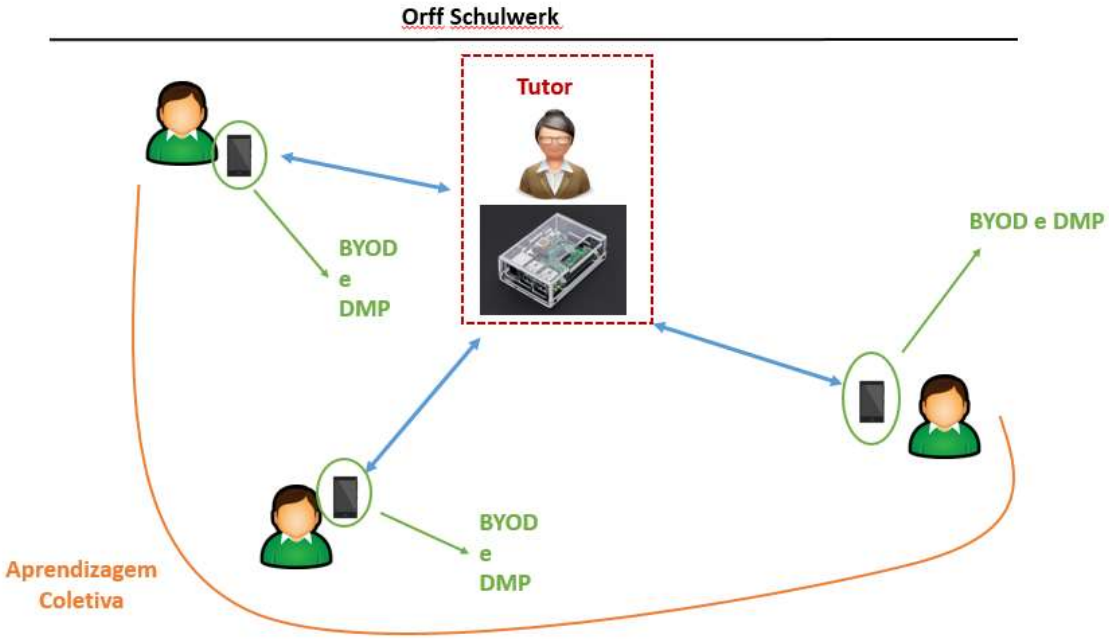


FONTE: O autor (2017).

4.2 ARQUITETURA DO SOFTWARE E DO HARDWARE

O sistema Jongui consiste de um aplicativo móvel para smartphones/tablets. Esses se conectam a um Raspberry Pi 3 onde existem os servidores de aplicação, de banco de dados e MIDI descritos nesse capítulo. A arquitetura do sistema foi organizada tal como representado na Figura 2:

FIGURA 2 – ARQUITETURA DO SISTEMA JONGUI



FONTE: O autor (2018).

Para a aplicação móvel desenvolveu-se uma interface de usuário onde é possível para o estudante escolher um instrumento a ser tocado. Esse instrumento representa um código que será interpretado pelo servidor MIDI[6]. A definição dos instrumentos disponíveis aos estudantes se dá pelo professor quando esse decide o exercício da aula. Aos estudantes se apresentam apenas os instrumentos a serem utilizados na determinada aula. Juntamente aos instrumentos, possibilita-se ao estudante nessa tela ouvir um exemplo de exercício bem como a parte referente a cada um dos instrumentos musicais.

[6] Seguiu-se nesse trabalho o padrão *General MIDI Level 1 Instrument Patch Map* descrito nas especificações MIDI encontradas em https://www.midi.org/specifications/item/gm-level-1-sound-set - acesso 22/11/2017

Portanto, ao estudante inscrito no curso aparece na tela do seu smartphone as opções de instrumentos, dos quais ele deverá selecionar um. Selecionado o instrumento, exibir-se-á um teclado no seu celular análogo ao instrumento e acima dele botões para ele ouvir o exercício da aula, gravar sua resposta para o exercício proposto ou tocar livremente com o aparelho. Ao final do período ou quando o estudante se sentir seguro, este poderá enviar a resposta do exercício para o sistema. Dessa maneira, torna-se possível ao professor controlar os exercícios feitos pelos estudantes bem como sua evolução.

Todas as informações mencionadas acima passam pelo servidor de aplicação. Esse consiste de diversos serviços web disponíveis aos estudantes e acessados pelo dispositivo móvel. Esse acesso se dá através de requisições HTTP enviadas do smartphone ao Raspberry Pi. Nesse, de acordo com o serviço invocado, diferentes objetos são instanciados. Essas instâncias podem ser de objetos referentes à interação com o banco de dados ou objetos que enviam as mensagens MIDI para o servidor FluidSynth. Para o primeiro tipo de interação retorna-se objetos JSON referentes ao resultado, para o segundo tipo o próprio som emitido pelo sistema se afigura na resposta à ação.

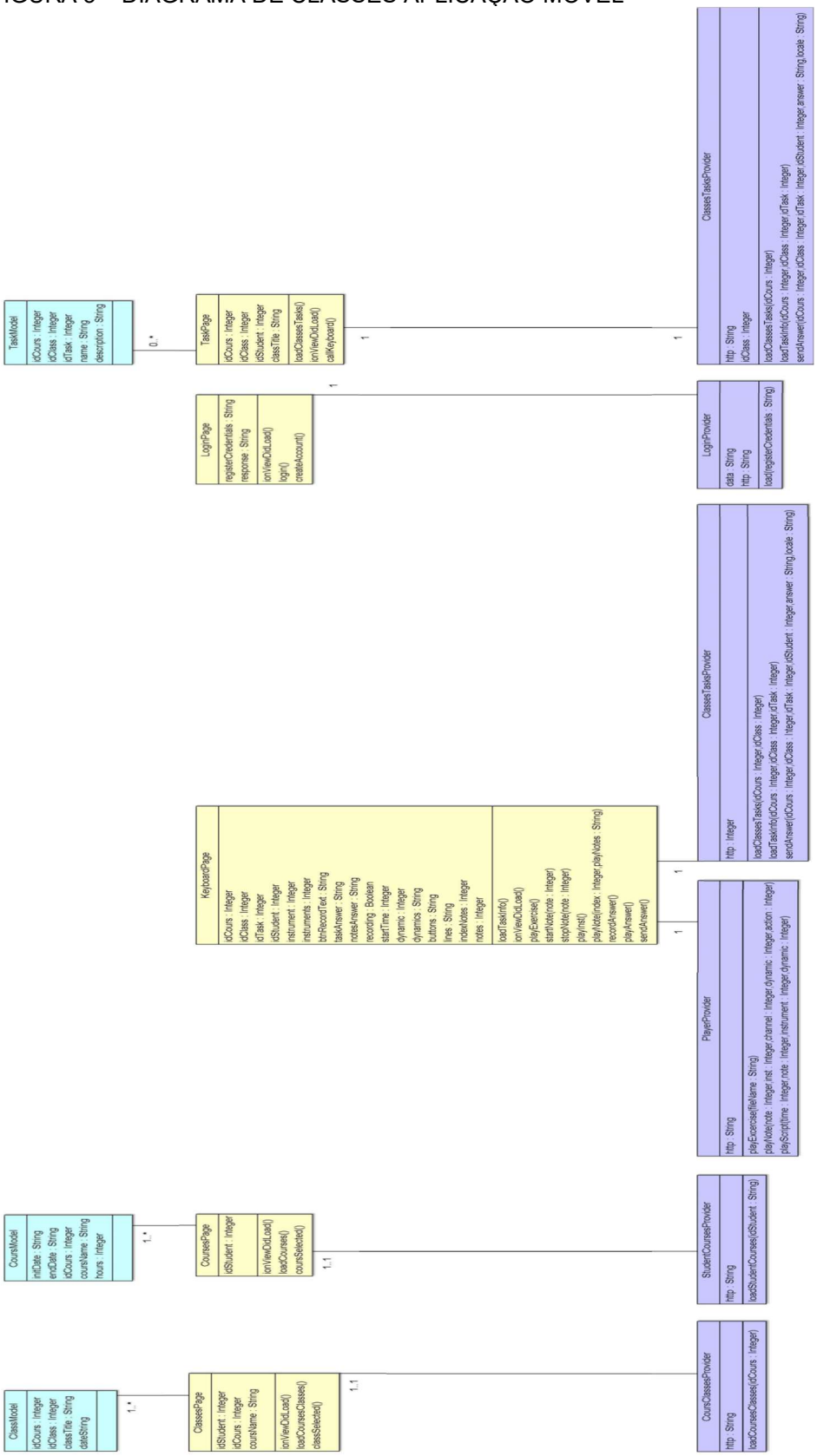
Com relação às informações do progresso dos estudantes, as informações são persistidas em uma instância de banco de dados MySQL executada no Raspberry Pi. Juntamente a isso, quando o estudante envia sua resposta para a atividade proposta, um arquivo JSON é criado e salvo em uma pasta no Raspberry Pi.

Descreve-se a seguir de maneira mais detalhada e técnica como ocorre a comunicação entre os componentes do sistema, bem como uma documentação do sistema desenvolvido e da aplicação móvel criada.

