UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

MAIKEU PAULINO DUARTE

UTILIZAÇÃO DO CONCEITO DE INTERNET DAS COISAS ASSOCIADA À TECNOLOGIA DE IBEACONS COMO FERRAMENTA COMPUTACIONAL APLICADA À INSTRUÇÃO DE USUÁRIOS NA UTILIZAÇÃO DE APARELHOS DE GINÁSTICA

CRICIÚMA

2018

MAIKEU PAULINO DUARTE

UTILIZAÇÃO DO CONCEITO DE INTERNET DAS COISAS ASSOCIADA À TECNOLOGIA DE IBEACONS COMO FERRAMENTA COMPUTACIONAL APLICADA À INSTRUÇÃO DE USUÁRIOS NA UTILIZAÇÃO DE APARELHOS DE GINÁSTICA

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel, no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Gustavo Bisognin

CRICIÚMA

2018

MAIKEU PAULINO DUARTE

UTILIZAÇÃO DO CONCEITO DE INTERNET DAS COISAS ASSOCIADA À TECNOLOGIA DE IBEACONS COMO FERRAMENTA COMPUTACIONAL APLICADA À INSTRUÇÃO DE USUÁRIOS NA UTILIZAÇÃO DE APARELHOS DE GINÁSTICA

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciências da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de Software.

Criciúma, 25 de junho de 2018.

BANCA EXAMINADORA

Prof. Me. Gustavo Bisognin - UNESC - Orientador

Prof. Esp. Fabricio Giordani - UNESC

Prof. Esp. Marcel Campos Inocêncio - UNESC

RESUMO

Com a constante procura por manter a saúde em dia, os ambientes de academia vêm ganhando espaço para a pratica de exercício físico, com a grande demanda e a quantidade limitada de profissionais capacitados para instruir os usuários na utilização dos aparelhos, alguns usuários optam por evitar as orientações e acabam executando movimentos incorretos nos aparelhos sendo possível ocorrer lesões. Devido a popularidade dos *gadgets* e alta procura por manter a saúde em dia, é possível criar uma aplicação que funciona como uma espécie de manual digital, onde auxilia o usuário na utilização dos aparelhos de academia. O desenvolvimento do protótipo foi realizado com a arquitetura REST para o lado server e com o framework *AngularJS* para o desenvolvimento do lado *cliente*, tendo como as duas principais linguagens de programação o *JAVA* e o *JavaScript*. Com o intuito de tornar a pratica de exercícios mais fácil, foram instruídos o conceito de Internet das Coisas na aplicação, onde ele é capaz de identificar um aparelho através dos *Beacons* e exibir um pequeno tutorial de como utilizar o equipamento.

Palavras-chave: Internet, Framework, Dispositivo móvel, Bluetooth, Beacon, Internet das Coisas, AngularJs, JAVA.

ABSTRACT

With the constant search for health, gym environments are gaining space for physical exercise, with the great demand and the limited number of professionals trained to instruct the users in the use of the devices, some users choose to avoid the orientations and end up executing incorrect movements in the devices being possible to occur injuries. Due to the popularity of the gadgets and high demand for keeping your health healthy, you can create an application that works as a kind of digital manual, where it assists the user in the use of the gym equipment. The development of the prototype was performed with the REST architecture for the server side and with the AngularJS framework for the development of the client side, having as the two main programming languages JAVA and JavaScript. In order to make exercise easier, the concept of Internet of Things in the application was instructed, where he is able to identify a device through the Beacons and to show a small tutorial on how to use the equipment.

Palavras-chave: Internet, Framework, Mobile Device, Bluetooth, Beacon, Internet of Things, AngularJs, JAVA.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
BE	*Back-End*
BLE	*Bluetooth Low Energy*
CLI	*Command line Interface*
CSS	*Cascading Style Sheets*
DAO	*Data Acess Object*
DOM	*Document Object Model*
FE	*Front-End*
HTML	*HyperText Markup Language*
IDE	*Integrated Development Environment*
IoT	*Internet of Things*
JDK	Java *Development Kit*
JSON	*JavaScript Object Notation*
MVC	*Model-View-Controller*
MVW	*Model View Whatever*
NFC	*Near Field Communication*
OOP	*Object-oriented programming*
REST	*Representational State Transfer*
RFID	*Radio-Frequency IDentification*
SDK	*Software Development kit*
SQL	*Structured Query Language*
TI	*Tecnologia da Informação*
UML	*Unifield Modeling Language*
URL	*Uniform Resource Locator*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A crescente evolução da tecnologia computacional aplicada ao conceito de internet das coisas vem apresentando diversas soluções que facilitam a vida dos usuários nos mais diversos níveis. Desta forma, algumas aplicações práticas destacam-se no mercado em geral, dentre elas, podemos citar as tecnologias interativas que criam uma conexão do usuário com as coisas cotidianas, como refrigeradores interativos, aparelhos de TV, pisos inteligentes e a domótica em geral.

Um dos objetos de estudo deste trabalho, aborda a Internet das Coisas, por ser uma tecnologia que vem ganhando espaço no mercado, com possibilidades infinitas de negócios. Como está tecnologia é uma tecnologia recente, ela possui um amplo mercado, empresas como Google, Microsoft entre outras estão trabalhando firme nesse conceito, com o intuito de revolucionar a tecnologia e o nosso dia-a-dia. Outra constatação importante, é a associação do conceito de *Internet of Things* (*IoT)* com os dispositivos beacons, através deles é possível recolher dados e informações que nos permitiriam criar aplicações que nos possibilitariam interagir com os objetos do nosso dia-a-dia.

A tecnologia mobile se caracteriza pela sua mobilidade, assim nos permitiria criar aplicações que nos trariam comodidade e facilidade em grande parte do nosso dia-a-dia, como no serviço, em casa e até mesmo em viagens. Com a ajuda das tecnologias citadas acima, aplicações mais ricas dê conteúdo e com finalidades ainda mais avançadas poderiam ser criadas, tornando tarefas rotineiras e cansativas mais simples. Atualmente a maioria da população tem acesso a gadgets, tornando viável a utilização destes conceitos.

Com a crescente procura por soluções referentes à qualidade de vida destaca as academias como sendo um dos principais recursos utilizados para a prática de exercícios associados à saúde e bem-estar. A maioria dos usuários iniciantes tem dificuldade em associar o nome ao exercício, e qual o equipamento usar para a prática do mesmo, nesses casos, torna-se necessário à ajuda de um personal trainer ou profissional capacitado, para que o mesmo lhe mostre qual o exercício e o equipamento que pode ser utilizado para a pratica.

O presente trabalho pretende abordar o estudo da internet das coisas com a junção de algumas tecnologias como *AngularJs*, *Ionic* dentre outras na implementação de um protótipo que visa à resolução do problema na falta de

conhecimento para determinadas práticas de exercícios nos ambientes de academia e também a falta de personal trainers disponíveis. Além disso, visa o estudo da tecnologia *Beacon* utilizando um aplicativo móvel que detecta a região que o usuário está contido e a localização do *Beacon*, e fornece informações ao usuário através do *web service*, como quais exercícios podem ser executados em determinados aparelhos e um breve vídeo de como executar o movimento.

1.1 OBJETIVO GERAL

Desenvolver um protótipo para auxiliar na execução de exercícios físicos no ambiente de academia.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste trabalho consistem em:

a) compreender o conceito de Internet das Coisas;
b) disponibilizar um protótipo para dispositivos móveis;
c) comunicação entre *Smartphones* e equipamentos de ginastica;
d) possibilitar a criação de treinos dinâmicos.

1.3 JUSTIFICATIVA

Nos dias atuais, as pessoas passam uma grande parte do tempo utilizando gadgets, como smartphones, tablets entre outros aparelhos interativos conectados à internet. Neste mundo da tecnologia da informação, tudo se torna mais prático e de rápido acesso, podendo resolver problemas, fazer compras, gerenciar encontros, participar de reuniões e acessar uma enorme quantidade de informação em poucos segundos. Desta forma, podemos destacar o surgimento de novas necessidades, algumas ferramentas já vêm auxiliando os usuários de forma disruptiva, criando um novo conceito de interação e de aplicação das tecnologias (ALMEIDA, 2015).

Com isso uma das tecnologias que vem crescendo é a chamada Internet das Coisas. Do inglês Internet of Things (IoT), a Internet das Coisas refere-se à integração de objetos físicos e virtuais em redes conectadas à Internet, permitindo

que “coisas” coletem, troquem e armazenem uma enorme quantidade de dados numa nuvem, em que uma vez processados e analisados esses dados, gerem informações e serviços em escala inimaginável (ALMEIDA, 2015).

Outra tecnologia que vem ganhando uma audiência importante no mercado, são os sensores emissores de bluetooth de baixo consumo de energia, denominados Beacons. Existem vários tipos de beacons presentes no mercado atual, porem um dos mais conhecidos é o iBeacon.

Basicamente, este dispositivo é um protocolo padronizado pela Apple, que utiliza os serviços de localização, fazendo com que emita um alerta com a localização do dispositivo (APPLE, 2016, tradução nossa), permitindo assim a interação deste, com diversas aplicações computacionais, podendo ser utilizado como forma de interação digital com as coisas.

Com a crescente utilização dos gadgets e destes tipos de tecnologia, a junção das tecnologias citadas, permite a criação de uma ferramenta computacional com o conceito de internet das coisas associada à tecnologia de ibeacons, para o acesso a conteúdos pré-determinados, como manuais de funcionamento, instruções de uso, consumo de produtos ou treinamentos dentre diversas outras aplicações.

Tendo em vista o crescimento da utilização das academias como forma de melhoria da qualidade de vida através da prática de exercícios, evidencia-se a necessidade de informações sobre como utilizar os mais diversos aparelhos no sentido de otimizar os treinos, bem como evitar lesões ocasionadas pelo mau uso de aparelhos de ginástica.

Além disso, a aplicação pretende proporcionar a criação de treinos dinamicamente para cada aluno, baseado no que ele procura, como: perda de peso, fortalecimento muscular entre outros. Com base nisso, a aplicação pretende criar as fichas de treino pré-escrito dinamicamente, proporcionando de uma forma mais ágil e prática o acompanhamento dos treinos, com informações de como o exercício deve ser executado, qual o equipamento pode ser utilizado entre outras informações.

Com base no que foi acima apresentado, a proposta de utilização do conceito de internet das coisas associada à tecnologia de ibeacons como ferramenta computacional aplicada à instrução de usuários na utilização de aparelhos de ginástica se justifica.

1.4 ESTRUTURA DO TRABALHO

Este trabalho divide-se em capítulos de forma a estruturar e facilitar a leitura e compreensão, sendo que o primeiro capítulo é destinado à introdução, justificativa, os objetivos gerais e específicos e estrutura do trabalho.

No segundo capítulo é abordado sobre comunicação disruptiva, explicando o conceito e abordando especificamente a instrução automática do uso de aparelhos e interação com aparelhos de ginastica que se encaixam no conceito de comunicação disruptiva.

No terceiro capítulo é apresentada a evolução das tecnologias da informação, onde é constatado a sua evolução e novas tecnologias que vão surgindo nesse meio, como o conceito de Internet das Coisas.

O quarto capítulo consta sobre a comunicação de sensores, dando ênfase na comunicação de sensores do tipo beacon, contendo informações de como configurar e sobre o seu funcionamento.

No quinto capítulo são apresentadas as tecnologias¸ abordando sobre HTML, CSS, *AngularJs*, Ionic e banco de dados *MySql*, dando ênfase nos conceitos base e um pouco sobre sua história.

O sexto capítulo apresenta a importância das aplicações móveis na vida dos usuários e a forma que agrega facilidade no cotidiano, dando ênfase em suas áreas de aplicação e também sobre o seu conceito.

Os trabalhos correlatos que possuem relação com este trabalho estão contidos no sétimo capítulo.

O desenvolvimento da aplicação web e do protótipo estão contidos no oitavo capítulo, explicando a metodologia aplicada no desenvolvimento.

O capitulo 9, o ultimo, contém a conclusão e trabalhos futuros.

2 EVOLUÇÃO DAS TECNOLOGIAS DA INFORMAÇÃO

A Tecnologia da Informação (TI) normalmente é definida como um conjunto de tecnologias, soluções digitais e sistemas que permitem a captura, o registro, o armazenamento e a análise de dados. Sua aplicabilidade é tamanha que pode ser inserida em diversas áreas, da agricultura às vendas de e-commerce (UNIPÊ, 2018).

No inicio a TI era usada apenas para fins militares, do governo e em algumas universidades. Alguns anos depois as empresas começaram a identificar que a TI poderia tornar a gestão de negócios mais fácil, com o armazenamento de dados, tornando esses dados em informações úteis.

Rezende (2002) afirma que as empresas viram a importância da informação para a gestão dos negócios, integrando a “informática” em seus sistemas, transformando-se em tecnologia da informação, integrando recursos modernos.

Com tudo isso surge a Internet, a conquista do milênio. A Internet nasceu da soma de pequenas conquistas tecnológicas feitas por cientistas brilhantes (BRASIL, 2000). Com a Internet veio mudanças gigantescas, novos meios de comunicação, novas ferramentas de trabalhos.

A tecnologia da informação alterou o mundo de uma forma drástica, desde que ela foi introduzida, os modelos de negócios de empresas já não são mais os mesmos, ao lançar um produto às empresas já pensam na parte tecnológica do mesmo, sempre visando inovação (McGEE; PRUSAK, 2004).

Fora dos negócios a tecnologia de informação mudou o nosso dia a dia, conseguimos enviar e trocar informações de uma maneira mais rápida e muito mais eficiente, oque antes era necessário enviar uma carta, hoje em apenas alguns minutos você pode resolver a sua pendencia pelo *Whatsapp* trocando algumas mensagens.

2.1 INTERNET DAS COISAS

A Internet das Coisas é uma inovação tecnológica, baseada na comunicação de aparelhos com objetos. O termo Internet das Coisas, ou *Internet of Things (IoT)* em inglês, foi apresentado primeiramente por Kevin Ashton da MIT Auto Centre, em uma apresentação sobre RFID e a cadeia de suprimentos de uma grande companhia, em 1999 (ASHTON, 2009).

Apesar do termo aparecer primeiramente em 1999, é possível encontrar em Venkatesh (1996) uma ideia, ou uma aplicação, bem aproximada do conceito de Internet das Coisas, ainda que não fosse chamado desta maneira.

Singer (2002) acredita que outra possível origem do termo Internet das Coisas pode ser encontrada na publicação do artigo *When Things Start to Think* de Neil Gershenfeld (1999), no qual o autor esboçou um cenário no qual objetos processam informação. Segundo a autora, o primeiro eletrodoméstico inteligente foi uma geladeira, lançada pela LG em 2002, que permitia conexão com a internet, poderia ser usada para refrigerar alimentos e, também, para navegar na internet, fazer compras, acessar agendas, ver TV ou ouvir rádio.

A partir de 2005, a discussão sobre a Internet das Coisas se generalizou, começou a ganhar a atenção dos governos e aparecer relacionada a questões de privacidade e segurança de dados. Foi neste ano que a Internet das Coisas se tornou a pauta do *International Telecommunication Union* (ITU), agência das Nações Unidas para as tecnologias da informação e da comunicação, que publica anualmente um relatório sobre tecnologias emergentes. Assim, depois da banda larga e da internet móvel, a Internet das Coisas ganhou a atenção do órgão e passou a figurar como o “próximo passo da tecnologia ‘always on’ [...] que prometem um mundo de dispositivos interconectados em rede” (ITU, 2005, p. 1).

Singer (2012) apresenta alguns exemplos de aplicação de internet das coisas, entre eles, a imagem de uma pessoa dirigindo um carro que vai mostrando a rota menos congestionada ao motorista, cuja casa está sendo limpa por um aspirador de pó inteligente, que trabalha sozinho, enquanto o fogão, também inteligente, está se preparando para cozinhar uma refeição. A mesma autora também cita um exemplo real, do Rio de Janeiro, no qual sensores, câmeras e camadas de informação mostram trânsito e ocorrências diversas em tempo real no Centro de Operações.

2.1.1 Domótica

O termo domótica é um neologismo da junção do radical latim *domus* que significa residência e robótica (CHAMUSCA, 2006), seguindo este contexto é a junção de sistemas informáticos, mecânicos, arquitetônicos, eletrônicos e de telecomunicações, aplicados a melhorias da segurança, comunicações, gestão energética e conforto.

O termo Domótica designa e referencia a utilização de processos automatizados em casas, apartamentos e escritórios. Pode-se utilizar outras denominações sinônimas, tais como, Automação Doméstica, Automatização Residencial ou Automação Residencial.

Atualmente, o mercado de automação tem crescido consideravelmente e a área de Domótica (Automação Residencial) tem ganhado muito destaque. Um ambiente pode possuir um amplo número de equipamentos elétricos e eletrônicos e a possibilidade de integração desses equipamentos é algo concreto. Podemos citar como exemplo, um sensor de presença que ao verificar sua entrada em algum cômodo, ele verifica a temperatura e o mesmo toma decisão de ligar o ar-condicionado deixando o ambiente com uma temperatura agradável.

Além do conforto, a Domótica pode proporcionar segurança, agregando a esta integração dispositivos de segurança como sensores de presença, *streaming* de câmeras de segurança, entre outros. Alguns serviços semelhantes já estão disponíveis no mercado, que ao identificar alguma movimentação em algum cômodo, é enviado uma notificação para o seu Smartphone onde você pode acompanhar as câmeras de segurança em tempo real.

2.1.2 Internet das Coisas na comunicação de Aparelhos

A “Internet das Coisas” se refere a uma revolução tecnológica que tem como objetivo conectar os itens usados do dia a dia à rede mundial de computadores. Cada vez mais surgem eletrodomésticos, meios de transporte e até mesmo tênis, roupas e maçanetas conectadas à Internet e a outros dispositivos, como computadores e smartphones (ZAMBARDA, 2014).

A maior característica dessa tecnologia é a possibilidade de conectar equipamentos eletrônicos com objetos do nosso cotidiano. Hoje empresas já possuem projetos em praticas com resultados satisfatórios.

Vários artefatos que se encaixam no conceito de *IoT* já existem comercialmente e a tendência é que cada vez mais aparelhos sejam desenvolvidos para isso, formando, assim, uma infraestrutura doméstica semelhante a de casas de filmes de ficção científica.

2.1.2.1 Amazon Echo

O *Amazon Echo* é uma torre que sempre está ouvindo e responde ao comando “Alexa”, como se fosse uma pessoa sendo chamada mesmo. No caso, uma pessoa com conexão à internet capaz de responder a uma variedade de comandos diferentes, que vão desde tocar uma música a adicionar itens numa lista de compras e interagir com os outros eletrodomésticos (OPSERVICES, 2016).

2.1.2.2 Sensores no datacenter

Os medidores de temperatura e umidade são um exemplo já consolidado de internet das coisas nos ambientes de TI. Tanto os sensores construídos a partir de Arduino como os comercializados pelo mercado já para um fim específico têm sido amplamente utilizados para controlar a temperatura e umidade de datacenters. Estes sensores são integrados à alguma ferramenta de monitoramento de redes e sistemas e, caso a sua temperatura ultrapasse 24ºC ou 70% de umidade, por exemplo, são gerados alertas para a tomada de ações corretivas. Esses alertas podem indicar, que os ar-condicionados não estão funcionando, colocando em risco a integridade dos servidores alocados no datacenter (OPSERVICES, 2016).

2.1.2.3 Tesla

Os carros da Tesla observam inicialmente seu design elegante aclamado pela imprensa, motores elétricos, mas a montadora inovadora também forneceu um modelo para trazer carros para a internet das coisas. Tesla fez um acordo no início de 2014 com AT&T para fornecer conectividade sem fio para seus carros, permitindo

diagnóstico remoto seguro, mapas atualizados e habilitados para internet proporcionando entretenimento para condutores e passageiros. Proprietários do novo modelo S desfrutam de rádio por internet e conectividade a seus smartphones via apps que controlam seus carros à distância, e facilitado encontrar quando se esquece do local onde foi estacionado.

3 COMUNICAÇÃO DISRUPTIVA

Disruptiva tem o sinônimo de “inovador, moderno, radical”, com a junção de comunicação torna-se algo admirável. A comunicação disruptiva é uma maneira de ganhar espaço no mercado, produtos ou serviços já existentes são modificados e novos produtos são criados com a finalidade de abrir novas oportunidades de negócio (DRAFT, 2015).

A inovação disruptiva, conforme Christensen (2006) descreve um processo pelo qual um produto ou serviço começa por aplicações simples, na parte inferior de um mercado e, progressivamente, se move para “acima do mercado”, acabando por deslocar ou eliminar concorrentes estabelecidos.

Seguindo o contexto de Christensen, a comunicação disruptiva segue o mesmo caminho, encontrar falhas em produtos já existentes e melhora o mesmo, iniciando com pequenos clientes, ganhando espaço e se tornando algo consolidado no mercado.

Através da comunicação disruptiva podemos criar aplicações interativas com maquinas e equipamentos, que trocam informações de uma maneira disruptiva, onde ambos enviam e recebem informações.

3.1 INSTRUÇÃO AUTOMÁTICA DO USO DE APARELHOS

Para Silvano (2017), uma das principais dificuldades de qualquer usuário de aparelhos eletrônicos ou analógico é o domínio sobre as instruções de uso. Estas instruções geralmente estão dispostas em manuais longos e complexos o qual possuem pouca audiência pelos usuários, fazendo com que, na sua maioria, executem sua utilização antes da leitura. Esta prática pode gerar problemas tanto para o aparelho ou até mesmo para quem está manuseando o mesmo.

Segundo Silvano (2017), o consumidor deve ficar atento e sempre ler com atenção as orientações sobre a utilização dos produtos e serviços que adquire ou contrata. O uso inadequado ou em desconformidade com as instruções contidas no manual poderá gerar a perda de garantia.

Com o surgimento das tecnologias associadas à comunicação das coisas, podemos entender que a aplicação de manuais interativos bem como instruções gamificadas, tem tomado um espaço importante na rede mundial de computadores.

Dentro desse contexto vemos surgir um novo fenômeno, chamado de gamificação (WERBACH; HUNTER, 2012, tradução nossa), que consiste na utilização de elementos dos games (mecânicas, estratégias, pensamentos) fora do contexto dos games, com a finalidade de motivar os indivíduos à ação, auxiliar na solução de problemas e promover aprendizagens (KAPP, 2012, tradução nossa).

Com a gamificação é possível à criação de manuais interativos, que antes de utilizar o aparelho, iria passar instruções importantes na hora do uso, como precauções, meio de uso e outras informações importantes, de uma maneira disruptiva e de um nível de entendimento mais fácil.

3.2 INTERAÇÃO COM APARELHOS DE GINÁSTICA

Preocupados com a saúde, diversos usuários procuram o ambiente das academias para manter a boa forma e a saúde em dia. Alguns fatores, como a falta de entendimento de como utilizar o aparelho de ginastica corretamente provoca algumas lesões ou não traz o resultado esperado.

Segundo Nogueira (2015), isso acontece tanto com os iniciantes como também com os mais experientes. Os erros variam desde a execução da atividade, excesso de carga, forma de segurar os equipamentos, até mesmo na velocidade do movimento e má postura corporal.

De acordo com Carolina (2017), a gamificação e a busca por experiências marcantes em diversos segmentos do mercado são também tendência nos ambientes *fitness*.

Com o crescente conceito de gamificação no mercado, é possível o uso dessa tecnologia para o auxilio de como executar os movimentos de forma correta, auxiliando na prevenção de lesões e ajudando a obter resultados satisfatórios como também a capacidade de interagir com o usuário no equipamento para tornar os treinos singulares, motivacionais e mais desafiadores.

4 COMUNICAÇÃO DE SENSORES

Com o avanço das tecnologias, hoje é possível conectar alguns dispositivos uns com os outros através de alguns sensores. Hoje a maioria dos aparelhos já contem alguns sensores, as TVs, por exemplo, já veem com sensores, onde conseguem obter acesso à internet e usufruir dos seus benefícios.

Os *smartphones* possuem um extenso número de sensores, como sensores de proximidades, sensores de temperaturas, Giroscópio entre outros. Através desses sensores, aplicativos conseguem coletar informações e serem reutilizados para algumas funções, o giroscópio, por exemplo, identifica como está a posição do seu celular e ajusta a tela girando, tornando a mesma mais confortável (DAVID NIELD, 2017).

Através da comunicação foi possível recolher um vasto numero de informações, temos uma série de sensores disponíveis hoje no mercado, *Beacons*, *Bluetooth, NFC* que são utilizados para recolher informações sem que seja necessários cabos ou algo do tipo.

4.1 BEACON

Beacons são dispositivos de proximidades em ambientes fechados que permitem localizar objetos ou pessoas que carreguem objetos (TEIXEIRA, 2014). Pode-se concluir que os beacons estão para ambientes fechados assim como os GPS estão para ambientes externos.

O *Estimote Beacon*, modelo utilizado para a realização desse trabalho, é considerado um *iBeacon* por se comunicar através do padrão tecnológico desenvolvido pela Apple (ROCHA et al., 2014). Na figura 1 pode-se visualizar um *beacon* do modelo *Estimote* em visão explodida.

> O Estimote Beacon trata-se de um pequeno super-computador, possuindo um processador 32-bit ARM® Cortex M0 CPU com 256kB memória flash, acelerômetro, sensor de temperatura e 2.4 GHz Bluetooth 4.0 Smart, também conhecido como BLE ou Bluetooth Low Energy, envoltos por uma estrutura flexível feita de silicone. (ROCHA et al., 2014).

Figura 1 – Beacon estimote em visão explodida.

Fonte: Rocha (2014).

Ainda segundo Rocha et al (2014, p. 28), “O dispositivo Estimote utiliza a comunicação BLE através de um formato padronizado pela Apple no qual cada pacote transmitido consiste em cinco partes principais de informação":

a) prefixo *iBeacon* (9 bytes): string padronizada pela *Apple*;
b) UUID (16 bytes): utilizado para diferenciar um alto grupo relacionado de *beacons*;
c) major (2 bytes): utilizado para diferenciar pequenos grupos de *beacons*;
d) minor (2 bytes): identificador individual de cada *beacon;*
e) tx power (2 bytes): definido com a força do sinal a um metro de distância do aparelho.

Na figura 2 está representado esquematicamente o pacote de dados transmitido.

Figura 2 – Representação pacote de dados transmitido pelo Estimote beacon.

Prefixo iBeacon (9 bytes)	UUID (16 bytes)	Major (2 bytes)	Minor (2 bytes)	TX Power (2 bytes)

Fonte: Rocha (2014).

O pacote segue inicialmente com o prefixo iBeacon, UUID, *major*, *minor*, *tx power*. Cada *Estimote beacon* vem com um UUID fixo e *major* e *minor* randomizados. É possível também alterar o UUID, *major* e *minor* do *Estimote beacon*, porém se faz necessário realizar uma autenticação pela fabricante em sua plataforma web chamada *Estimote Cloud* (STECZKIEWICZ, 2016, tradução nossa).

4.1.1 Sinal do Beacon

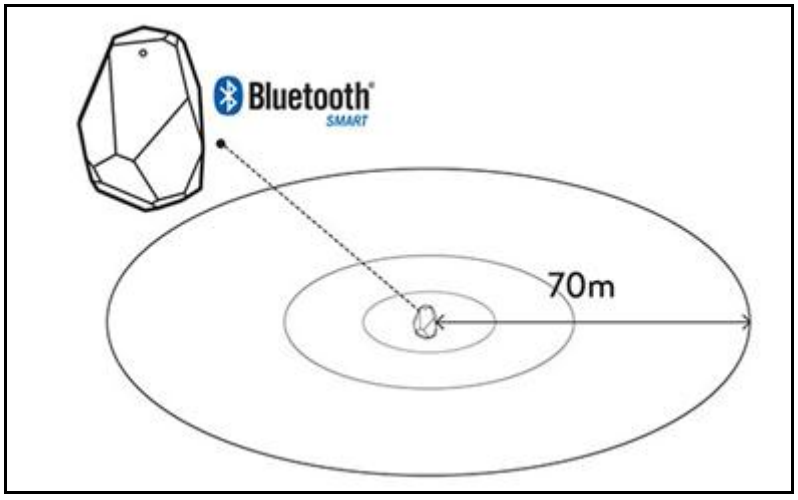
Os *beacons* trocam informações com outros aparelhos através do sinal de *Bluetooth Low Energy* (BLE). Celulares e outros aparelhos recebem os sinais enviados periodicamente pelos beacons, sem necessidade de pareamento prévio, e estimam sua distância através da força do sinal recebido, sendo que quanto mais perto o usuário estiver, mais forte será o sinal. Dependendo da implementação, aparelhos podem examinar o sinal a cada 1 segundo ou 10 vezes por segundo, quanto mais frequente for, mais estável será o sinal e melhor a experiência do usuário (STECZKIEWICZ, 2016, tradução nossa).

É importante também ressaltar que é possível receber sinal de mais de um *beacon* ao mesmo tempo, desde que a aplicação seja bem implementada e o *beacon* seja configurado corretamente.

A distância máxima que o *Estimote Beacon* consegue enviar sinais depende do ambiente em que ele está localizado, dado que o sinal pode ser difratado, sofrer interferência ou absorvido por água, por exemplo. Em geral, ele é capaz de enviar e receber sinais a distâncias que variam de 5 centímetros a até aproximadamente 70 metros, sendo essa distância categorizada em três regiões distintas, como visto na figura 3.

a) *immediate*: poucos centímetros;
b) *near*: menor do que 10 metros;
c) *far*: maior do que 10 metros.

Figura 3 – Desenho esquemático das três regiões (*near*, *far* e *immediate*) de atuação do Estimote Beacon.



Fonte: estimote.com.

O dispositivo pode, ainda, ser programado para enviar diferentes tipos de mensagens de acordo com qual das três regiões o usuário se encontrar.