4.3 DOCUMENTAÇÃO DO APLICATIVO MÓVEL

Para a criação do aplicativo utilizou-se como fundamento o Diagrama de Classes exibido na figura 3.

FIGURA 3 – DIAGRAMA DE CLASSES APLICAÇÃO MÓVEL



FONTE: O autor (2018).

Esse dividiu-se de acordo com o conceito *Model, View, Controller* (MVC) e implementado de acordo com as especificações do Ionic descritas a seguir.

Como já mencionado, optou-se para esse trabalho a implementação de tecnologia híbrida para permitir que usuários de sistemas operacionais diferentes tenham a mesma interface gráfica. Para isso criou-se uma aplicação utilizando Ionic e organizada em seis interfaces de usuário. Dentro do *framework* Ionic, ao se criar uma página também se criam quatro arquivos cada um com as funções descritas abaixo(IONIC, 2017):

- pagina.html: Código HTML onde é possível inserir os componentes Ionic desejados;
- pagina.scss: Arquivo para definir os estilos e aparência de cada página;
- pagina.module.ts: Local onde se define os módulos utilizados na página;
- pagina.ts: Arquivo onde se escreve o código para processamento de cada página.

As seis interfaces de usuário criadas para essa aplicação podem ser definidas de acordo com a tabela 6:

TABELA 6 – DESCRIÇÃO DAS INTERFACES DO USUÁRIO

Interfaces	**Descrição**
CLASSES	Tela destinada a exibir as aulas de cada curso que o estudante está matriculado.
COURSES	Exibe os cursos que cada estudante está matriculado.
KEYBOARD	Tela com o teclado para execução musical por parte dos estudantes
LOGIN	Tela de login do usuário.
TASK	Tela que exibe as atividades de cada aula.

FONTE: O autor (2017).

Para permitir a execução do aplicativo criou-se três classes ‘Model’ para instanciar objetos com as informações dos cursos, aulas e atividades a serem

desenvolvidas pelos estudantes. Essas não possuem métodos, apenas atributos que serão manipulados e exibidos pela aplicação. Elas obedecem a tabela 7:

TABELA 7 – CLASSES MODEL

Classe	**Atributo**	**Desc. Atributo**
ClassModel	idCours	Identificador do curos ao qual a aula pertence
	idClass	Identificador único da aula
	classTitle	Título da aula
	date	Data da aula
CoursModel	idCours	Identificador único do curso
	coursName	Nome do curso
	initDate	Data de início do curso
	endDate	Data de encerramento do curso
TaskModel	idCours	Identificador único do curso ao qual a atividade pertence
	idClass	Identificador único da aula ao qual a atividade pertence
	idTask	Identificador único da atividade
	name	Nome da atividade
	description	Descrição da atividade

FONTE: O autor (2017).

Os atributos, nomes e organização das classes seguem o padrão do diagrama de banco de dados descrito na subseção 4.5.

Relacionado ao consumo de serviços *web*, o *framework* Ionic disponibiliza *Providers* que se destinam a disparar processamento em *background*. Para consumir os serviços necessários criou-se os seguintes *providers* (Tabela 8):

TABELA 8 – DESCRIÇÃO PROVIDERS

Provider	**Descrição**
CLASSES-TASK	Consome os serviços referentes às atividades a serem realizadas em sala de aula
COURSES-CLASSES	Consome os serviços referentes às aulas do curso selecionado pelo usuário
LOGIN	Realiza o login do usuário no sistema
PLAYER	Consome os serviços para execução musical.
STUDENT-COURSES	Consome os serviços referentes aos cursos do estudante.

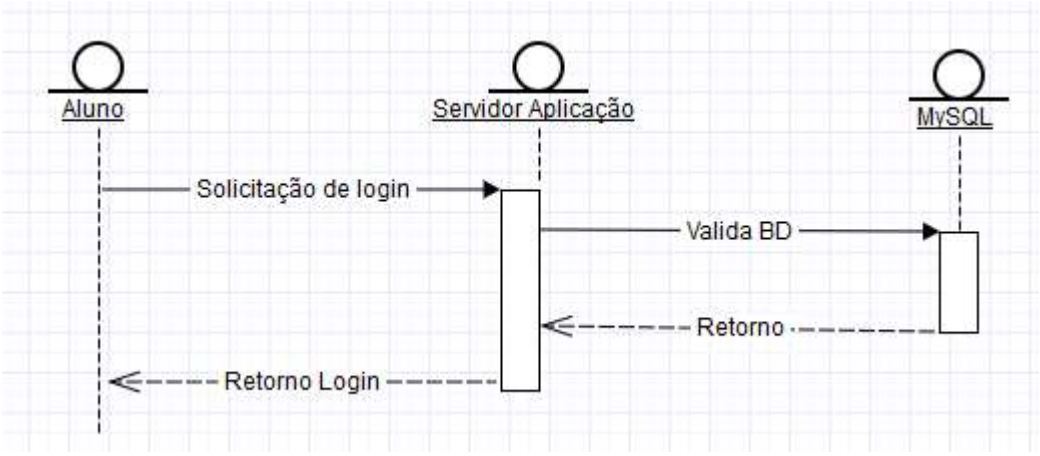
FONTE: O autor (2017).

A primeira parte dessa subseção destina-se a documentar as telas criadas bem como o código desenvolvido para seu gerenciamento.

4.3.1 TELA LOGIN

O processo de login do usuário busca primeiramente no celular se já existe um usuário/senha persistidos no aparelho. Caso exista, esses valores são passados para o Raspberry Pi validar. Em caso de validação positiva o estudante é encaminhado para a tela COURSES. Porém, se a validação falhar retorna-se para a tela LOGIN onde o estudante pode tentar logar manualmente. O diagrama UML de sequência, apresentado na figura 4, descreve este processo.

FIGURA 4 – UML LOGIN DO USUÁRIO



FONTE: O autor (2017).

A tela de login possui o arquivo logon.ts onde se encontra a classe responsável pelo seu processamento. Nessa ocorre a validação do retorno do servidor para a solicitação de login do usuário.

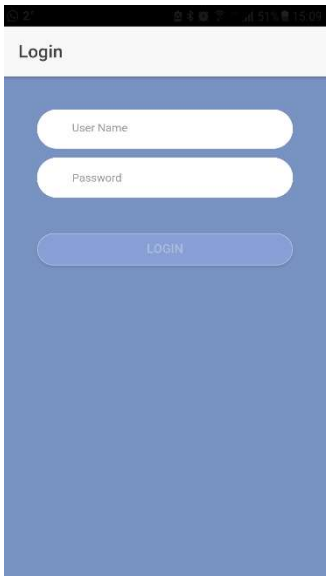
Essa classe criada para a tela de login possui um método. A tabela 9 apresenta uma breve descrição e na figura 5 pode-se ver a tela de login:

TABELA 9 – DESCRIÇÃO MÉTODO DA INTERFACE DE LOGIN

Método	**Descrição**
LOGIN()	Método que chama o serviço de login do usuário. Caso o retorno seja positivo encaminha o estudante para a tela de cursos. Em caso negativo apenas exibe alerta na tela.

FONTE: O autor (2017).