4.1.2 Aplicação do Beacon como apoio na Internet das Coisas

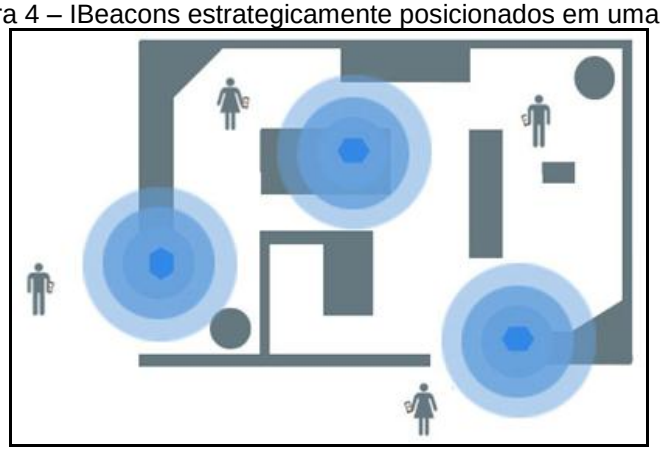
Com a crescente da internet das coisas, novos dispositivos são criados para interagir com este conceito. O *Beacon* é um aparelho que nos permite receber sinais e com estes sinais é possível manipula-los e criar aplicações mais ricas que conseguem interagir com objetos e clientes.

Segundo Teixeira (2014) a Apple, como não podia deixar de ser, já está utilizando a tecnologia em 254 lojas nos EUA. As funcionalidades já estão embutidas na versão oficial do aplicativo da Apple Store para iOS. O que ele faz? Quando o usuário se aproxima de uma loja física, o aplicativo oferece toda uma camada extra de informações e serviços que são específicos para aquela unidade — como por exemplo ofertas locais, tamanho da fila para ser atendido no Genius Bar, eventos e treinamentos que estão agendados na loja etc.

4.1.2.1 Interagir com os clientes dentro da loja

Muito popular nas lojas de varejo nos Estados Unidos, os iBeacons são utilizados para aumentar a interação entre os clientes e os produtos expostos na loja. O objetivo é identificar em qual departamento da loja o cliente está e com essa informação é possível enviar notificações customizadas para o celular do cliente (CARNEIRO, 2016) (figura 4).

Figura 4 – IBeacons estrategicamente posicionados em uma loja.



Fonte: usemobile.com.br.

Recentemente o Walmart atualizou seu aplicativo, adicionando os recursos de iBeacon. Com esse novo recurso e iBeacons estrategicamente espalhados pelas lojas, é possível identificar qual setor da loja o visitante ficou mais tempo e mostrar promoções baseados nesses interesses (CARNEIRO, 2016).

4.1.2.2 Industria Hospitalar

Segundo Carneiro (2016) o *beacon* tem apresentado diversas soluções para o ambiente hospitalar, mas o que mais tem se destacado é a substituição da tradicional prancheta com o prontuário do paciente. O Hospital Universitário de Lausanne, na Suíça foi o pioneiro. As informações de cada paciente eram atribuídas a um *beacon*, onde os funcionários utilizavam um aplicativo que se aproximar do *beacon* o aplicativo recebia as informações do prontuário atualizado do paciente (figura 5).

Figura 5 – Beacons facilitando o dia-a-dia em hospitais.

Fonte: usemobile.com.br.

Durante o dia, os médicos e enfermeiros ao fazerem a visita de rotina, utilizando-se de um aplicativo recebem de forma rápida e eficiente o prontuário de cada paciente. Dessa forma, conseguem saber quais remédio devem ser aplicados e quais procedimentos devem ser feitos (CARNEIRO, 2016).

5 TECNOLOGIAS

Com o crescimento das necessidades, novas tecnologias são criadas para suprirem as mesmas, atualmente existe um número imensurável de tecnologias no mercado.

Com a popularização da *internet,* hoje existe a opção de trocar informações de maneira remota, simples e segura. Essa tecnologia é usada tanto para fins de lazeres como comerciais. A internet vem se destacando como um meio de comunicação que possui um alto nível de eficácia, devido à fácil interação e agilidade para realizar o tráfego e trocas de informações. Hoje existem inúmeras aplicações que possibilitam a interação de usuários através de vídeo conferencias ou ligações. Junto dela veio à popularização das redes sócias, que são páginas *web* onde usuários criam contas e fazem posts e compartilham suas ideias.

Seguindo esse contexto, uma das tecnologias que vêm crescendo de uma forma satisfatória é a *WEB*, com a demanda de criações de sites com o propósito institucional, informativos, e-Commerce, armazenamento de informações, portais, mídias sociais ou então instrumentos de publicidades, vem crescendo de forma exponencial no comércio mundial. Além desses fatores a popularização dos *smartphones* e afins contribui para grandes empresas apostarem nesse tipo de negócio para aumentarem suas vendas.

As aplicações *web* são interpretadas por navegadores, que ficam responsáveis de converter o código para uma maneira visual. Com o surgimento de navegadores diferentes, foi necessária a criação de uma padronização para que os navegadores interpretem as páginas *web* da mesma maneira, sendo assim todos os navegadores interpretam as três linguagens padrões: o *HyperText Markup Language* (HTML), *Cascading Style Sheets* (CSS) e *JavaScript* que com a junção dessas três tecnologias é possível a criação de páginas ricas.

5.1 JAVA

Segundo Indrusiak (1996), *Java* é uma linguagem computacional completa, adequada para o desenvolvimento de aplicações baseadas na rede Internet, redes fechadas ou ainda programas *stand-alone*.

A linguagem *Java* surgiu para resolver alguns problemas conhecidos pelas outras linguagens utilizadas na década de 1990. Ela foi criada com o propósito de solucionar os problemas: ponteiros; gerenciamento de memória; organização; falta de bibliotecas; problemas com sistemas operacionais diferentes.

A linguagem Java até o presente momento conseguiu resolver estes problemas que ocorriam com certa frequência nas linguagens da época.

Segundo Perry (2016) a linguagem Java deriva da linguagem C, portanto suas regras de sintaxe assemelham-se às regras de C. Por exemplo, os blocos de códigos são modularizados em métodos e delimitados por chaves *({ e })* e variáveis são declaradas antes que sejam usadas.

Estruturalmente, a linguagem Java começa com pacotes. Um pacote é o mecanismo de *namespace* da linguagem Java. Dentro dos pacotes encontram-se as classes e dentro das classes os métodos, variáveis, constantes e mais (PERRY, 2016).

5.1.1 História

O Java foi criado na década de 90, foi desenvolvido por uma equipe de programadores chefiada por James Goling considerado o pai do Java.

A *Sun* criou um time (conhecido como *Green Team*) para desenvolver inovações tecnológicas em 1992. O time voltou com a ideia de criar um interpretador para pequenos dispositivos, facilitando a reescrita de *software* para aparelhos eletrônicos, como vídeo cassete, televisão e aparelhos de TV a cabo. A ideia não deu certo. Tentaram fechar diversos contratos com diversos fabricantes de eletrônicos, como *Panasonic*, mas não houve êxito devido ao conflito de interesses e custos. Hoje, sabemos que o Java domina o mercado de aplicações para celulares com mais de 2.5 bilhões de dispositivos compatíveis, porém em 1994 ainda era muito cedo para isso (CAELUM, p.13).

Com a internet em crescimento, a *Sun* identificou que poderia usar o *Java* para rodar aplicações no *browser,* havia uma semelhança com os aparelhos, devido a ambos terem vários sistemas operacionais, e seria de uma enorme vantagem programar em uma linguagem que seria aceita por todas as plataformas, foi nesse momento que foi lançado a versão 1.0 do *Java*.

A partir daí a velocidade dos acontecimentos seguintes foi assustadora. O número de usuários cresceu rapidamente, e grandes fornecedores de tecnologia como a IBM anunciaram suporte para a tecnologia Java.

Em 2009 a Oracle comprou a Sun, fortalecendo a marca. A Oracle sempre foi, junto com a IBM, uma das empresas que mais investiram e zeram negócios através do uso da plataforma Java. Em 2014 surge a versão Java 8 com mudanças interessantes na linguagem (CAELUM, p.14).

5.2 IONIC

O Ionic é um *framework* para o desenvolvimento de aplicações mobile híbridas através de um SDK - *Software Development Kit HTML5* que permite que as aplicações sejam construídas através da utilização de tecnologias utilizadas no desenvolvimento para a *WEB*, *HTML*, *CSS* e *Javascript* (IONIC, 2017, tradução nossa).

O Ionic é construído em cima do *AngularJS* e utiliza o *Apache Cordova* para construção da aplicação de um *Webview* que de maneira simplificada seria a execução em um navegador que não possui barra de endereço e que é capaz de renderizar as tecnologias acima citadas. Devido a esta integração com o *AngularJS* o *Ionic* herdou várias diretivas, que é uma das *features* do ecossistema do AngularJS que permite estender o HTML de modo a fornecer reuso, componentização e lógica para a marcação.

O Ionic é um *framework Cross-Platform*, ou seja, você cria aplicações simples e ele converte os para aplicativos móveis nativos para todas as principais lojas de aplicativos.

O Ionic tem seu foco o *Look and Feel*, ou seja, na interação do usuário com a interface da aplicação, através da construção de interfaces poderosas e de simples desenvolvimento. Embora exista uma dependência do AngularJS para desenvolvimento para consumir todo o recurso que o Ionic pode oferecer, pode-se utilizar apenas o CSS mas perdendo assim componentes de interação, animação dentre outros já fornecidos na plataforma (IONIC, 2017, tradução nossa).

O SDK do Ionic é de código aberto hospedado no Github, oque permite que toda a comunidade contribua com melhorias assim beneficiando todos e tornando uma ferramenta com ampla influencia no mercado. Está sob a licença da

MIT o que permite o desenvolvimento de aplicações pessoais ou comerciais sem custo.

5.2.1 História

A Ionic foi fundada em 2012, quando o uso de tecnologias da web como um meio para construir aplicativos nativos ainda estava em sua infância. Quando começamos, queríamos apenas criar uma maneira melhor para os desenvolvedores web usar suas habilidades existentes para criar aplicativos para as lojas de aplicativos (IONIC, 2017, tradução nossa). A empresa foi fundada pelos dois amigos Ben Sperry e Max Lynch.

Hoje, a Ionic é a plataforma de tecnologia de desenvolvimento móvel de plataforma cruzada mais popular do mundo, que impulsiona startups de rápido crescimento para algumas das maiores empresas do mundo (IONIC, 2017, tradução nossa).

5.2.2 Ferramentas

O Ionic possui um conjunto de ferramentas que auxiliam os desenvolvedores, tendo isso com um diferencial, além de sua excelente documentação. Podemos citar alguns itens desse conjunto de ferramentas:

a) **uma CLI simples**: com o auxilio da CLI você pode criar aplicações de uma maneira fácil e ágil, ela disponibiliza um conjunto de comandos que auxiliam os desenvolvedores em inúmeras ocasiões, como iniciar projetos, emular a aplicação em dispositivos entre outros;

b) **ionic Native**: com ele você pode usar recursos nativos do seu dispositivo, como câmera, contatos. Ele também tem o auxilio de identificar e solucionar problemas quando você tanta abrir algum item nativo e o mesmo não funciona bem.

c) live reload: uma ferramenta muito importante na hora do desenvolvimento, onde fica observando qualquer mudança no código fonte da aplicação e automaticamente recarrega a aplicação quando necessário, permitindo assim que seja menos necessário a reconstrução da aplicação a cada pequena alteração.

d) **ionicons**: são centenas de ícones mais comuns utilizados nas aplicações com a licença MIT.

Estes são alguns dos itens que tornam o Ionic um *framework* muito forte no mercado.

5.3 BANCO DE DADOS MYSQL

O *MySQL* é um servidor gerenciador de banco de dados (SGDB) relacional, de licença dupla (sendo uma delas de software livre), projetado inicialmente para trabalhar com aplicações de peque e médio porte, mas hoje atendendo a aplicações de grande porte e com mais vantagens do que seus concorrentes. Possui todas as características que um banco de dados de grande porte precisa, sendo reconhecido por algumas entidades como o bando de dados *open source* com maior capacidade para concorrer com programas similares de código fechado, tais como SQL Server (da *Microsoft*) e Oracle (MILANI, 2007).

O MySQL á uma alternativa atrativa porque, mesmo possuindo uma tecnologia complexa de banco de dados, seu custo é bastante baixo. Tem como destaque suas características de velocidade, escalabilidade e confiabilidade, o que vem fazendo com que ele seja adotado por departamentos de Tecnologia da Informação (TI), desenvolvedores web e vendedores de pacotes de softwares.

Para utilizar o MySQL, é necessário instalar um servidor e uma aplicação cliente. O servidor é o responsável por armazenar os dados, responder às requisições, controlar a consistência dos dados, bem como a execução de transações concomitantes entre outras. O cliente se comunica com o servidor através da SQL. A versão gratuita do MySQL é chamada de Edição da Comunidade e possui o servidor e uma interface gráfica cliente (MySQL Workbench).

5.3.1 História

O MySQL teve origem quando os desenvolvedores David Axmark, Allan Larsson e Michael Widenius, na década de 90, precisavam de uma interface SQL compatível com as rotinas ISAM que utilizavam em suas aplicações e tabelas. Em um primeiro momento tentaram utilizar a API mSQL, contudo a API não era tão rápida quanto eles precisavam, pois utilizavam rotinas de baixo nível (mais rápida

que rotinas normais). Utilizando a API do mSQL, escreveram em C e C++ uma nova API que deu origem ao MySQL (MILANI, 2007).

Com o ultimo resultado gerado por essa nova API, o MySql começou a ser difudindo e seus criadores fundaram a empresa responsável por sua manutenção, que é a MySQLLAB.

De acordo com Milani (2007) a partir dessa fase, o MySQL tornou-se mais conhecido por suas características de rápido acesso e cada vez mais utilizado. Novas versões foram lançadas, contemplando novas necessidades e firmando, assim, sua posição no mercado. Suas versões mais recentes no qual já constam novas funcionalidades, estabelecem sua capacidade de competir com bancos privados de maior popularidade (SQL Server e Oracle).

5.3.2 MySql Workbench

O MySQL Workbench é uma ferramenta gráfica para trabalhar com servidores MySQL e bancos de dados. O MySQL Workbench suporta completamente o servidor MySQL versões 5.5 e superiores. Também é compatível com versões anteriores do servidor MySQL 5.x, exceto em determinadas situações (como exibir a lista de processos) devido a tabelas de sistema alteradas. Não suporta versões do servidor MySQL 4.x (MYSQL, 2017, tradução nossa).