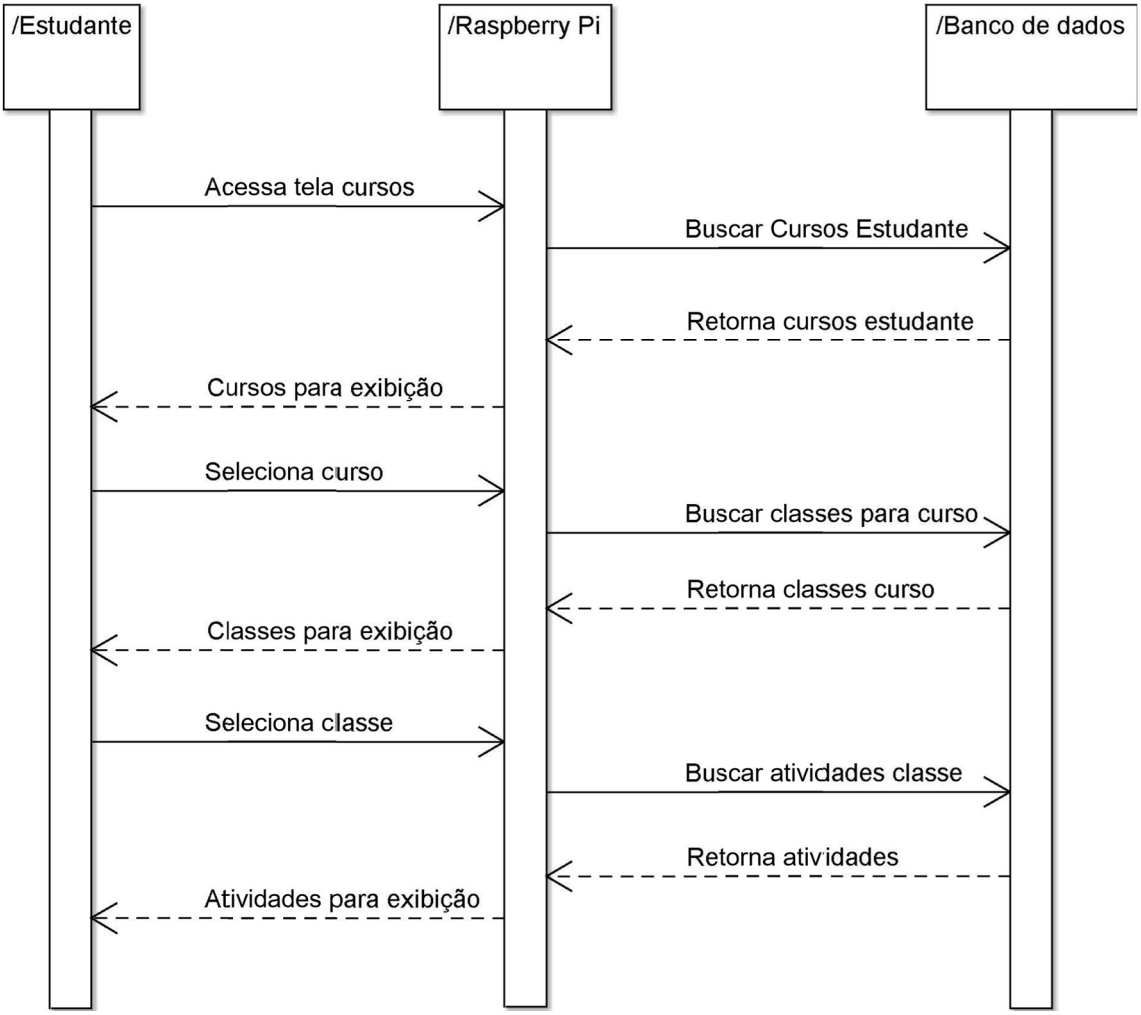
FIGURA 5 – TELA DE LOGIN



FONTE: O autor (2017).

Após acessar com sucesso o sistema, as telas COURSES, CLASSES e TASKS. A figura 6 exibe o diagrama de sequência referente às telas, ações disparadas para o servidor e acessos ao banco de dados.

FIGURA 6 – SEQUÊNCIA DAS TELAS APÓS LOGIN



FONTE: O autor (2017).

Após selecionar uma atividade da aula, abre-se para o estudante a tela KEYBOARD onde está uma série de botões para interação musical dos estudantes.

Voltando à sequência de telas, já se explicou o funcionamento para as telas de LOGIN. Segue-se agora com as demais telas seguindo a ordem COURSES-CLASSES-TASKS-KEYBOARD.

4.3.2 TELA COURSES

Tela onde são exibidos todos os cursos nos quais o estudante está matriculado. A classe responsável por exibir esses cursos é a classe CoursesPage. Essa possui apenas dois métodos, um para carregar as informações dos cursos e outro para processar a interação do estudante na tela e encaminhar o aplicativo para o curso desejado pelo estudante.

Os métodos da classe mencionados estão descritos na tabela 10 e na figura 7 apresenta-se a tela com os cursos:

TABELA 10 – MÉTODOS PARA TELA DE CURSOS

Método	**Descrição**
loadCourses()	Chamado no momento de criação da tela. Busca, em um processamento paralelo, as informações consumindo o serviço '/studentCourses' descrito no subtópico referente aos serviços disponibilizados.
coursSelected(cours:CoursModel)	Chama a tela para exibir as informações das aulas do curso passando como parâmetros o identificador único do curso, nome do curso e identificador do estudante. Recebe como parâmetro um objeto CoursModel descrito na seção dos objetos Model.

FONTE: O autor (2017).

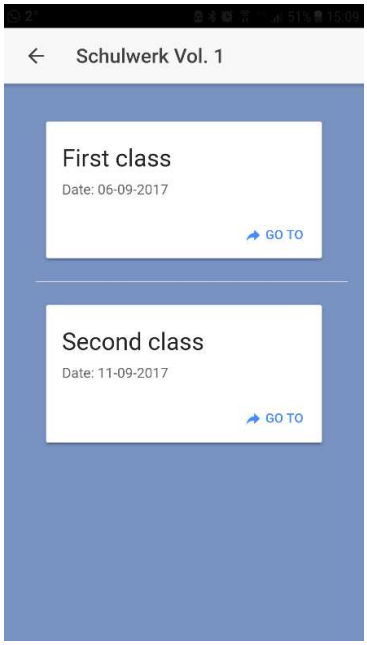
FIGURA 7 – TELA DE CURSOS

FONTE: O autor (2017).

4.3.3 TELA AULAS

Ao selecionar o curso desejado e ser encaminhado para a tela contendo as aulas do curso ao estudante exibe-se uma tela semelhante à figura 8:

FIGURA 8 – TELA COM AULAS



FONTE: O autor (2017).

Nessa tela exibe-se as informações das aulas, essas são obtidas consumindo os serviços disponibilizados pelo Provider CoursesClasses*Providers*. Essas informações são passadas para objetos da classe ClassModel e exibidos. Ao se apertar o botão ‘Go To’ invoca-se o método ‘classSelected’ da classe ClassesPage e o usuário é encaminhado à tela de atividades. A classe ClassesPage, responsável pelo processamento da tela com as aulas e possui os dois métodos descritos na tabela 11:

TABELA 11 – DESCRIÇÃO MÉTODOS PARA INTERFACE DE AULAS

Método	**Descrição**
loadCoursesClasses	Método responsável por invocar os serviços do Provider CoursesClassesProvider e processar seu retorno instanciando objetos da classe ClassModel.
classSelected(classModel: ClassModel)	Método chamado pelo evento de click no botão ‘Go To’ e que encaminha o usuário para a tela com as atividades da aula.

FONTE: O autor (2017).

4.3.4 TELA ATIVIDADES

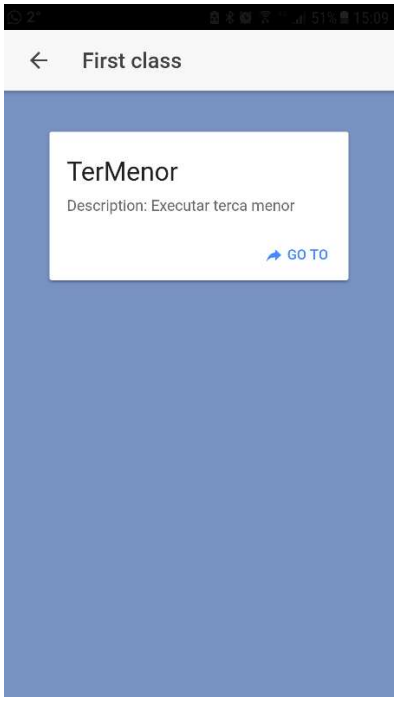
Nessa tela apresentam-se as atividades referentes à aula, tendo como classe responsável pelo seu processamento a classe TaskPage. Essa possui dois métodos também conforme tabela 12. Na Figura 9 apresenta-se a tela de atividades:

TABELA 12 – DESCRIÇÃO MÉTODOS PARA INTERFACE DE ATIVIDADES

Método	**Descrição**
loadClassesTasks	Método responsável por invocar o Provider ClassesTasksProvider e depois instanciar objetos da classe TaskModel para exibição em tela.
callKeyboard(task: TaskModel)	À partir da atividade selecionado pelo usuário invoca-se a tela 'keyboard' com o teclado musical para procução musical por parte dos estudantes.

FONTE: O autor (2017).

FIGURA 9 – INTERFACE DE ATIVIDADES

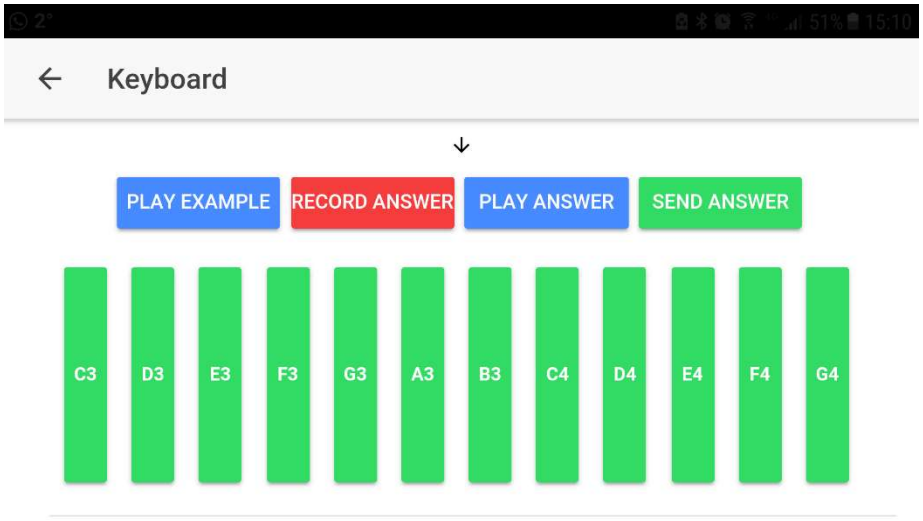


FONTE: O autor (2017).

4.3.5 TELA KEYBOARD

Tela mais importante da aplicação pois é nela que os estudantes podem interagir com a finalidade de se fazer música. A classe responsável por gerencia-la é a KeyboardPage. Figura 10 apresenta a interface do usuário:

FIGURA 10 – TELA COM TECLADO MUSICAL

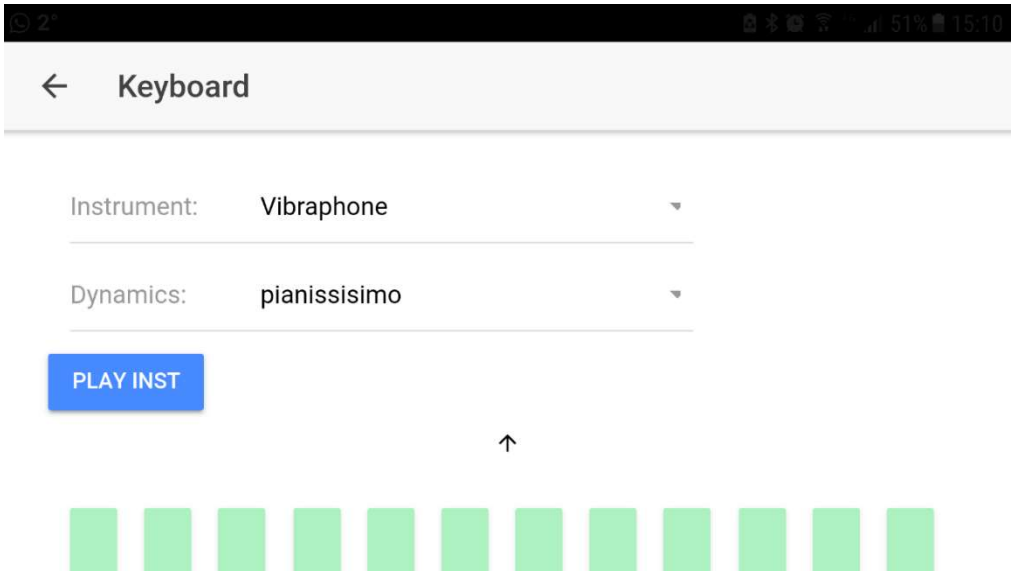


FONTE: O autor (2017).

Para compreender melhor o funcionamento dela, antes de explicar como a classe responsável pelo seu funcionamento foi desenvolvida, dividir-se-á a tela para então explicar a função de cada um dos seus elementos constituintes.

Na parte superior da tela encontra-se o botão com a flecha apontando para baixo (↓), esse tem como função abrir um submenu conforme figura 11:

FIGURA 11 – TELA PARA SELECIONAR INSTRUMENTO E DINÂMICA



FONTE: O autor (2017).

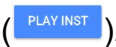
Nesse pode-se escolher o instrumento para execução no primeiro select-options. A determinação desses instrumentos se dá no momento de criação da tarefa e não é possível ao estudante alterar a lista. A ele cabe apenas selecionar um.

Abaixo dos instrumentos existe a possibilidade de escolher dinâmica ou volume de execução do instrumento. A cada dinâmica relacionou-se um código de dinâmica que será passado ao sintetizador MIDI pela aplicação web desenvolvida e descrita adiante no trabalho. Dividiu-se os códigos conforme tabela 13:

TABELA 13 – VALORES PARA CADA DINÂMICA MUSICAL

Código dinâmica	**Descrição musical**
22	*Pianississimo*
37	*Pianissimo*
52	*Piano*
67	*Mezzo piano*
82	*Mezzo forte*
97	*Forte*
112	*Fortissimo*
127	*Fortississimo*

FONTE: O autor (2017).

Abaixo das opções de dinâmica localiza-se o botão '*Play Inst*' (

). Com esse possibilita-se ao estudante ouvir a parte de cada instrumento no exercício musical proposto. Para auxiliar na visualização muda-se a cor da tecla a ser tocada no teclado musical da tela.

Voltando à tela com o teclado musical, pode-se dividi-la em duas partes. Na parte superior existe os botões de ação do estudante conforme a figura 12.

FIGURA 12 – BOTÕES PARA INTERAÇÃO COM RASPBERRY PI

FONTE: O autor (2017).

Esses possuem diferentes funcionalidades descritas na tabela 14:

TABELA 14 – DESCRIÇÃO DA FUNCIONALIDADE DE CADA BOTÃO

Botão	**Funcionalidade**
PLAY EXAMPLE	Envia comando para o servidor de aplicação executar todo o exercício musical proposto. Realiza essa tarefa invocando o método playExercise().
RECORD ANSWER	Permite ao estudante grava sua resposta para o exercício.
PLAY ANSWER	Executa a resposta do estudante.
SEND ANSWER	Envia a resposta do estudante para o servidor de aplicação persisti-la.

FONTE: O autor (2017).

Abaixo desses botões encontra-se o teclado musical onde os estudantes podem interagir e fazer a música desejada conforme mostra a figura 13.

FIGURA 13 – TECLADO MUSICAL

FONTE: O autor (2017).

Nele estão dispostos 12 botões, cada um referente a uma nota musical. Para ordena-los seguiu-se a escala maior da música ocidental tendo como primeira nota o Dó3 indo até o Sol4.

Para permitir ao sistema tocar uma nota, captura-se o evento ‘touchstart’ de qualquer botão da tela. A cada botão está atrelado um código de nota MIDI que será passado via string URL para o servidor de aplicação. A partir do campo ‘Dinâmica’ apresentado na figura 9 e detalhado na tabela 9 escolhe-se a dinâmica musical desejada. O valor selecionado é então passado ao servidor juntamente com o valor do canal do estudante. Esse último está presente no banco de dados e não é alterado em momento algum do processamento. Esse evento ‘touchstart’ invoca o método playNote do *provider* PlayerProvider passando os parâmetros da nota (código MIDI para a nota), código do instrumento definido no campo Instrumento descrito acima, dinâmica conforme explicação anterior e 0 como último parâmetro. Esse valor zero se faz necessário pois é como o servidor de aplicação saberá se ele deve enviar o comando NOTE_ON ou NOTE_OFF para o servidor MIDI.

A seguir um exemplo da String URL enviada pelo aplicativo para o servidor de aplicação:

```
? note=60&inst=20&channel=0&dynamic=120&action=0
```

Onde “note” igual a nota, “inst” o instrument, “channel” o canal MIDI, “dynamic” a dinâmica e “action” a ação a ser tomada pelo servidor de aplicação. No caso citado, o servidor de aplicação passará o comando NOTE_ON no canal 0, instrumento 20, nota 60 e dinâmica 120.

Ao soltar o botão referente a uma nota musical, o aplicativo invoca o método stopNote da classe KeyboardPage. Esse invoca o mesmo método da classe PlayerProvider descrito acima com a única diferença de passar o parâmetro 1 no campo action. Dessa forma o servidor de aplicação saberá que deve enviar o comando NOTE_OFF com os parâmetros.

Para propiciar ao estudante executar cada instrumento individualmente e ainda visualizar a tecla a ser tocada, o programa possui em memória um objeto JSON com as informações acerca das notas e tempo de cada instrumento. O programa então simula a execução de cada nota invocando o método playScript do *provider* PlayerProvider.

Ao clicar no botão ‘*Record Answer*’ a classe inicializa a variável TASKANSWER. Essa variável é um objeto JSON onde se gravará as informações

sobre a resposta do estudante. Esse objeto possui dois objetos JSON internos, a saber, um objeto ‘inst’ onde se encontra o código do instrumento que o estudante selecionou e outro objeto ‘notes’ onde se encontram as informações sobre as notas (tempo de execução, dinâmica e nota) tocadas pelos estudantes. As notas nesse objeto são gravadas na sequência que o estudante tocou.

Quando o estudante clicar em ‘*Play Answer*’ o sistema buscará as informações contidas na variável TASKANSWER (a mesma alterada pelo botão ‘*Record Answer*’). Essa variável poderá estar vazia, caso o estudante nunca tenha enviado ou gravado uma resposta. Outra opção é que nela esteja contida uma resposta enviada anteriormente pelo estudante e persistida na base de dados. A última opção é nela estar contida a última resposta gravada pelo estudante. Por padrão o sistema sempre busca a resposta enviada pelo estudante anteriormente. Essa busca se dá no momento em que as informações sobre a atividade é carregada pelo aplicativo.

A resposta do estudante é enviada ao clicar no botão ‘*Send Answer*’. Esse busca o valor de TASKANSWER e envia como string para o servidor de aplicação.

Essas telas descritas constituem a parte visível da aplicação para os estudantes. Porém para consumir os serviços necessários para o funcionamento do sistema existem as classes Provider. Essas são descritas individualmente a seguir. Importante mencionar que essas classes consomem os serviços descritos adiante no trabalho, porém nenhuma delas formata a resposta do servidor para instanciar objetos das classes Model.

4.3.6 CLASSESTASKSPROVIDER

Provider invocado pela tela TaskPage e KeyboardPage. Para a tela onde se exibe as atividades de cada aula invoca-se o método loadClassesTasks. Essa consome o serviço classesTask passando o identificador do curso e da aula.