As suas principais funcionalidades estão divididas em cinco tópicos principais:

a) **desenvolvimento SQL**: permite que você crie e gerencie conexões para servidores de banco de dados. Além de permitir que você configure parâmetros de conexão, o MySQL Workbench fornece a capacidade de executar consultas SQL nas conexões do banco de dados usando o SQL Editor incorporado;

b) modelagem de dados: permite que você crie seu modelo de dados graficamente. Possui funcionalidades fáceis de usar para edição de tabelas, colunas, índices, opções, inserções e privilégios, rotinas e visualizações;

c) administração do servidor: permite que você configure e administre a instancia do MySQL, configure e crie usuários, programar backup e visualizar o seu desempenho;

d) **migração de dados**: permite que você migre do Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere, PostreSQL e outras tabelas RDBMS, objetos e dados para o MySQL. A migração também suporta a migração de versões anteriores do MySQL para os lançamentos mais recentes;
e) **suporte empresarial do MySQL**: suporte para produtos corporativos, como MySQL Enterprise Backup, MySQL Firewall e MySQL Audit.

O MySQL Workbench está disponível em duas edições: a Community Edition e a Commercial Edition. A Community Edition está disponível gratuitamente. A Commercial Edition fornece recursos empresariais adicionais, como acesso ao MySQL Enterprise Backup, MySQL Firewall e MySQL Audit (MYSQL, 2017, tradução nossa).

5.4 ANGULARJS

O AngularJS é um *framework open-source* desenvolvido pela Google, que fornece aos desenvolvedores *JavaScript* uma estrutura robusta para elevar a produtividade e desenvolvimento de aplicações ricas e com potencial (DIETZ, tradução nossa). Seu objetivo é aumentar aplicativos que podem ser acessados por um navegador web, sob o padrão *Model–View–Controller* (MVC), em um esforço para facilitar tanto o desenvolvimento quanto o teste dos aplicativos.

Foi criado por Misko Hevery e Adams Abrons em 2009. Inicialmente, o projeto tinha como objetivo facilitar a criação de aplicações web. Alguns anos depois Misko foi trabalhar no Google (FELIZARDO, 2015). Misko trabalhava em um projeto que continha algo em torno de 17.000 linhas de código, Ele apostou com seu gerente que reescreveria o código do projeto em 2 semanas, contanto que pudesse utilizar o framework que tinha desenvolvido. Eles terminaram de reescrever o código em 3 semanas, mas reduziu o projeto para 1.500 linhas. Com isso a *Google* foi adicionando esse framework em seus projetos internos lentamente, e apoiar este projeto, sendo que a primeira versão foi lançada em junho de 2012.

Uma das principais particularidades que torna a utilização deste *framework* tão assíduo no desenvolvimento de páginas *web* é devido à funcionalidade de trabalhar como uma extensão do documento HTML, com o *DataBinding*, *templates* e a fácil utilização do *Ajax*.

Segundo Schmitz e Lira (2012), este *framework* é praticamente uma linguagem declarativa, pois utiliza a adição de propriedades para modificar o comportamento da linguagem e interage de forma dinâmica com os elementos HTML, sendo que estes parâmetros são denominados de diretivas.

Adicionado recentemente à lista de *frameworks* que utiliza o padrão MVC, acabou atraindo muita atenção devido ao sistema de modelos inovadores, facilidade de desenvolvimento e práticas de engenharia sólidas, sendo que a modelagem de sistema se torna diferencial devido à utilização do HTML como linguagem de *templates*, abstração da manipulação do *Document Object Model* (DOM) explicitamente e possui componentes extensíveis.

Inicialmente a aplicação do AngularJS utilizava o modelo arquitetural de MVC no desenvolvimento de software, e posteriormente os próprios autores informaram a adesão do modelo *Model-View-Whatever (*MVW). Independente do padrão de projeto de software que será utilizado para o desenvolvimento da página web, é necessário sempre seguir os conceitos de *model*, *view* e *controller*, para separar corretamente as funcionalidades para cada arquivo, dos recursos mais importantes. O que difere o *framework* AngularJS com os demais *frameworks* JavaScript, é a utilização de diretivas, que agregam um valor elevado ao HTML pois lida diretamente com o DOM.

5.4.1 DataBind

Segundo Schmitz e Lira (2014), uma das principais vantagens do *AngularJS* é o seu *DataBind*. Este termo é compreendido como uma forma de ligar automaticamente uma variável qualquer a uma outra. Geralmente, o *DataBind* é usado para ligar uma variável do JavaScript (ou um objeto) a algum elemento do documento HTML.

O *DataBind* auxilia na captura de dados de uma forma fácil e ágil, tornando o fluxo de informações algo simples e rápido, sendo assim tendo um considerável na performance do desenvolvimento.

Conforme a figura 6, é um exemplo de utilização de *DataBind*, onde utilizamos a diretiva ng-model para vincular o campo a variável “seuNome” na linha 7, a editar o conteúdo do campo, ocorre o processo de *DataBind* que atualiza todas as outras instancias de “seuNome” disponível na aplicação.

Figura 6 – Exemplo de Databind.

```
<html ng-app>
  <head>
    <title>Exemplo DATABIND</title>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.4/angular.min.js"></script>
  </head>
  <body>
    Seja bem-vindo, <input type="text" ng-model="seuNome"/>
    <br/>
    <h1>Seja bem-vindo, {{seuNome}}</h1>
  </body>
</html>
```

Fonte: Do autor.

No *DataBind*, alterações na *view* são refletidas na fonte de dados e atualizações na fonte refletem na *view* sem a necessidade de manipulação explícita do DOM (ALMEIDA, 2016) (figura 6).

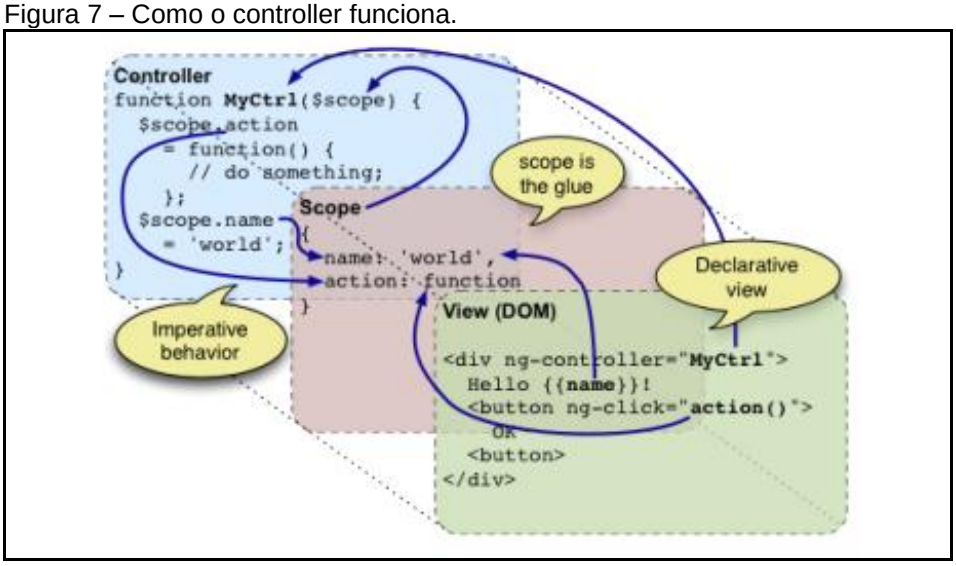
5.4.2 Controllers

Os controllers em AngularJS centralizam as regras de negócios das telas desenvolvidas, sendo que não deve fazer referência ou manipulações no DOM de forma direta (DIETZ, 2013, tradução nossa).

Um *controller* é, na maioria das vezes, um arquivo *JavaScript* que contém funcionalidades pertinentes à alguma parte do documento HTML. Não existe uma regra para o *controller*, como por exemplo ter um *controller* por arquivo HTML, mas sim uma forma de sintetizar as regras de negócio (funções *javascript*) em um lugar separado ao documento HTML (SCHIMTZ; LIRA, 2013).

Segundo Fisher (2014), um aplicativo AngularJS, o controlador e a exibição compartilham um objeto chamado de escopo, este objeto é o núcleo de sua admirável ligação de dados bidirecionais. O controlador define propriedades no escopo e a exibição se liga a essas propriedades. O AngularJS assume a responsabilidade de manter os dois em sincronia (figura 7).

Figura 7 – Como o controller funciona.



Fonte: Schmitz e Lira (2014).

5.4.3 Injeção de dependências

Injeção de dependência é uma forma de desacoplar e tornar o código mais flexível, testável e organizado (CANTARINO, 2014).

Segundo Cantarino (2014), O AngularJS possui uma funcionalidade de injeção de dependência que é muito inteligente. Ele consegue reconhecer de forma automática os parâmetros dos controllers e injetar as respectivas dependências.

Segundo Green e Seshadri (2013, tradução nossa), a injeção de dependências do AngularJS é o que permite aos desenvolvedores a seguir um estilo de desenvolvimento, podendo reutilizar as funções criadas em diferentes escopos. Este recurso possibilita a injeção do código fonte em outra parte do código da aplicação, requerendo então uma divisão de funcionalidades, como serviços, controladores ou provedores.

5.5 JAVA SCRIPT

O JavaScript é uma das linguagens de programação mais populares por ser a linguagem de HTML, para a *web*, computadores, servidores, notebooks, tablets, smartphones e muito mais (CROCKFORD, 2008, tradução nossa). Esta é uma linguagem de programação interpretada, utilizada para ser executada do lado do cliente, e para fazer as interações com o usuário sem a obrigação de fazer a ida e

volta dos dados ao servidor, controles no navegador, comunicações assíncronas e alterações das informações no documento.

Para introduzir o código JavaScript, basta incluir a *tag* <script> para que a página aguarde o *parsing* do *script* e a sua execução, podendo estar introduzido dentro da *tag*, ou então em um documento externo. Conforme Zakas (2013) quando o navegador se depara com a *tag* <script>, o mesmo interrompe o processamento da página e executa o código JavaScript, para posteriormente prosseguir com o carregamento da página, caso o código esteja em um documento externo declarado no atributo *src*, primeiramente o navegador irá fazer o download do documento para em seguida prosseguir com a execução do código.

A linguagem JavaScript suporta os elementos da sintaxe de programação da linguagem C, tendo como exceção o escopo, pois o JavaScipt faz uso do escopo somente a nível de função, segundo Zakas (2013), o escopo implica na determinação de quais variáveis a função poderá ter acesso através do *this*. Outra característica é a tipagem dinâmica, que são os tipos associados aos valores e não a variáveis, sendo assim uma linguagem de tipagem fraca, sendo que também possui a características de funções de primeira classe, que são objetos que contém propriedades e métodos, que permite a passagem de argumentos e retornar qualquer objeto.

Este tipo de linguagem tem um uso interessante quando se tenta tornar páginas web mais dinâmicas, porém em certos casos há necessidade de escrever grandes códigos para fazer funções relativamente simples, principalmente quando se tenta utilizar requisições *Asynchronous Javascript and* XML (AJAX), que são requisições assíncronas, estas são submetidas para o servidor por diversas ações diferentes da página, e não necessariamente precisam levar todos os campos do formulário (RUEBBELKE, 2012, tradução nossa).

Pensando na dificuldade em programar recursos complexos com JS, começaram a surgir diferentes *frameworks* que controlam funções mais onerosas para os programadores, como por exemplo, o AngularJS, este *framework* foi desenvolvido com objetivo principal de facilitar formulários com requisições AJAX (GREEN; SESHADRI, 2013, tradução nossa).

5.6 CSS

Cascading Style Sheet - Folha de Estilo em Cascata (CSS) é usado para estilizar e criar *layout web*. Por exemplo, para alterar fonte, cor, tamanho e espaçamento do conteúdo, dividir o conteúdo em múltiplas colunas, ou adicionar animações e outros componentes decorativos (LIMA; SILVA, 2017).

Segundo Olsson e O'Brien (2008, tradução nossa) a primeira versão do CSS, possuía as propriedades para controle de tipografia, como fontes, alinhamento de textos, margens, espaçamentos e formatação da lista, permitindo também especificações das dimensões das caixas de blocos e especificações de caixas com bordas, porém deixava a desejar com as personalizações de *layout* e *design*.

A versão atual (CSS 3), já contem um número de funcionalidades expressivas que auxiliam na manipulação das páginas como:

a) selecionar primeiro e último elemento;
b) selecionar elementos pares ou ímpares;
c) selecionar elementos específicos de um determinado grupo de elementos;
d) gradiente em textos e elementos;
e) bordas arredondadas;
f) sombras em texto e elementos;
g) manipulação de opacidade;
h) controle de rotação;
i) controle de perspectiva;
j) animação.

A sintaxe do CSS é definida por seletores que tem como principal característica selecionar um elemento, para posteriormente modificar de acordo com as propriedades e valores determinados na regra de formatação. Segundo Olsson e O'Brien (2008, tradução nossa), os seletores é um conjunto de regras, que estão contidos em blocos de chaves *{ }*, sendo que todos os elementos contidos nesta seleção, deverão ter a sua regra aplicada.

Os seletores têm ligações diretas com os elementos das páginas HTML, é através dele que se aplica o estilo desejado para o elemento, através de regras de estilo, com essas regras é possível mudar a cor do texto como também posicionar da maneira desejada (figura 8).

Figura 8 – Sintaxe seletor.

Fonte: Do autor.

Os seletores são identificados nos elementos HTML através de CLASS e por ID, sendo atribuído um valor para essa propriedade onde ela funcionara como um identificador para ser utilizado para aplicar estilos por meio das regras.

5.7 HTML

O HTML, do inglês *HyperText Markup Language*, é uma linguagem de marcações padrão utilizada para criação de páginas web. Os navegadores interpretam estas marcações e formatam o texto de acordo com elas. Além disso, a linguagem pode ser utilizada para adicionar outros conteúdos às páginas, como imagens, áudio e até vídeos (MILETTO; BERTAGNOLLI, 2014).

Conforme Powel (2010, tradução nossa), um arquivo HTML é um documento que possui diversas informações as quais se deseja publicar na *web*, e as instruções de marcações para que o navegador estruture e apresente a página conforme o desejado.

Esta linguagem é baseada no conceito de hipertexto, que seria a junção de vídeos, áudios, imagens e textos, que unidos compõe uma rede de informação, no qual nos permite trafegar por conteúdos através de *links*.

A ultima versão do HTML, o chamado HTML5, possui com um dos principais objetivos a manipulação de elementos, em versões anteriores era necessário a utilização de *divs* e identificadores para a organização das páginas. O HTML5 trouxe um novo conceito para isso, criando algumas *tags* padrões para esse comportamento. Segundo Batista (2009), com o HTML5 é possível utilizar as *tags* de estruturas do corpo da página, eliminando então a necessidade de criação de *divs* encadeadas.

As *tags* possuem uma marca de início, e não necessariamente uma marca de fim, mas quando possuem a marca de fim é definida por uma barra dentro da *tag*, sendo que esta *tag* pode envolver algum conteúdo que deverá ser alterado de acordo com o comando especificado (POWEL, 2010, tradução nossa).

Utilizando as *tags* em conjunto conseguimos obter os resultados vistos em paginas *webs* constantemente acessados por diferentes usuários, para obter informações ou até mesmo como forma de lazer.