Os outros dois métodos desse *provider*, loadTaskInfo e sendAnswer, chama-se a partir da classe KeyboardPage responsável pelo processamento da tela com o teclado musical. No primeiro método busca-se os detalhes da atividade através do serviço loadTaskInfo passando o identificador do curso, da aula e da atividade. Para

o segundo método consome-se o serviço sendAnswer passando as mesmas informações, porém adicionando o identificador do estudante, a resposta por ele gravada e o local com as informações do idioma. Essa última informação faz-se necessária para o servidor de aplicação retornar mensagem no idioma do logon do estudante.

4.3.7 CoursesClassesProvider

Busca a lista de aulas de um curso. Realiza essa tarefa consumindo o serviço coursesClasses, informando o identificador do curso.

4.3.8 LoginProvider

Chamado pela tela LOGIN.TS, esse *provider* consome o serviço de LOGIN desenvolvido no servidor de aplicação. Nesse momento não se valida o retorno do servidor, se o usuário está logado ou não. Essa validação ocorre na tela ao analisar o retorno do servidor.

4.3.9 PlayerProvider

Provider mais importante da aplicação porque é nele que ocorrem as interações dos estudantes para emissão de sons por parte do Raspberry Pi. A relação do método com o serviço consumido segue a tabela 14:

TABELA 15 – DESCRIÇÃO PLAYERPROVIDER

Método	**Serviço**	**Descrição**
playExercise	playTask	Executa o exercício proposto tocando todos os instrumentos.
playNote	Play	Envia comando para o Raspberry iniciar ou encerrar a execução de uma nota
playScript	playScript	Executa Script com as notas individuais de cada instrumento na atividade proposta

FONTE: O autor (2017).

4.3.10 STUDENTCOURSESPROVIDER

Provider que consome o serviço studentCourses informando apenas o identificador do estudante.

Para uma análise mais aprofundada do código desenvolvido recomenda-se acessar o repositório criado no site GitHub[7].

4.4 DOCUMENTAÇÃO SERVIDOR DE APLICAÇÃO

A aplicação móvel descrita acima não emite som a partir do dispositivo móvel, mas envia comandos para o Raspberry Pi interpretar e então gerar o som. Para executar essa tarefa criou-se três serviços em Python. De maneira análoga, as informações sobre o curso, aulas, atividades e estudantes também se encontram

[7] https://github.com/Jongui/MusicPlayerIonic

persistidas no Raspberry Pi. Essas são acessadas através de 7 serviços desenvolvidos. Nesta seção a apresentação desses 10 serviços divide-se em descrever os serviços destinados a produção musical para depois documentar os serviços relacionados à interação com os dados.

Para permitir a integração dos serviços com a instância do banco de dados criado, desenvolveu-se quatro classes, sendo uma para cada tabela do banco de dados. Cada uma dessas classes possui um método construtor e um outro método chamado ‘to_json’ que transforma o objeto em um objeto JSON. Essas classes estão em arquivos “.py” no Raspberry Pi e em cada um desses arquivos criou-se também uma classe DAO (Data Access Object) para fazer a busca propriamente dita no banco de dados.

Outra função da aplicação se configura em enviar as mensagens MIDI para o servidor MIDI. Essas mensagens são passadas através da classe Player. Descrições mais detalhadas encontram-se a seguir.

4.4.1 SERVIÇOS MUSICAIS

Por Serviços Musicais entende-se nesse trabalho aqueles onde uma instância da classe Player é utilizada. No presente trabalho três são esses serviços e a descrição deles segue.

- play: Serviço que propicia a execução ou parada de uma nota musical definida. Isso se torna possível pois o método play da classe Player é chamado em um processamento paralelo (Thread). Dessa forma permite-se que o aplicativo fique disponível para interação por parte do usuário. A tabela 16 descreve cada um dos parâmetros utilizados.

TABELA 16 - MAPEAMENTO DOS PARÂMETROS DO SERVIÇO PLAY

Parâmetro	**Descrição**
Time	Tempo de execução da nota medido em segundos.
Note	Código da nota de acordo com a especificação MIDI.
Inst	Código de instrumento de acordo com a especificação MIDI
Channel	Canal para execução da nota no servidor FluidSynth.
Action	Ação a ser tomada pela aplicação. 0 – iniciar execução da nota; 1 – encerrar execução da nota.
Dynamic	Dinâmica (volume) da execução da nota de acordo com a especificação MIDI.

FONTE: O autor (2017).

- playScript: Esse serviço executa no servidor o comando musical enviado pelo aparelho quando o usuário tenta executar a música que é a tarefa da aula ou quando executa sua resposta à tarefa da aula. Nessa situação o método da classe Player responsável por executar a nota não é chamado em processamento paralelo. Decidiu-se por essa abordagem pois dessa forma possibilita-se a alteração da cor da tecla no aplicativo além da execução da nota ser igual ao tempo estabelecido pela atividade.

TABELA 17 – MAPEAMENTO PARÂMETROS DO SERVIÇO PLAYSCRIPT

Parâmetro	**Descrição**
Time	Tempo de execução da nota medido em segundos.
Note	Código da nota de acordo com a especificação MIDI.
Inst	Código de instrumento de acordo com a especificação MIDI
Channel	Canal para execução da nota no servidor FluidSynth.
Dynamic	Dinâmica (volume) da execução da nota de acordo com a especificação MIDI.

FONTE: O autor (2017).

- playerTask: Esse serviço tem como parâmetro de entrada o nome do arquivo contendo o exercício desejado para o método play_task da classe Player.

TABELA 18 – MAPEAMENTO PARÂMETROS DO SERVIÇO PLAYTASK

Parâmetro	**Descrição**
fileName	Nome do arquivo a ser executado.

FONTE: O autor (2017).

- Serviços integração banco de dados

Para a interação com o banco de dados criou-se sete serviços a serem consumidos pelo aplicativo móvel. A seguir a descrição e parâmetros para cada um deles.

- login: Serviço chamado para realizar o login do usuário.

TABELA 19 – MAPEAMENTO PARÂMETROS DO SERVIÇO LOGIN

Parâmetro	**Descrição**
idStudent	Identificador único do usuário.
Password	Senha do usuário
Locale	Localização do usuário seguindo o padrão código do idioma – código do país

FONTE: O autor (2017).

- studentCourses: A partir do identificador do usuário, esse serviço busca os cursos nos quais o estudante está matriculado.

TABELA 20 - MAPEAMENTO PARÂMETROS DO SERVIÇO STUDENTCOURSES

Parâmetro	**Descrição**
idStudent	Identificador único do usuário

FONTE: O autor (2017).

- coursClasses: A partir do identificador do curso esse serviço busca as aulas relacionadas a ele.

TABELA 21 - MAPEAMENTO PARÂMETROS DO SERVIÇO COURSCLASSES

Parâmetro	**Descrição**
idCours	Identificador único do curso

FONTE: O autor (2017).

- classesTasks: A partir da classe e do curso solicitado esse serviço retornar as informações sobre tarefas relacionadas.

TABELA 22 – MAPEAMENTO PARÂMETROS DO SERVIÇO CLASSESTASKS

Parâmetro	**Descrição**
idClasses	Identificador único da aula
idCours	Identificador único do curso

FONTE: O autor (2017).

- loadTaskInfo: Esse serviço busca as atividades a serem realizadas pelos estudantes de acordo com o curso, aula e tarefa.

TABELA 23 – MAPEAMENTO PARÂMETROS DO SERVIÇO LOADTASKINFO

Parâmetro	**Descrição**
idClasses	Identificador único da aula
idCours	Identificador único do curso
idTask	Identificador da tarefa

FONTE: O autor (2017).

- sendAnswer: Esse serviço destina-se a salvar as respostas dos estudantes para as atividades propostas.

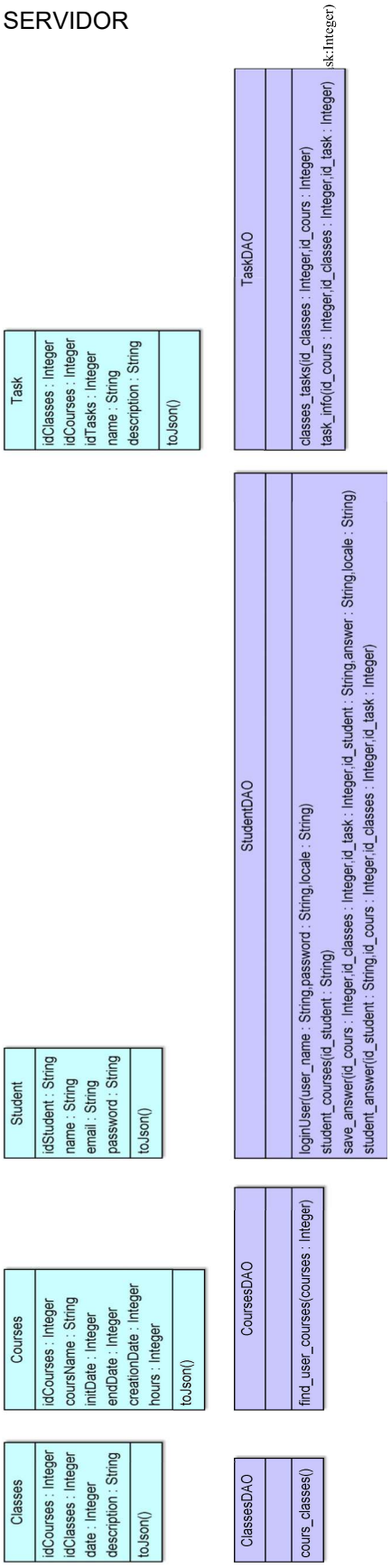
TABELA 24 – MAPEAMENTO PARÂMETROS DO SERVIÇO SENDANSWER

Parâmetro	**Descrição**
idClasses	Identificador único da aula
idCours	Identificador único do curso
idTask	Identificador da tarefa
idStudent	Identificador do estudante
Locale	Localização do estudante
Answer	Resposta do estudante

FONTE: O autor (2017).