6 APLICAÇÕES MÓVEIS

Um aplicativo móvel ou aplicação móvel é um sistema desenvolvido para ser instalado em um dispositivo eletrônico móvel, como tablets e smartphones. Os aplicativos são normalmente conhecidos como “*apps*” ou “*app mobile*”. Em 2010, o termo se tornou tão popular que foi assinalado como “palavra do ano” pela Sociedade Americana de Dialeto (FERNANDES, 2016).

Os aplicativos são instalados nos dispositivos móveis por meio de uma loja online, como Google Play, App Store ou Windows Phone Store. Alguns aplicativos disponíveis podem ser baixados de forma gratuita, outros são pagos. Um mesmo aplicativo pode custar um valor diferente dependendo do dispositivo para o qual é baixado (FERNANDES, 2016).

Assistentes pessoais ou agendas eletrônicas, os dispositivos móveis passaram a serem computadores que podem ser facilmente levados a qualquer lugar, criados para atender profissionais e pessoas em movimento que necessitam de rapidez, facilidade e segurança no acesso a informações corporativas e pessoais.

Cada vez mais pessoas utilizam os seus dispositivos móveis para navegar na web, acessar sites, utilizar aplicativos. Muitos *apps* se tornaram indispensáveis para o dia a dia dos usuários, simplesmente por serem úteis (TOTALCROSS, 2017).

Seguindo esse contexto, empresas renomadas apostam nesse tipo de tecnologia como o diferencial para o seu negócio. De acordo com Simova (2017), com a forte tendência de serviços digitais e a massiva utilização de dispositivos móveis, a tecnologia mobile para empresas é cada vez mais empregada e, mais do que isso, tem sido uma forma de alinhar as empresas ao mercado atual.

Segundo Campos (2017), o uso das Aplicações móveis para empresas na vida diária das pessoas é algo que vem crescendo dia após dia ao redor do mundo, tendo em vista que os aplicativos móveis facilitam a gestão de empresas, nossas vidas e de nossos trabalhos.

As aplicações móveis são utilizadas frequentemente como uma ferramenta de *marketing*, com um potencial enorme podendo atingir usuários do mundo todo, de uma forma disruptiva e eficaz.

De acordo com Convertte (2014), esse é um canal robusto por representar um novo estilo de vida, de individualidade, conectividade e mobilidade. O marketing em canais como *tablets* e smartphones oferece oportunidades únicas,

já que normalmente este é um momento de total atenção do usuário, onde ele está em busca de entretenimento, informação ou serviço.

Além disso, os populares *apps* podem ser utilizados também como forma de lazer, através de jogos e redes sociais. A tecnologia trás com ela um amplo espaço para inovações.

7 TRABALHOS CORRELATOS

Para viabilizar este trabalho foram pesquisados alguns trabalhos na mesma área, porém não foi encontrado uma grande quantidade destes. Os itens encontrados são aplicações simples, que utilizam o conceito de internet das coisas, para a comunicação de aparelhos com objetos do nosso cotidiano, que auxiliaram no desenvolvimento do trabalho proposto, sendo assim está seção tem o objetivo de apresentar os trabalhos relacionados com a presente pesquisa.

7.1 GREATROOM: UMA APLICAÇÃO ANDROID BASEADA EM PROXIMIDADE PARA A CRIAÇÃO DE SALAS VIRTUAIS INTELIGENTES

No ano de 2016, foi desenvolvido pelos acadêmicos Ticianne Darin, Jefferson Barbosa, Belmondo Rodrigues e Rossana M. C. Andrade um Trabalho de Conclusão de Curso pela Universidade Federal do Ceará, uma aplicação com o conceito de Internet das Coisas onde é possível a criação de salas virtuais pela proximidade de aparelhos móveis.

GreatRoom é uma aplicação Android *freeware*, baseada em proximidade, para a criação de salas virtuais no contexto de Smart Offices. Estas salas são definidas a partir da proximidade dos usuários com beacons e tem como objetivos compartilhar arquivos e gerar a lista de presença dos participantes de uma reunião (DARIN et al., 2016).

7.2 TÔ AQUI: APLICATIVO PARA GEORREFERENCIAMENTO EM AMBIENTES RESTRITOS

Na Universidade Regional de Blumenau, o acadêmico Diego Tondim Rocha, desenvolveu uma aplicação no ano de 2015, cujo objetivo era o aprendizado sobre o georreferenciamento e localização via beacons em ambientes restritos.

Tô aqui é um aplicativo que visa aprimorar o georreferenciamento em ambientes restritos permitindo ao usuário locomover-se pelos ambientes e visualizar a sua localização através de um mapa, disponibilizado em dispositivos móveis com os sistemas operacionais Android e iOS. O aplicativo exibe um mapa do Google

Maps e executa a integração com o AGPS e os Beacons para determinar a localização do usuário (ROCHA, 2015).

7.3 DESENVOLVIMENTO DE UMA APP ANDROID E PLATAFORMA WEB PARA COMUNICAÇÃO COM BEACONS

Em 2016, Nuno Alexandre Lemos de Paiva realizou uma pesquisa sobre desenvolvimento de uma APP Android e Plataforma Web para comunicação com Beacons pelo Instituto Superior de Engenharia de Coimbra (ISEC) para a obtenção do grau de Mestre em Informática e Sistemas, onde a pesquisa visava à comunicação e troca de informações entre aplicações webs e dispositivos móveis através do uso de Beacons.

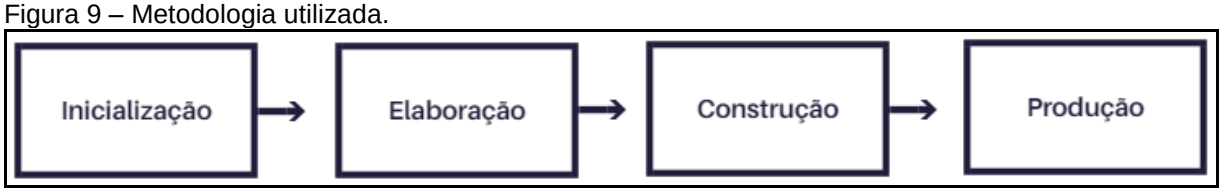
A solução tem como principais objetivos melhorar a experiência da comunicação em tempo real com o cliente, bem como trazer ao retalhista um conjunto de dados de forma a traçar o perfil de cada cliente (o que possibilitará adaptar mais eficazmente as suas campanhas), potenciando o seu crescimento económico (PAIVA, 2016).

8 DESENVOLVIMENTO DO PROTÓTIPO

O trabalho tem como objetivo proporcionar a criação de uma ferramenta computacional, para o auxilio de praticantes de exercícios físicos no ambiente de academia, onde será possível acompanhar a sua ficha de treino, além de exibir tutorias de como executar o movimento no aparelho, que será exibido assim que o usuário se aproximar do mesmo.

A metodologia de pesquisa é constituida por quatro etapas bem definidas, conforme exposto na figura 9:

Figura 9 – Metodologia utilizada.



Fonte: Do autor.

8.1 INICIALIZAÇÃO

Com a popularidade dos dispositivos móveis, foi realizado um levantamento de requisitos através de pesquisas bibliográficas para disponibilizar uma ferramenta capaz de auxiliar indivíduos na prática de exercícios físicos no ambiente de academias, por meio de um manual digital gamificado ou através de vídeos explicativos.

Com isso foi levantado alguns requisitos funcionais para dar início na elaboração do protótipo. Assim foi levantado todas a telas da aplicação e o fluxo de cadastro e além do fluxo entre as telas.

8.2 ELABORAÇÃO

Com o levantamento dos requisitos, se tornou importante a criação da modelagem de dados, para auxiliar em uma visão geral do contexto para construir a ferramenta. Após está etapa, ficou explícito a necessidade de escolher as tecnologias a serem utilizadas para a criação da aplicação.

Por uma melhor organização de código e separação de responsabilidades, foi decidido que seriam criados dois projetos destintos, um responsável pela parte de negócio e outro pela parte visual da aplicação, respectivamente um projeto *back-end* e outro *front-end*.

O projeto *back-end* será desenvolvido sobre o conceito de *webservice*, separando as responsabilidades e rotinas, sendo assim cada rotina terá um *endpoint* específico que será responsável por apenas uma funcionalidade, como por exemplo criar novos registros ou atualizar os mesmos quando necessário. As tecnologias escolhidas para o *back-end* foram as seguintes: Jersey 1.19.1; JAX-RS 1.1.1.

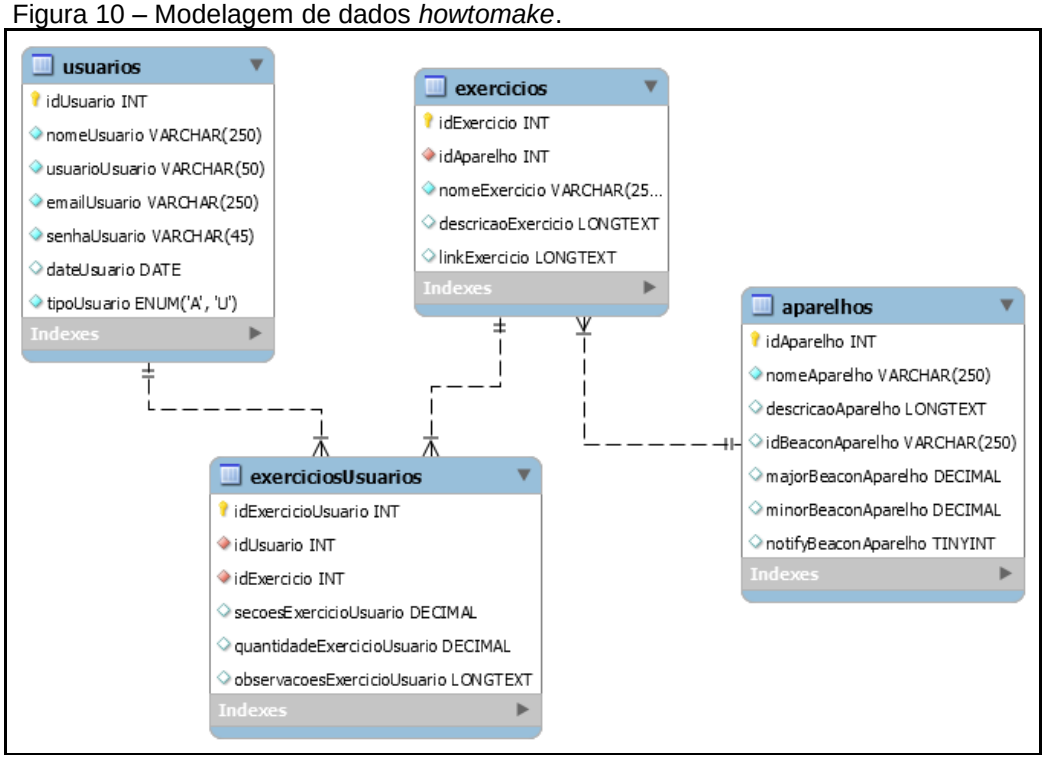
Para o projeto *front-end* será utilizado o padrão MVC, separando as responsabilidades em modelos, controles responsáveis pelas lógicas, *views* responsável por exibir e capturar informações dos usuários. As tecnologias utilizadas são: AngularJs 1.5.3; Ionic Pro 1.0.20; Anguar Ui Router 0.2.13.

A comunicação entre as duas aplicações se dará por meio de chamadas REST. Devido ao projeto BE ter sido desenvolvido no padrão *webservice*, ele pode ser consumido por diversas aplicações, não sendo necessário toda uma estrutura robusta, basta apenas enviar as informações que os *endpoints* estão esperando receber, sendo assim a aplicação FE conseguirá consumir os recursos disponíveis de um modo mais ágil e facíl.

8.2.1 Modelagem de dados

Para se obter uma visão mais técnica dos requisitos necessários e dar início na criação do *back-end* e no *protótipo*, foi criado o modelo de dados conforme a figura 10.

Figura 10 – Modelagem de dados *howtomake*.



Fonte: Do autor.

O modelo lógico serve para mostrar as ligações entre as tabelas no banco de dados. Com isso foi possível dar início no desenvolvimento em ambos os projetos, auxiliando na diminuição de erros de programação e dando uma visão mais ampla de como deve se comportar a aplicação.

8.3 CONSTRUÇÃO

Após a definição do modelo de dados, o próximo passo foi dar início nas aplicações *back-end* e *front-end*. A configuração dos ambientes e estrutura dos projetos são explicados e exemplificados nos proxímos tópicos.

8.3.1 Configuração do servidor de aplicação e criação do projeto BE

Para o desenvolvimento do projeto foi utilizado a IDE Netabens, que é disponibilizada gratuitamente no site do fabricante. A IDE foi escolhida devido a possuir integração com diversos servidores, como Tomcat e Glassfish e facilitar na criação de projetos em diversas linguagens, além de ser a ferramenta utilizada em aulas durante o curso, tendo assim um domínio maior sobre a mesma.

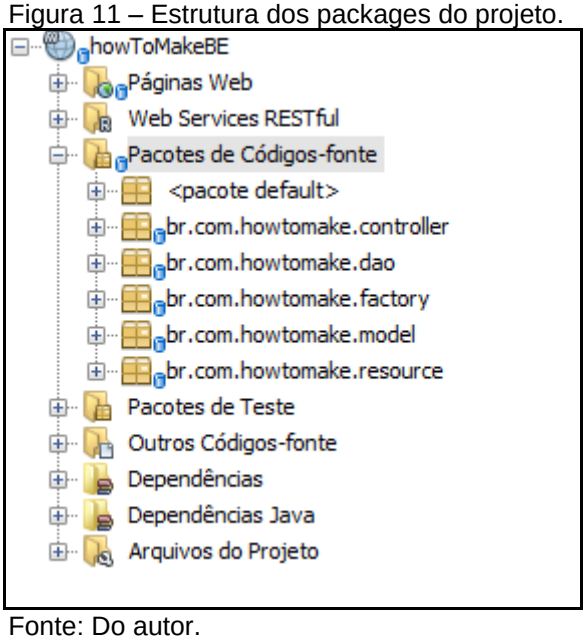
A criação do projeto foi feita através da IDE do Netbeans, utilizando a opção novo projeto, Maven e em seguida selecionar aplicação web, que iniciará um passo a passo auxiliando na configuração do projeto, como escolha do servidor e a versão do JAVA. O projeto é criado com o gerenciador de dependências do Maven, que auxilia na instalação de dependências e manutenção do projeto. Para o servidor do projeto foi optado pelo Apache Tomcat 7, devido ser de facíl utilização e desempenho agradável.

Após a criação do projeto é possível visualizar o arquivo *pom.xml*, nele está contido todas as informações de configurações do Maven para a configuração do projeto, como dependencias a serem baixadas e informações do projeto.

8.3.1.1 Início do desenvolvimento JAVA

Após toda a parte de configuração do projeto, demonstrada a cima, foi dado inicio no desenvolvimento. O primeiro passo foi de estruturar o projeto, criando os *packages*, ficando conforme a figura 11.

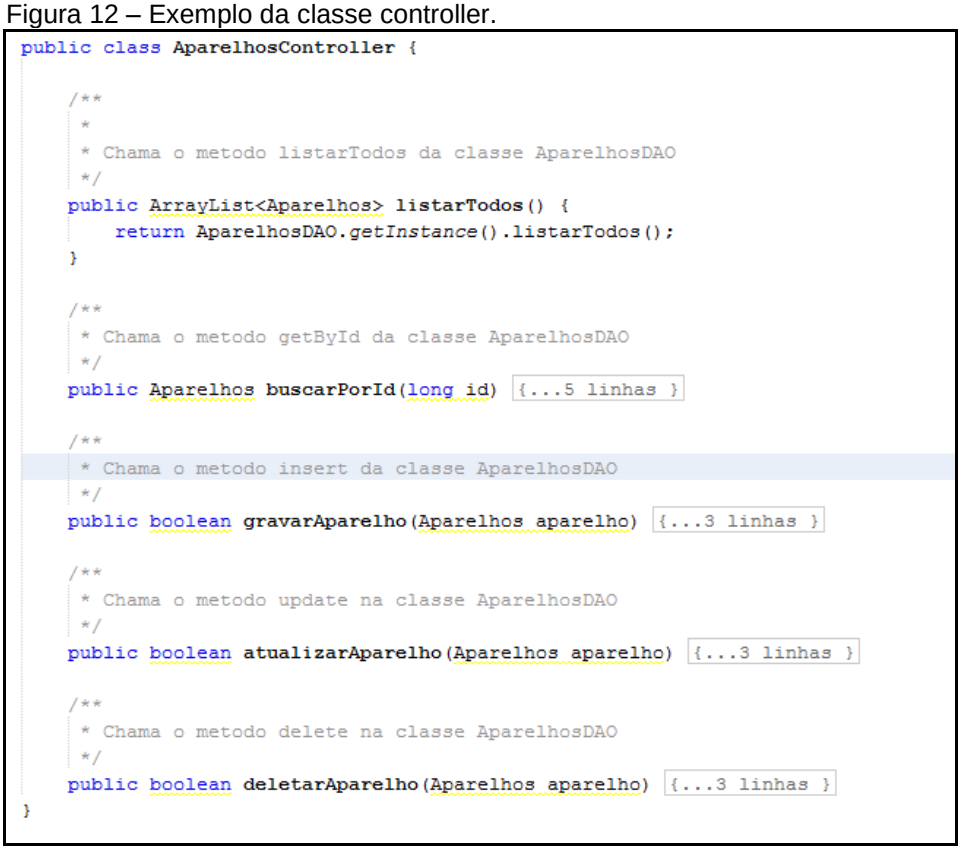
Figura 11 – Estrutura dos packages do projeto.



Fonte: Do autor.

O projeto foi dividido em cinco *packages,* neles estão contidas todas as classes JAVA do projeto. O *package br.com.howtomake.controller* é responsável por amarzenar todos os *controllers*, um controle tem literalmente a função de controlar a aplicação (figura 12).

Figura 12 – Exemplo da classe controller.



```
public class AparelhosController {

    /**
     *
     * Chama o metodo listarTodos da classe AparelhosDAO
     */
    public ArrayList<Aparelhos> listarTodos() {
        return AparelhosDAO.getInstance().listarTodos();
    }

    /**
     * Chama o metodo getById da classe AparelhosDAO
     */
    public Aparelhos buscarPorId(long id) {...5 linhas }

    /**
     * Chama o metodo insert da classe AparelhosDAO
     */
    public boolean gravarAparelho(Aparelhos aparelho) {...3 linhas }

    /**
     * Chama o metodo update na classe AparelhosDAO
     */
    public boolean atualizarAparelho(Aparelhos aparelho) {...3 linhas }

    /**
     * Chama o metodo delete na classe AparelhosDAO
     */
    public boolean deletarAparelho(Aparelhos aparelho) {...3 linhas }
}
```

Fonte: Do autor.

Controller tem a função de controlar a comunicação dos *resources* com a camada DAO que veremos a seguir. Neles se encontram todos os métodos que serão consumidos pelos *resources*, como buscarPorId, que deve receber um id e retornar o aparelho com o id específicado.

No *package br.com.howtomake.dao* estão contidas todas as classes DAO, elas são encarregadas por todo o CRUD da aplicação, efetuando consultas ou inserindo registros no banco de dados (figura 13).

Figura 13 – Exemplo da classe DAO.

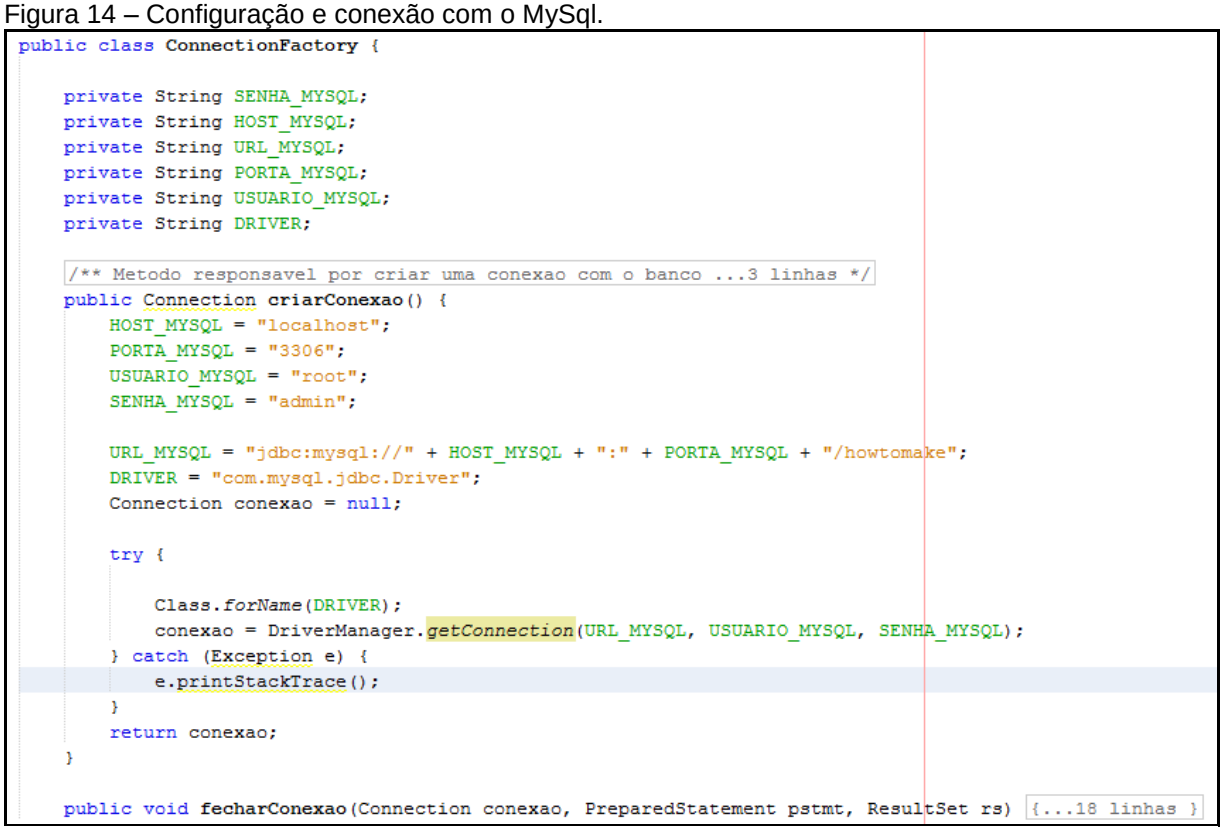
```
public class AparelhosDAO extends ConnectionFactory {

    private static AparelhosDAO instance;

    /** Metodo responsavel por criar uma instancia da classe AparelhosDAO ...3 linhas */
    public static AparelhosDAO getInstance() {...6 linhas }

    /**...5 linhas */
    public ArrayList<Aparelhos> listarTodos() {
        Connection conexao = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;
        ArrayList<Aparelhos> aparelhos = null;

        conexao = criarConexao();
        aparelhos = new ArrayList<Aparelhos>();
        try {
            pstmt = conexao
                    .prepareStatement("SELECT * FROM aparelhos ORDER BY nomeAparelho");
            rs = pstmt.executeQuery();
```

Fonte: Do autor.

Todas as classes DAO extendem a classe *ConnectionFactory* que é responsável por toda a configuração de banco de dados, como criar uma conexão com o banco e fechar, veremos sobre ela a frente. Para a criação dos SQL é utilizado o *prepareStatement* que seria como um ajudante na criação dos scripts. Após a criação do SQL, é necessário executar ele através do *executeQuery*. Todos estes métodos pertencem a API do *Java Database Connectivity* (JDBC).

O *package br.com.howtomake.factory* é designado por alocar todos os serviços em comum da aplicação. Nele temos a classe *ConnectionFactory* que é responsável por criar a conexão com o banco de dados *MySql* e também encerrar a mesma. Para criar a conexão e configurá-la, foi utilizado a API do JDBC, a figura 14 mostra os detalhes da configuração.

Figura 14 – Configuração e conexão com o MySql.



```
public class ConnectionFactory {

    private String SENHA_MYSQL;
    private String HOST_MYSQL;
    private String URL_MYSQL;
    private String PORTA_MYSQL;
    private String USUARIO_MYSQL;
    private String DRIVER;

    /** Metodo responsavel por criar uma conexao com o banco ...3 linhas */
    public Connection criarConexao() {
        HOST_MYSQL = "localhost";
        PORTA_MYSQL = "3306";
        USUARIO_MYSQL = "root";
        SENHA_MYSQL = "admin";

        URL_MYSQL = "jdbc:mysql://" + HOST_MYSQL + ":" + PORTA_MYSQL + "/howtomake";
        DRIVER = "com.mysql.jdbc.Driver";
        Connection conexao = null;

        try {

            Class.forName(DRIVER);
            conexao = DriverManager.getConnection(URL_MYSQL, USUARIO_MYSQL, SENHA_MYSQL);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return conexao;
    }

    public void fecharConexao(Connection conexao, PreparedStatement pstmt, ResultSet rs) {...18 linhas }
```

Fonte: Do autor.

A classe contem os métodos *criarConexao* e *fecharConexao*. O primeiro método é responsável por estabeler a conexão e repassar para as classes DAO. O segundo método é responsável por fechar a conexão.

O *package br.com.howtomake.model* contem todos os modelos, entidades da aplicação, todos os objetos representacionais com que a aplicação ira controlar e gerenciar dados, cada modelo representa uma tabela do banco, quais aributos ela tem e quais os tipos de dados suportados por ela (figura 15).

Figura 15 – Exemplo de entidade.

```
public final class Aparelhos {
    private Integer id;
    private String nome;
    private String descricao;
    private String identificadorBeacon;
    private String idBeacon;
    private Integer majorBeacon;
    private Integer minorBeacon;
    private boolean notifyBeacon;

    public Aparelhos() {...2 linhas }

    public Aparelhos(Integer id, String nome, String descricao, String identificadorBeacon

    public Integer getId() {...3 linhas }

    public void setId(Integer id) {...3 linhas }

    public String getNome() {...3 linhas }

    public void setNome(String nome) {...3 linhas }

    public String getDescricao() {...3 linhas }

    public void setDescricao(String descricao) {...3 linhas }

    public String getIdentificadorBeacon() {...3 linhas }

    public void setIdentificadorBeacon(String identificadorBeacon) {...3 linhas }
```

Fonte: Do autor.

Todas as entidades possuem seus atributos e tipos de dados, além de possuir dois contrutores, um padrão e outro recebendo todas as informações para instanciar um novo objeto. Além disso possuem os metodos *getters* e *setters* que são necessarios para buscar e setar as informações das propriedades das entidades respectivamente.

O ultimo *package* é o *br.com.howtomake.resource* nele contem todos os serviços da aplicação, estes serviços serviram como uma espécie de ponte para a comunicação entre o projeto BE e FE. Para criar os serviços foi utilizado a bliblioteca *Jersey*, que é uma implementação do padrão REST.

Em um primeiro momento foi necessário configurar o *Jersey*, no arquivo *web.xml* da aplicação, para que o *Servelet* do *Jersey* consiga gerenciar os serviços criados (figura 16).

Figura 16 – Configuração do Jersey.

```
<servlet>
        <servlet-name>Jersey REST Service</servlet-name>
        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
        <init-param>
                <param-name>com.sun.jersey.config.property.packages</param-name>
                <param-value>br.com.howtomake.resource</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
        <servlet-name>Jersey REST Service</servlet-name>
        <url-pattern>/*</url-pattern>
</servlet-mapping>
```

Fonte: Do autor.

No parametro *com.sun.jersey.config.property.packages* é especificado onde estarão os serviços e na configuração do *Servelet* foi definida a URL de como acessar estes serviços, nesse caso através do "/*", sendo assim todos os diretórios.

Depois de configurado, foi criado cada serviço no pacote definido na configuração. Todos os serviços possuem um caminho, que é identificado através da notação *@path,* que será utilizada na hora de consumir esse serviço na aplicação FE. Também contem o tipo de requisição que é feito para acessar, podendo ser as seguintes notações *@POST, @GET, @PUT, @DELETE* entre outros. Além disso também é especificado o tipo de mídia que o *endpoint* irá consumir e produzir, sendo respectivamente configurados pelas notações *@Consumes* e @Produces (figura 17).

Figura 17 – Exemplo de classe resource.

```
@Path("/aparelhos")
public class AparelhosResource {
        @GET
        @Path("/listarTodos")
        @Produces(MediaType.APPLICATION_JSON)
        public ArrayList<Aparelhos> listarTodos() {...3 linhas }

        @GET
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/id/{id}")
        public Response getById(@PathParam("id") Long id) {...10 linhas }

        @POST
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/salvar")
        public Response salvarAparelhoJson(Aparelhos aparelho) {...11 linhas }

        @PUT
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/atualizar")
        public Response altualizarAparelho(Aparelhos aparelho) {...11 linhas }

        @POST
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/deletar")
        public Response deletarAparelho(Aparelhos aparelho) {...15 linhas }
}
```

Fonte: Do autor.

Na figura 17, existem cinco *endpoints* que poderam ser consumidos futuramente pela aplicação FE. O serviço acima será usado para salvar os dados dos aparelhos, para salvar um registo será enviado um *POST* para a URL "*/aparelhos/salvar"*, está URL estará esperando por uma midia do tipo *JSON* e nos retornará um *JSON* com a resposta.

8.3.2 Criação do projeto FE

Após finalização do projeto BE, assim liberando os recursos necessários pela aplicação FE, pode se dar inicio no desenvolvimento da proxíma etapa. Para o desenvolvimento do projeto foi utilizado a IDE *Visual Code*, que é disponibilizada gratuitamente no site do fabricante.

Para o desenvolvimento FE foi decidido utilizar *IONIC*, que é uma ferramenta *Cross Platform,* onde pode ser utilizado linguagens WEB conhecidas, desenvolvendo toda a aplicação com base nos conhecimentos WEB. *IONIC* tem suporte ao cordova, que nos permite acessar recursos nativos do dispositivo, como Câmera, *Bluetooth* entre outros.