Conforme mencionado anteriormente, no servidor de aplicação criou-se quatro classes para mapear as tabelas do banco de dados para objetos em tempo de execução. Essas classes possuem um objeto DAO (Data Access Object) atrelado a cada uma. Essas últimas realizam o acesso ao banco de dados propriamente dito. A seguir está a descrição das quatro classes. Figura 14 exibe o diagrama de classes referente a essas.

FIGURA 14 – DIAGRAMA CLASSES SERVIDOR



FONTE: O autor (2017).

- classes.py: Classe onde se processam as aulas de cada curso. Para o acesso ao banco de dados criou-se a classe classesDAO.

Classe: Classes

TABELA 25 – ATRIBUTOS CLASSE CLASSES

Nome	**Descrição**
idCourses	Identificador do curso ao qual a aula pertence.
idClasses	Identificador da aula
Date	Data da aula
Description	Descrição da aula

FONTE: O autor (2017).

- courses.py: Arquivo onde se codificou a classe Courses. Essa possui como tabela de referência a tabela Courses. Para o acesso ao banco criou-se a classe CoursesDAO.

Classe: Courses

Tabela 26 abaixo apresenta os atributos dessa classe.

TABELA 26 – ATRIBUTOS CLASSE COURSES

Nome	**Descrição**
idCourses	Identificador do curso
coursName	Nome do curso
initDate	Data inicial do curso
endDate	Data final do curso
creationDate	Data de criação do curso
Hours	Horas totais do curso

FONTE: O autor (2017).

- students-py: Arquivo onde estão as classes Student e StudentDAO. Na tabela 27 apresentada a seguir encontram-se os atributos e suas respectivas descrições.

TABELA 27 – ATRIBUTOS CLASSE STUDENT

Nome	**Descrição**
idStudent	Identificador do estudante
Name	Nome do estudante
Email	E-mail do estudante
Password	Senha do estudante

FONTE: O autor (2017).

- task.py: Classe para manipular as atividades propostas para a aula. Possui os seguintes atributos:

TABELA 28 – ATRIBUTOS CLASSE TASK

Nome	**Descrição**
idCourses	Identificador do curso
idClasses	Identificador da aula
idTasks	Identificador da atividade
Name	Nome da atividade
Description	Descrição da atividade
Keyboard	Teclado musical utilizado pela atividade
Participants	Número de estudantes na aula

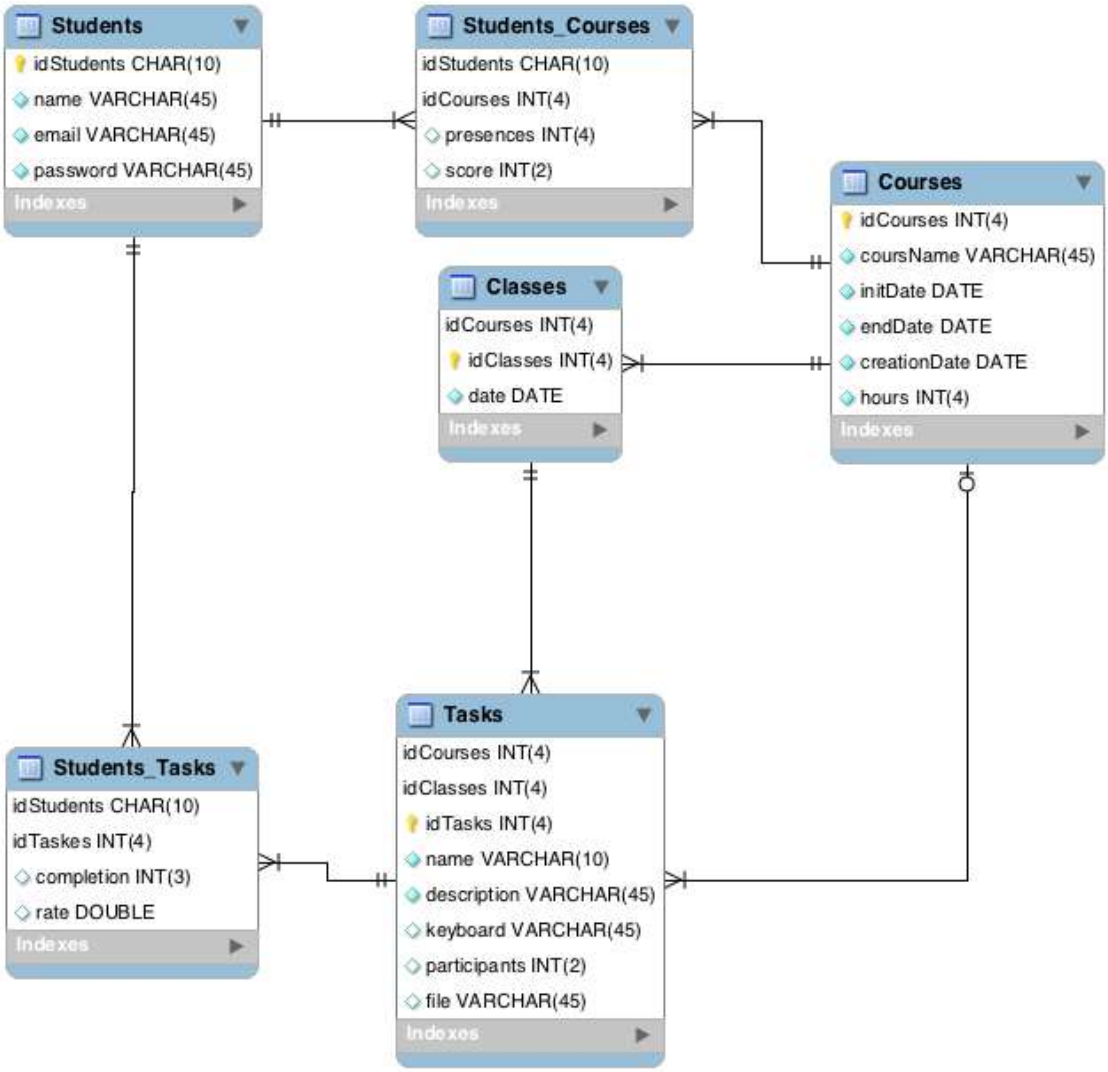
FONTE: O autor (2017).

Além dessas classes já apresentadas, criou-se outras duas utilizadas pelo o sistema. A classe Player tem por função interagir com o servidor MIDI e produzir os sons desejados pelos usuários. A outra é a classe Resource destinada a buscar o arquivo de textos com o idioma do usuário informado pelo serviço.

4.5 BANCO DE DADOS

A persistência das informações do sistema emprega uma instância do banco de dados MySQL no Rasperry Pi. Esse banco de dados então seguiu a modelagem da figura 15:

FIGURA 15 – MODELO DO BANCO DE DADOS EMPREGADO



FONTE: O autor (2017).

Dessa modelagem tem-se que as tabelas Students, Courses, Classes e Tasks são as principais e por isso possuem uma classe na aplicação no servidor. As demais são classes necessárias pois a ligação entre as tabelas é de muitos para muitos.

5. AVALIAÇÃO DO SISTEMA JONGUI

Três voluntários se disponibilizaram para realizar os testes e responder ao questionário sobre o sistema. Os aparelhos utilizados para os testes foram um smartphone Samsung S7 com sistema operacional Android 7, um Motorola MotoG 2ª geração com Android 6.0 e um Tablet Galaxy Note Pro 12.2 com Android 4.4.2. A seguir apresenta-se as respostas para cada uma das questões. A tabela 29 explicita melhor as informações sobre os voluntários.

TABELA 29: PARTICIPANTES DOS TESTES

Participante	**Aparelho**	**Sistema operacional**
Participante 1(P1)	Motorola MotoG 2ª Ger.	Android 6.0
Participante 2(P2)	Samsung S7	Android 7.0
Participante 3(P3)	Galaxy NotePro 12.2	Android 4.4.2

FONTE: O autor (2017).

Desses três participantes, P1 e P2 possui curso superior de Música e experiência como instrumentistas. Além dessa formação, P2 leciona música para crianças utilizando o método Orff. P3 por sua vez possui apenas formação básica de Música, não tendo tido experiência nem ensino mais avançado em Música.

5.1 QUESTÃO 1

A primeira questão consistiu de uma pergunta de múltipla escolha onde cada participante informou seu nível de conhecimento musical. Participante 1 e Participantes 2 informaram ter conhecimentos avançados de música enquanto Participante 3 assinalou conhecimento intermediário.