Em um primeiro momento, foi instalado o gerenciador de pacotes NPM, que nos ajudará na instalação de algumas dependências. Para instalar o NPM, basta acessar o site do fabricante, que disponibiliza um executavel que instala o NPM e NodeJs.

O Segundo passo, após a instalação dos itens anteriores, é instalar o IONIC através do NPM em um contexto global (figura 18).

Figura 18 – Instalação do IONIC.

```
npm install -g cordova ionic
```

Fonte: Do autor.

Após a instalação do IONIC, ele fornecerá uma CLI com uma alta variedade de opções, que auxiliará na criação de novas aplicações e instalações de dependências. O próximo passo foi criar a aplicação FE (figura 19).

Figura 19 – Criando o projeto howToMakeFE.

```
ionic start howToMakeFE sidemenu --type ionic1
```

Fonte: Do autor.

Finalizado a criação do projeto, que levou alguns minutos, é necessário adicionar as plataformas que o aplicativo dará suporte, isso pode ser feito com ajuda da CLI e com o *Cordova* (figura 20).

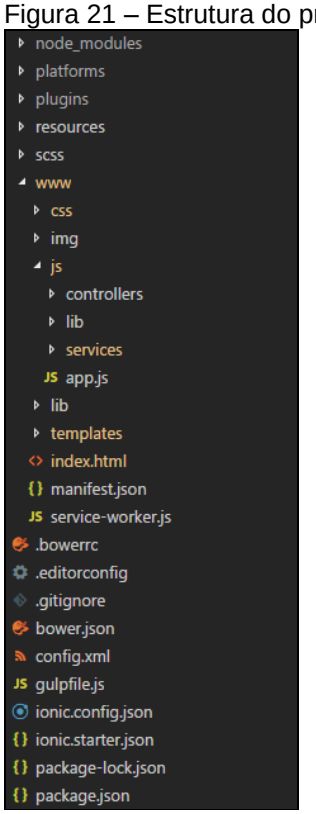
Figura 20 – Adicionando as plataformas.

```
ionic cordova platform add android
ionic cordova platform add ios
```

Fonte: Do autor.

Os passos acimas exemplificam a configuração do projeto, sendo possível dar inicio no desenvolvimento do projeto, o resultado final da estrutura do projeto ficou conforme a figura 21.

Figura 21 – Estrutura do projeto FE.



Fonte: Do autor.

Para o desenvolvimento das telas foi utilizado o *framework AngularJS* e HTML, tornando o desenvolvimento ágil e rápido, ganhando em produtividade. Para desenvolver uma aplicação em *AngularJS* é necessário identificar que a mesma será uma aplicação *AngularJS* através do *ng-app=”howToMake”,* onde o *howToMake* é o nome da aplicação, e também o nosso módulo principal.

Com a ajuda das tags *<ion-nav-view></ion-nav-view>* é possível manipular as páginas para que utilizem o conceito de menu lateral, onde definimos como a página principal o menu, e as demais páginas como subpáginas do menu.

Ui-Router é um *LIB* gratuita disponibilizada para aplicações *AngularJS*, que servem para configurar as rotas de uma maneira fácil e ágil. O *State* define como será chamada a rota, URL como acessar a página pelo navegador, *templateUrl*, é utilizado para definirmos o *template* que será usado para renderizar a página quando acessarmos a rota definida, *controller* é utilizado para definirmos quem será responsável por manter e transitar os dados no contexto definido (figura 22).

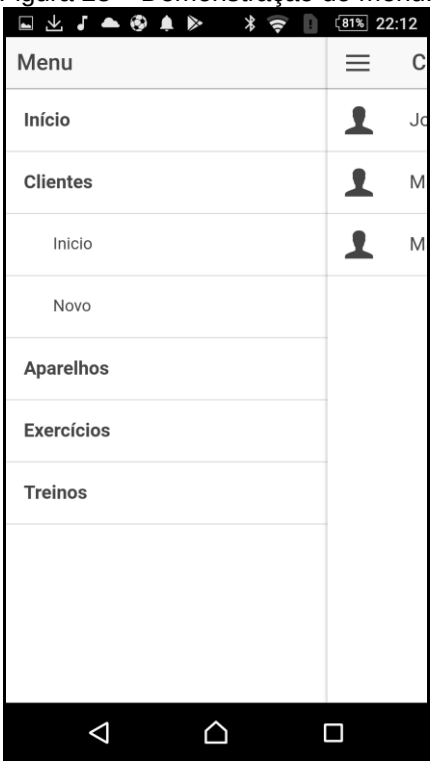
Figura 22 – Definições de rotas com *Ui-Router*.

```
$stateProvider
  .state('app', {
    url: '/app',
    abstract: true,
    templateUrl: 'templates/menu.html',
    controller: 'AppCtrl',
    controllerAs: '$ctrl'
  })
  .state('app.clientes', {
    url: '/clientes',
    views: {
      'menuContent': {
        templateUrl: 'templates/clientes.html',
        controller: 'ClientesCtrl',
        controllerAs: '$ctrl'
      }
    }
  })
  .state('app.novoCliente', {···
  })
```

Fonte: Do autor.

As demais rotas serão subrotas do menu, que seguem o mesmo padrão citado anteriormente, com o diferencial de possuirem o atributo *views*, e possuem a view *menuContent*, que defini que conteúdo será exibido dentro dá view (figura 22).

Figura 23 – Demonstração do menu.

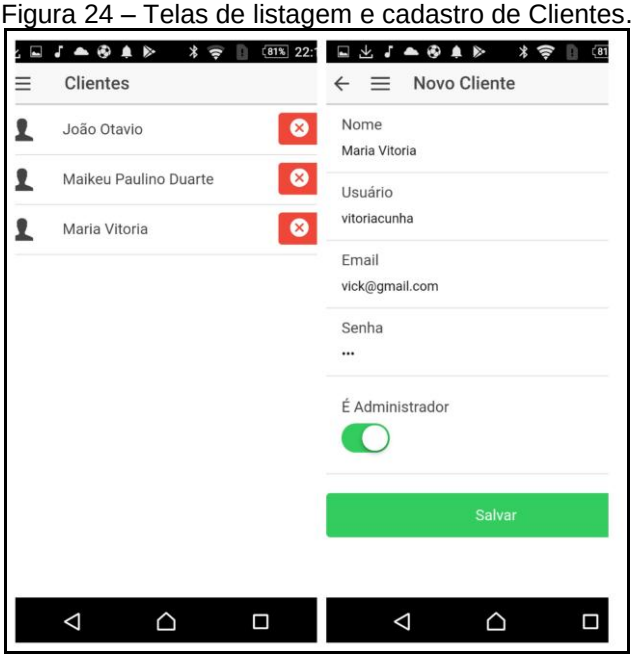


Fonte: Do autor.

Estão disponíveis dois menus, a demonstrada na figura 23 é o menu utilizado por usuários administradores, onde cada item do menu possui dois subitens, Ínicio onde é exibida a listagem e Novo por onde é feita a entrada de novos registros para a aplicação.

A figura 24 demonstra a tela de listagem de clientes, onde é possível excluir registros e também a criação de novos registros, tanto administradores como usuários da aplicação. Os campos Usuário e Senha serão utilizados para acessar a aplicação.

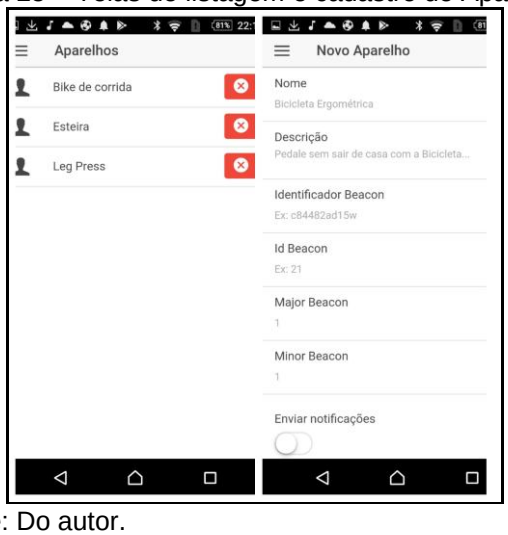
Figura 24 – Telas de listagem e cadastro de Clientes.



Fonte: Do autor.

Um dos itens mais importantes da aplicação é o cadastro de aparelhos, conforme a imagem 24 demostra a listagem de aparelhos e também a parte de novos aparelhos, onde é possível observar os dados de configurações dos Beacons. *identificador*, que é um código único por beacon, *Id Beacon (Region)* que é um ID de região, que especifica um tipo de rede de beacons, onde você pode fazer vários beacons trabalhar em uma mesma região ou em regiões diferentes, e por último os atributos *Major* e *Minor*, que são únicos de *ibeacon* para *ibeacon*, que são uados para diferenciar os *ibeacons*, por meio de localização ou conforme desejado (figura 25).

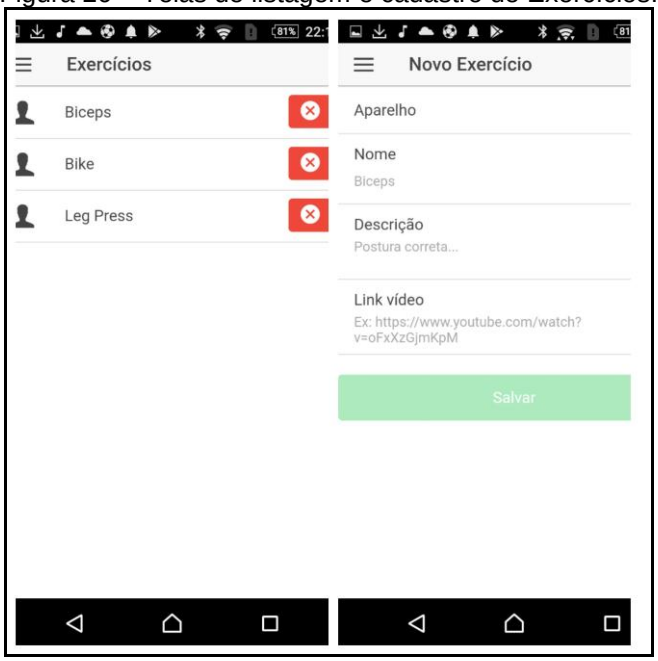
Figura 25 – Telas de listagem e cadastro de Aparelhos.



Fonte: Do autor.

A figura 26 demonstra as telas de cadastro de exercícios e listagem do mesmo, ao escolher o aparelho, automaticamente esse exercício adota as configurações dos beacons, devido as configurações serem por aparelhos e um único aparelho pode ser utilizado para a pratica de inúmeros exercícios.

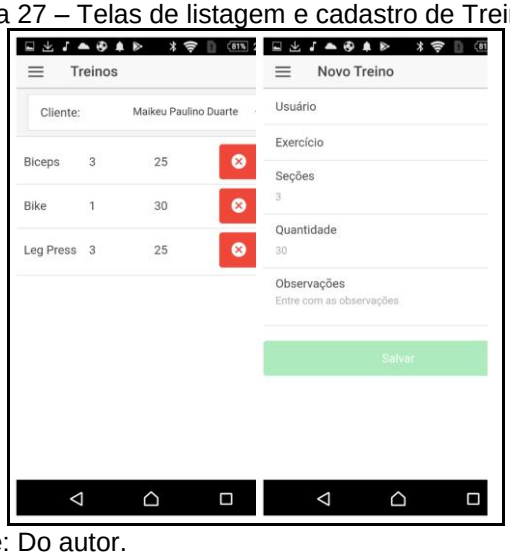
Figura 26 – Telas de listagem e cadastro de Exercícios.



Fonte: Do autor.

A figura 27 demostra as telas de listagem e cadastro de treinos para os usuários. Para exibir os treinos do usuário é necessário selecionar e em seguida será exibido o seu treino.

Figura 27 – Telas de listagem e cadastro de Treinos.



Fonte: Do autor.

Todas as telas citadas anteriormente, possuem um *controller*, um *service* e seus respectivos *templates*. A figura 28 demonstra um *controller*, o de listagem de dados da tela de aparelhos, que é semelhante em todas as demais telas.

Figura 28 – *Controller* da tela de listagem de aparelhos.

```
angular.module('starter.controllers.aparelhos', [])

    .controller('AparelhosCtrl', ['aparelhoService', 'loaderService', '$state', '$ionicPopup',
    function (aparelhoService, loaderService, $state, $ionicPopup) {
        var that = this;
        that.init = function () {
            that.listaAparelhos = [];

            loaderService.show();
            aparelhoService.getAll().then(function (result) {
                if(result.data){
                    if(angular.isArray(result.data.aparelhos)){
                        that.listaAparelhos = result.data.aparelhos;
                    }else{
                        that.listaAparelhos.push(result.data.aparelhos);
                    }
                }
            }).finally(function () {
                loaderService.hide();
            });
        };
        that.init();

        that.editar = function (id) {···
        };

        that.remover = function (aparelho) {···
        };
    }]);
```

Fonte: Do autor.

O *AngularJS* tem uma ferramenta muito forte que é a parte de injeção de dependências, onde você pode dividir o seu código em modulos, e depois injetalos e reusalos em toda a aplicação. Nesse *controller* estamos injetanto *aparelhoService*, que é responsavel por todas as nossas chamadas *REST(POST, GET, DELETE, PUT)*, em seguida o nosso serviço de *loader*, que é responsavel por colocar o *loading* nas páginas, *$state* responsável de controlar as nossas rotas, conforme vimos anterioremente, *$ionicPopup* um componente que o IONIC nos fornece, que tem algumas funcionalidades interessantes, no nosso contexto foi usado para a confirmação ao excluir registros.

A figura 29 exemplifica um *service*, o serviço abaixo é o responsável pela tela de aparelhos, neles estão todos os métodos que são responsáveis por salvar um registro, atualizar, deletar, listar todos e também buscar um registro específico. Para executar as chamadas dos métodos foi utilizado o serviço *$http,* que é um serviço fornecido pelo *AngularJS.*

Figura 29 – Service Aparelhos.

```
angular.module('starter.services.aparelhos', [])
    .service('aparelhoService', aparelhoService);

aparelhoService.$inject = ['$http', '$q'];

function aparelhoService($http, $q) {
    var baseService = 'http://192.168.0.102:8084/howToMakeBE/aparelhos';
    this.save = function (aparelho) {
        return $http.post(baseService + '/salvar', aparelho);
    }

    this.put = function (aparelho) {
        return $http.put(baseService + '/atualizar', aparelho);
    }

    this.delete = function (aparelho) {
        return $http.post(baseService+'/deletar', aparelho);
    }

    this.getAll = function () {
        return $http.get(baseService + '/listarTodos');
    }

    this.get = function (id) {
        return $http.get(baseService + '/id/' + id);
    }
}
```

Fonte: Do autor.

As tags *<ion-view></ion-view>* demostram o conteúdo que será exibido entre o menu, *<ion-nav-buttons></ion-nav-buttons>* demarcam os ícones que apareceram no menu, como voltar para a tela anterior entre outros. A tag *<ion-content></ion-content>* exibem o conteúdo principal da tela, como campos de entrada ou listagem de dados (figura 30).