5.2 QUESTÃO 2

Outra questão de múltipla escolha onde se solicitava a impressão do usuário sobre quão responsivo o sistema era. Para P2 e P3, a impressão foi ‘B Com um pouco de retardo, mas quase imperceptível’. Apenas P1 indicou alternativa ‘C Com um pouco de retardo e muito perceptível’.

5.3 QUESTÃO 3

Para essa questão todos indicaram que entenderam o exercício após a primeira explicação.

5.4 QUESTÃO 4

Nessa pergunta buscou-se perceber se houve problemas ao executar o aplicativo e solicitou-se uma breve descrição dele. Todos comentaram terem problemas com o tempo de resposta do sistema. Para P3, as teclas do teclado musical eram pequenas demais e para P1 apenas quando os toques eram mais prolongados o sistema respondia. A causa para isso não foi identificada, podendo ser um problema do sistema Jongui ou do sistema de reconhecimento de toques do smartphone. P1 teve algumas situações problemáticas ao longo do experimento, por muitas vezes o grupo inteiro parava com a atividade pois ele não conseguia realizar sua parte da tarefa.

5.5 QUESTÃO 5

Sobre o método colaborativo todos voluntários aprovaram. Porém, P2 poderia haver uma diferenciação maior entre os instrumentos musicais. Dessa forma a interação entre os alunos se daria de maneira mais musical.

5.6 QUESTÃO 6

Sobre a aprovação do sistema, P1 e P3 aprovaram parcialmente e P2 totalmente.

5.7 QUESTÃO 7

Ainda sobre o aprendizado colaborativo, todos voluntários concordaram totalmente que o sistema possibilita esse aprendizado.

5.8 QUESTÃO 8

Todos voluntários utilizariam o sistema para ensino de música, diferenciando-se os motivos de cada um. Para P3 o sistema permite o aprendizado coletivo a um preço acessível. P2 afirma que o sistema permite aos alunos entrarem no mundo da música utilizando seus próprios dispositivos. Já para P1 o sistema ainda precisa de algumas melhorias com relação ao tempo de resposta, porém o utilizaria. Para esse participante deve-se buscar outras maneiras mais eficientes e rápidas de conexão entre o aplicativo e o Raspberry Pi.

5.9 QUESTÃO 9

Para todos os voluntários a aceitação foi positiva, porém todos mencionaram a necessidade de melhoria no que se refere ao tempo de resposta do sistema às interações dos usuários com o aplicativo.

5.10 OBSERVAÇÕES

Juntamente com as respostas ao questionário observou-se o teste e abaixo descreve-se alguns pontos considerados relevantes:

- Durante os testes observou-se que o conhecimento musical dos participantes influenciou a maneira como eles interpretaram a tarefa. Ao lerem que deveriam criar uma resposta ao tema musical proposto, os participantes compuseram coletivamente alguns compassos musicais ao invés de repetirem o tema proposto. Isso se deve ao fato de que na música uma resposta a um tema é um outro trecho musical, gerando assim uma pergunta e resposta musical.
- Observou-se também que os participantes tentaram algumas vezes gravar suas respostas para envio no sistema. Porém, muitas vezes as demoras nas respostas do sistema às interações dos usuários impossibilitaram a finalização da tarefa com sucesso.
- O processo de composição coletiva flui de maneira tranquila e natural para os participantes.
- Como o sistema por vezes demorava para emitir o som da nota tocada pelo usuário, isso causava frustrações nos participantes pois eles não conseguiam executar o trecho musical como eles haviam pensado.
- P1 e P2 comentaram em mostrar o sistema para algum organista, visto que os órgãos musicais também possuem um atraso entre ação do instrumentista e som emitido.
- Durante a última tarefa os participantes se preocuparam mais em entender seu próprio aplicativo e como interagir com ele do que em fazerem música com o sistema.

6. CONCLUSÃO

A seguir apresentam-se as conclusões do presente trabalho. Após a explanação das conclusões referentes aos tópicos envolvidos neste TCC, explicar-se-á as conclusões a partir dos objetivos propostos.

Como metodologia norteadora para o desenvolvimento do sistema, o *Orff Schulwerk* mostrou-se adequado. Seus princípios de aprendizagem colaborativa e graduação de conteúdos mais simples até mais complexos possibilitaram desenvolver um sistema mais claro. Entretanto, o presente sistema não possibilita a substituição de todo o instrumental Orff. A razão para tal está no próprio Schulwerk, pois esse possui como parte integrante o uso da voz e do corpo para o ensino de música. O presente trabalho destina-se apenas para substituir alguns instrumentos, notadamente os de percussão com altura definida.

Para efetuar essa substituição empregou-se o BYOD, um princípio para possibilitar a integração da tecnologia móvel no ensino. Ao permitir que o estudante utilize seu aparelho possibilita que esse interaja em sala de aula com uma ferramenta mais familiar. Porém, assim como ocorre com o uso de computadores gerais para instrumentos digitais apresentado no capítulo 2, as constantes atualizações de sistemas operacionais, dos demais softwares e hardwares tornam sua implementação mais complexas. Isso porque se deve controlar atualizações e adaptar os aparelhos existentes a essas atualizações. Bem como os próprios estudantes podem trocar seus aparelhos e exigir algum tipo de desenvolvimento ou adaptação do sistema para esse.

No presente trabalho o servidor de MIDI (FluidSynth), servidor de aplicação e servidor de banco de dados encontram-se no mesmo aparelho. Isso pode gerar conflitos entre o sistema operacional e as necessidades de execução em tempo real. Dessa forma não se atingiu o objetivo proposto por Franco e Wanderley (2016) ‘que os aspectos computacionais do instrumento deveriam se restringir a operações relacionadas a funções musicais e não interações no nível de sistema operacional’ (FRANCO e WANDERLEY, 2016). Durante alguns testes a resposta do Raspberry Pi para a interação com o usuário foram lentas. Porém, na maioria dos casos a execução ocorreu imediatamente após a interação do usuário.

Com relação ao MIDI, provou-se uma maneira fácil e prática de se desenvolver para a criação musical. Sua biblioteca Python possibilitou um desenvolvimento rápido e um código claro para análise. Porém o teste mostrou a necessidade de uma maior diferenciação entre os sons dos instrumentos. Isso se possibilita encontrando outras fontes de som MIDI ou com outras maneiras de se sintetizar os sons digitalmente.

A conclusão que se chega para o Python é a de uma linguagem de programação que possibilita uma execução rápida e fácil entendimento do código. Porém, também possui o problema de atualizações e manutenções por elas necessárias. Para esse trabalho utilizou-se Python2 porque as bibliotecas para criação dos serviços Web, conexão com banco de dados e conexão com o servidor MIDI estão em estágio maduro. Mas a própria *Python Software Foundation* liberou uma nota avisando que Python2 será descontinuado em 2020(PYTHON, 2017). Dessa forma já se faz necessária uma migração do servidor de aplicação para o Python3 ou outra tecnologia disponível.

O Desenvolvimento Multiplataforma provou-se um princípio interessante quando se busca um desenvolvimento único como no caso desse trabalho. Conseguiu-se criar, utilizando Ionic, um aplicativo para aparelhos com sistema operacional Android e sistema operacional iOS. Porém problemas podem surgir e comprometer o projeto como um todo. Nesse trabalho, ao compilar e testar o aplicativo no emulador do iOS não se conseguiu fazer a conexão com o servidor através de uma chamada GET, protocolo HTTP. Diversas buscas foram feitas e tentativas de corrigir o erro também, porém nenhum com o sucesso desejado.

Apesar de o instrumento musical criado com o Raspberry Pi ser de fácil instalação, ele ainda não é um instrumento exclusivamente dedicado. Isso significa dizer que o sistema foi montado em cima do sistema operacional Raspian e não se alterou nenhuma funcionalidade nesse. Isso pode gerar conflitos entre as tarefas realizadas ordinariamente pelo sistema operacional e as interações musicais dos estudantes. Portanto, para se contornar esse problema, evitar-se transtornos e permitir respostas em tempo real do sistema deve-se analisar esse problema. Dessa forma visa-se atingir um nível de resposta em tempo real que se assemelhe à resposta do instrumento musical real.

Um aplicativo para o ensino de música deve possuir um embasamento teórico musical vindo da Educação Musical. Nesse trabalho utilizou-se Orff, porém outros também podem ser implementados. A razão para tal é porque o aplicativo móvel propicia que diversas informações sejam apresentadas aos estudantes que em um instrumento tradicional não seriam.

Outro aspecto importante para um bom sistema constitui-se em criar um sistema com resposta em tempo real. Isso tornaria o aparelho celular em um objeto análogo a um instrumento musical.

Utilizar os princípios do *Orff-Schulwerk* possibilitou desenvolver-se um sistema pequeno e prático para controle de atividades. Criou-se apenas seis tabelas no servidor de banco de dados e sua conexão através dos módulos Python.

Os serviços web criados com a linguagem Python possibilitaram uma rápida interação entre o usuário e o servidor MIDI. Apesar de, por vezes, a interação não se deu em tempo real, o sistema apresentou uma resposta satisfatória da perspectiva dos participantes. Isso percebeu-se com os testes realizados com voluntários. Para esses o atraso entre a ação do usuário e a resposta do sistema não configura um problema em si, desde que ele seja constante. Durante os testes os participantes com conhecimentos avançados em música citaram o exemplo dos órgãos musicais instalados em catedrais onde existe uma demora entre o músico apertar uma tecla do instrumento e a resposta sonora. Com isso conclui-se que mais importante que o tempo absoluto de resposta do sistema é esse tempo ser constante e permitir aos usuários controlarem suas interações o considerando.

O presente sistema propicia aos estudantes de trazerem seus próprios dispositivos para a sala de aula e interagirem uns com os outros. Também possibilita o aprendizado colaborativo no que se refere à música. Conseguiu-se implementar conceitos diversos para um sistema pensado para utilização em sala de aula, porém também se descobriu diversos pontos a serem melhorados para criar uma implementação comercial. Um aspecto importante detectado com o teste está em que o participante com o aparelho celular com menor capacidade de processamento teve mais dificuldades que aqueles com aparelhos de melhor qualidade.

Para trabalhos futuros buscar-se-á desenvolver a interface gráfica do aplicativo móvel para torna-lo mais semelhante a um instrumento musical no mundo real, bem

como possibilitar uma execução em tempo real a todo o momento. Outro aspecto importante para o bom funcionamento do sistema é a parte que se encontra no Raspberry Pi. O *framework* Prynth, brevemente mencionado nesse trabalho, oferece uma alternativa para uma execução em tempo real dos instrumentos musicais. Porém ele torna o Raspberry Pi em um dispositivo totalmente dedicado para a execução de música, excluindo dessa forma as partes do presente sistema relacionadas ao controle das atividades, aulas e cursos aos quais os estudantes pertencem.

7. REFERÊNCIAS

ANATEL. **Brasil tem 236,2 milhões de linhas móveis em janeiro de 2018**. Disponível em: http://www.anatel.gov.br/dados/destaque-1/283-telefonia-movel-registra-aumento-de-156-155-linhas-em-agosto. Acesso em: 12/12/2017.

ATOM Documentation. 2017. Disponível em <http://flight-manual.atom.io/getting-started/sections/why-atom/>. Acesso em: 12/12/2017.

CASTRO, A. MENEZES, C. Aprendizagem colaborativa com suporte computacional. Rio de Janeiro, Brasil, 2011.

FLUIDSYNTH Manuel. 2017. Disponível em <https://github.com/FluidSynth/fluidsynth/wiki/UserManual>. Acesso em: 07/11/2017

FRANCO, I. WANDERLEY, M. (2016). Bridging People and Sound. *Prynth: A Framework for Self-contained Digital Music Instruments*, p. 357-370.

IONIC Documentation. 2017. Disponível em: <https://ionicframework.com/docs/>. Acesso em: 12/12/2017

JSON Documentation. 2017. Disponível em <https://www.json.org/>. Acesso em 21/11/2017.

KUGLER, M. - **Orff-Schulwerk - Elementare Musik- und Tanzpädagogik** - 2017 - http://www.orff.de/de/orff-schulwerk.html - acesso em: 01/11/2017.

MCLEAN, K. J. **The Implementation of Bring Your Own Device (BYOD) in Primary Schools**, *Frontier in Psychology*, Ballart, 15 nov. 2016.

MEIRELES, F. S.- **Pesquisa Anual do Uso de TI**. Disponível em: http://eaesp.fgv.br/sites/eaesp.fgv.br/files/pesti2017gvciappt.pdf. Acesso em 12/12/2017

MIDIORG Documentation. 2017. Disponível em: <https://www.midi.org/articles/about-midi-part-1-overview>. Acesso em 07/11/2017.

MYSQL Documentation. 2017. Disponível em: <https://dev.mysql.com/doc/refman/5.7/en/>. Acesso em 20/11/2017.

ORFF, C. KEETMAN, G – **Music for children Volume 1**, Londes: Schott Music Ltd, 1958.

PREECE, J. ROGERS, Y. SHARP, H. **Design de interação: além da interação homem-computador**. Porto Alegre, RS: Bookman, 2005. 548 p.

PRYNTH. 2017. Disponível em <https://prynth.github.io/>. Acesso em 22/12/2017.

PYGAME Documentation. 2017. Disponível em <http://www.pygame.org/docs>. Acesso em 12/12/2017.

PYTHON Documentation. 2017. Disponível em <https://www.python.org/>. Acesso em 22/12/2017.

RASPBERRY PI Documentation. 2017. Disponível em < https://www.raspberrypi.org/documentation/ >. Acesso em: 01/12/2017

RASPBIAN Documentation. 2017. Disponível em <http://raspbian.org/>. Acesso em: 21/11/2017

TREXLER, J. **Current State of Mobile Learning**, *International Review on Research in Open and Distance Learning*, Wolverhampton, v. 8 n. 2, jun. 2007.

WEBPY Documentation. 2017. Disponível em <http://webpy.org/>. Acesso 20/11/2017

XANTHOPOULOS, S. XINOGALOS, S. **A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications**, Tessalônica, set. 2013.

XCODE Documentation. 2017. Disponível em <https://developer.apple.com/xcode/>. Acesso em: 12/12/2017.

8. APÊNDICE I

Abaixo segue o Termo de Consentimento Livre e Esclarecido assinado pelos voluntários do teste.

Universidade Tecnológica Federal do Paraná - Curitiba

Departamento Acadêmico de Informática

Termo de Consentimento Livre e Esclarecido

Eu _____, portador do RG número _____, estou sendo convidado a participar de um estudo denominado **Sistema Jongui: Tecnologias integradas para o ensino colaborativo de música**, cujos objetivos e justificativas são: desenvolvimento de um sistema integrando um aplicativo móvel e um Raspberry Pi para permitir o ensino de música através deles.

A minha participação no referido estudo será no sentido de realizar tarefas pré-estabelecidas para testar funcionalidades desse sistema. Esses testes ocorrerão com a participação de mais voluntários para permitir a interação de mais pessoas com o sistema de maneira simultânea. Após a execução das tarefas, responderei um questionário onde expressarei impressões pessoais sobre o sistema.

Estou ciente de que minha privacidade será respeitada, ou seja, meu nome ou qualquer outro dado ou elemento que possa, de qualquer forma, me identificar, será mantido em sigilo.

Também fui informado de que posso me recusar a participar do estudo, ou retirar meu consentimento a qualquer momento, sem precisar justificar.

Os pesquisadores envolvidos com o referido projeto são João Guilherme Dyck e Prof. Dr. Leonelo Dell Anhol Almeida e com eles poderei manter contato pelos endereços eletrônicos joaogd53@gmail.com e leonelo@dainf.ct.utfpr.edu.br

É garantido a mim o livre acesso a todas as informações e esclarecimentos adicionais sobre o estudo e suas consequências, enfim, tudo o que eu queira saber antes, durante e depois da minha participação.

Enfim, tendo sido orientado quanto ao teor de todo o aqui mencionado e compreendido a natureza e o objetivo do já referido estudo, manifesto meu livre consentimento em participar, estando totalmente ciente de que não há nenhum valor econômico, a receber ou a pagar, por minha participação.

Curitiba, _____ de _____ de 2018.

_____

Assinatura do participante ou responsável

_____

João Guilherme Dyck

9. APÊNDICE II

Após os testes feitos pelos voluntários pediu-se para que cada um respondesse ao questionário abaixo:

Questionário

1) Levando-se em consideração ensino formal e aprendizado informal, qual o seu nível de conhecimento musical? (Assinale apenas uma alternativa)

a) Nenhum

b) Básico (1 a 2 anos de prática musical)

c) Intermediário (3 a 4 anos de prática musical)

d) Avançado (mais de 5 anos de prática musical)

2) Ao apertar um botão do teclado musical, sua percepção era que a nota era tocada: (Assinale apenas uma alternativa)

a) Imediatamente

b) Com um pouco de retardo, mas quase imperceptível

c) Com um pouco de retardo e muito perceptível

d) Com muito retardo.

3) Com relação à atividade musical proposta você: (Assinale apenas uma alternativa)

a) Entendeu o que deveria ser feito após a primeira explicação

b) Entendeu o que deveria ser feito após algumas explicações

c) Entendeu o que deveria ser feito após algumas explicações e interações com os demais voluntários

d) Não entendeu o que deveria ser feito em momento algum

4) Você teve problema(s) técnico ao usar o aplicativo? Escreva uma frase para cada problema.

_____

_____

_____

5) Dê sua opinião sobre o método colaborativo empregado no sistema.

_____

_____

_____

_____

6) Qual seu nível de aprovação do sistema? (Assinale apenas uma alternativa)

a) Reprovo totalmente.

b) Reprovo parcialmente.

c) Aprovo parcialmente.

d) Aprovo totalmente.

7) ‘O sistema permite o aprendizado colaborativo de música’. Sobre essa frase você:

a) Discorda totalmente.

b) Discorda parcialmente.

c) Condorda parcialmente.

d) Condorda totalmente.

8) Você usaria o sistema para ensinar e aprender música? Por quê?

_____

_____

_____

_____

9) Comente brevemente suas impressões sobre o sistema.

_____

_____

_____

_____