Figura 30 – HTML da tela de listagem de aparelhos.

```
<ion-view title="Aparelhos">
    <ion-nav-buttons side="left">
        <button menu-toggle="left" class="button button-icon icon ion-navicon"></button>
    </ion-nav-buttons>
    <ion-content class="has-header">
        <div class="list">
            <div class="item item-button-right" ng-repeat="aparelho in $ctrl.listaAparelhos">
                <div class="item-icon-left" ng-click="$ctrl.editar({{aparelho.id}})">
                    <i class="icon ion-person"></i>
                    {{aparelho.nome}}
                </div>
                <button class="button button-assertive" ng-click="$ctrl.remover({{aparelho}})">
                    <i class="icon ion-android-cancel"></i>
                </button>
            </div>
        </div>
    </ion-content>
</ion-view>
```

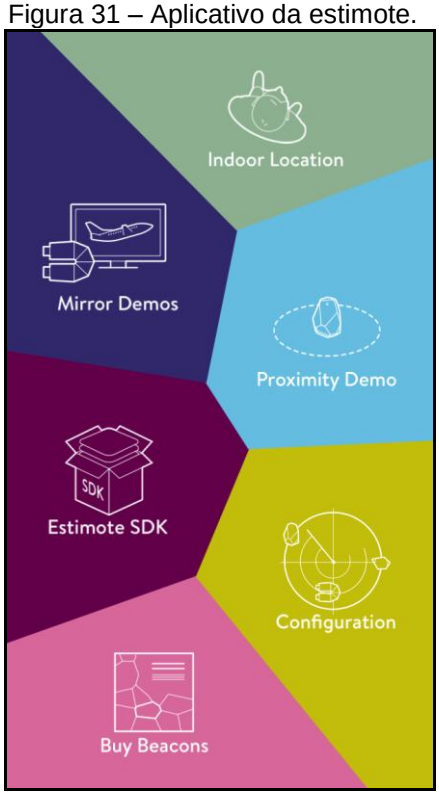
Fonte: Do autor.

Com a utilização das diretivas do *AngularJS* tornou-se mais fácil a criação de rotinas como listagem de dados, editar e excluir registros. O atributo *ng-repeat* nos da a possibilidade de criar uma lista de registros com pouco trecho de código, onde ele recebe um *array* e percorre todos os itens tornando o trabalho mais fácil e ágil. O atributo *ng-click* é usado para executar eventos quando o usuário clica em algum elemento com o atributo, na imagem acima ele foi utilizado para a edição de registros e exclusão.

8.3.2.1 Configuração e identificação dos *iBeacons*

Após a construção da aplicação *front-end*, foi dado inicio na utilização dos beacons na aplicação, o primeiro passo foi efetuar a configuração dos beacons. Os dispositivos utilizados são da estimote, uma distribuidora desse tipo de equipamento. A configuração dos beacons foi feita atraves da ferramente disponibilzada pela estimote tanto para iOS como Android (figura 31).

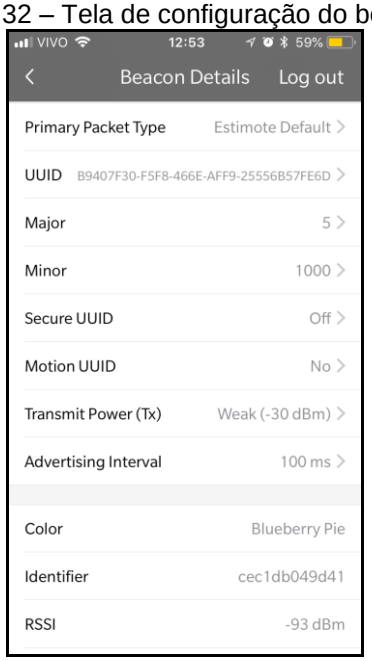
Figura 31 – Aplicativo da estimote.



Fonte: Do autor.

Após a instalação do aplicativo, foi dado inicio na configuração, para realizar esta estapa foi acessado o item de configurações(*Configuration*) do APP que em seguida demonstra todos os beacons nas localidades proxímas e escolhendo o beacon desejado, em seguida é demonstrado a tela de configuração do *Beacon* (figura 32).

Figura 32 – Tela de configuração do beacon.



Fonte: Do autor.

Conforme a figura 32, a aplicação disponibiliza uma serie de opções para a configuração dos beacons. O Item *UUID* é responsável por ser o identificador da rede, como já mencionado antes. *Major* e *Minor* são os responsáveis pela a identificação de uma determinada região através dos *beacons*, geralmente utilizada para ambientes com uma área significativa. *Identifier* é o identificador único de cada *beacon*, com ele pode se diferenciar um beacon do outro. A ferramente disponibiliza uma serie de protocolos para a configuração dos beacons, nesse trabalho foi utilizado o protocolo do *iBeacon* que é recomendada pela estimote.

O proxímo passo foi configurar a aplicação para dar suporte aos *Beacons*, para tal comportamento foi adicionado o plugin do *Cordova* para *Beacons* (figura 33).

Figura 33 – Instalação do plugin do cordova iBeacon.

```
cordova plugin add https://github.com/petermetz/cordova-plugin-ibeacon.git
```

Fonte: Do autor.

Foi utilizado um *wrapper* para *AngularJS* que está disponivel gratuitamente na internet e de código aberto. Após a configuração do projeto para ter suporte a *Beacons* o proxímo passo foi a implementação do mesmo.

Para que a aplicação seja capaz de detequitar a presença dos dispositivos móveis em determinadas localizações, é necessário que você inicie o monitoramento dos *beacons* na aplicação, no contexto da aplicação cada *beacon* representa um equipamento da academia (figura 34).

Figura 34 – Criando as regiões de identificação dos Beacons para cada treino.

```
treinoService.getAll(codigo).then(function (result) {
    if (result.data) {
        if (angular.isArray(result.data.treinos)) {
            that.treinosUsuarioLogado = result.data.treinos;
        } else {
            that.treinosUsuarioLogado.push(result.data.treinos);
        }
    }
}).finally(iniciarBeacons.bind({}));

function iniciarBeacons() {
    if (that.treinosUsuarioLogado.length) {
        that.treinosUsuarioLogado.forEach(function (item) {
            var exercicio = item.exercicio;
            var identicador = exercicio.aparelho.identificadorBeacon;
            var uui = exercicio.aparelho.idBeacon;
            var major = parseInt(exercicio.aparelho.majorBeacon);
            var minor = parseInt(exercicio.aparelho.minorBeacon);

            that.beaconsUsuarioLogado[identicador] = exercicio;
            $cordovaBeacon.startMonitoringForRegion($cordovaBeacon.createBeaconRegion(identicador, uui, major, minor));
        });
    }
}
```

Fonte: Do autor.

Após o usuário logar na aplicação, é dado ínicio na busca dos treinos disponíveis do usuário e criado um *array* com os itens, após finalizar esse processo em seguida é chamado o método “iniciarBeacons” que é responsável por criar as regiões de monitoramento. O método é responsável por percorrer todos os itens da lista e também é responsável por criar um novo objeto que vincula o identificar do *Beacon* com o exercício, esse trecho de código será utilizado a frente.

Com a criação das regiões, é necessário identificar quando um dispositivo móvel entra na região e em seguida exibir o alerta com a informação de que existe um video de instruções de uso do equipamento (figura 35).

Figura 35 – Identifica quando um Beacon entra na região.

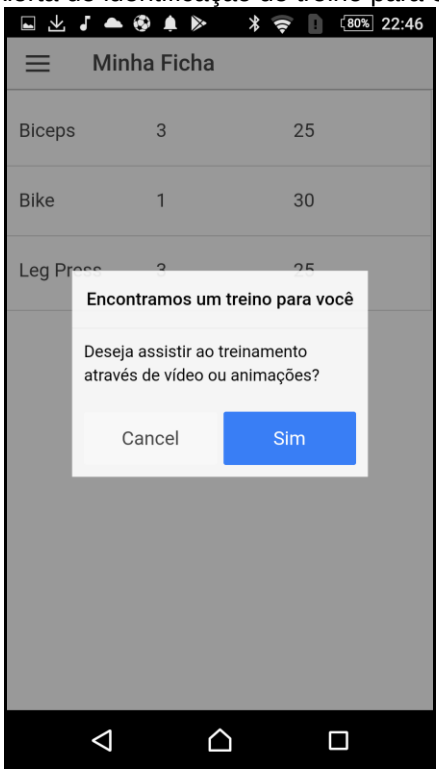
```
$rootScope.$on("$cordovaBeacon:didEnterRegion", function (event, pluginResult) {
    if (pluginResult.region.identifier != atualBeacon) {
        encontrouTreino(pluginResult.region.identifier);
    }

    atualBeacon = pluginResult.region.identifier;
    $scope.$apply();
});
```

Fonte: Do autor.

O código demostrado na figura 35 é chamado quando a aplicação identifica que um dispositivo móvel entrou em uma determinada região, o método é

incumbido de identificar se é uma nova região e então chamar o método responsável por exibir o alerta de identificação de treinos para o equipamento (figura 36).

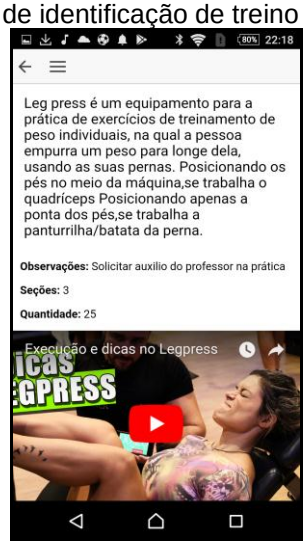
Figura 36 – Alerta de identificação de treino para equipamento.



Fonte: Do autor.

Se o usuário confirmar que deseja visualizar os vídeos e animações disponíveis para o equipamento, em seguida a aplicação abre a tela demostrando as informações do equipamento além de um vídeo autoexplicativo (figura 37).

Figura 37 – Alerta de identificação de treino para equipamento.



Fonte: Do autor.

8.3.2.2 Disponibilização da APK

Finalizado a aplicação é necessário a disponibilização da APK para que poça ser feita a instalação nos díspositivos móveis, como já foi citado anteriormente o protótipo foi produzido com a utilização do *IONIC* que é uma ferramenta *cross platform*, sendo assim é possível disponibilar a aplicação para alguns sistemas operacionais. Para fazer o *build* e gerar a aplicação é necessário ter alguns itens instalados, como o Java SDK, Android Studio, Android SDK Tools e SDK Manager.

Figura 38 – Build da aplicação para android.

```
ionic cordova run android
# or
ionic cordova build android
```

Fonte: Do autor.

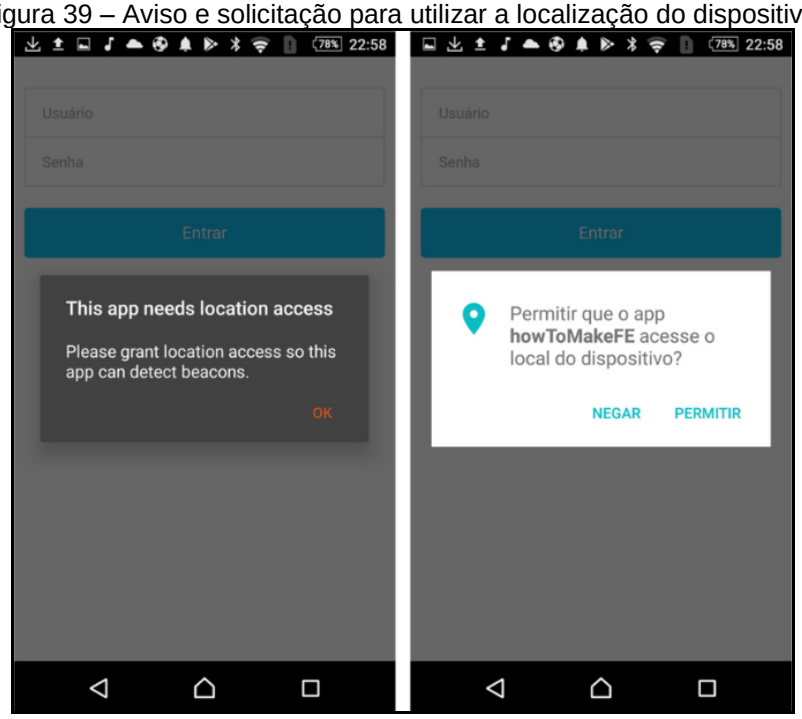
A figura 38 exemplifica como rodar a aplicação em um *device* através do cabo USB ao executar o primeiro comando, já a segunda instrução é responsável por executar o *build* da aplicação e disponibilar a APK que pode ser transferida para os dispisítivos desejados e instala-lá.

8.4 PRODUÇÃO

Após a finalização das etapas anteriores, a próxima etapa foi fazer todas as ferramentas trabalharem em conjunto, desde das aplicações *back-end* e *front-end* e a interação com os beacons. Foram simulados alguns ambientes de teste para comparar e verificar o desempenho da ferramenta.

Primeiro foi feito um teste se a aplicação estava identificando os beacons, para isso é necessário que o dispositivo permita que o protótipo use a sua localização, então ao abrir o protótipo o dispositivo solicita a permissão para utilizar esse recurso (figura 39).

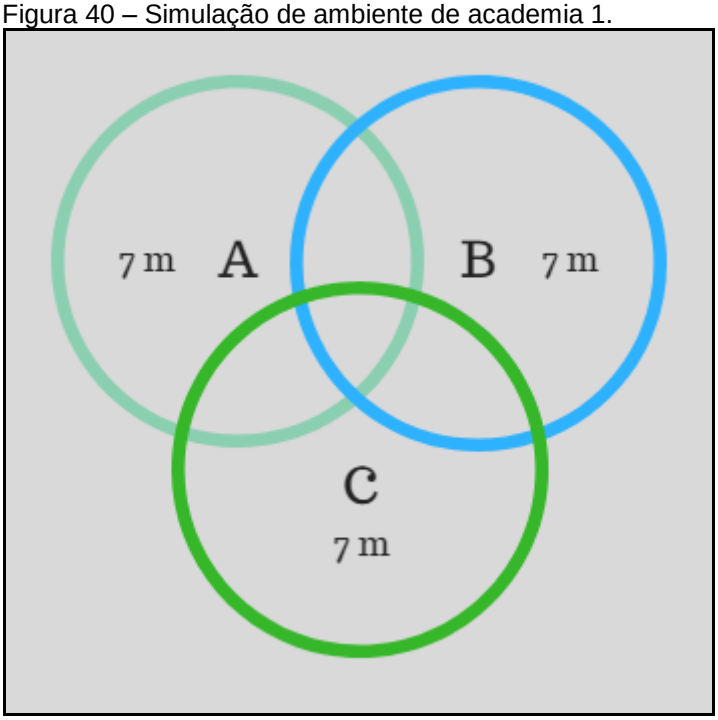
Figura 39 – Aviso e solicitação para utilizar a localização do dispositivo.



Fonte: Do autor.

Com o sucesso na primeira etapa, foi configurado os beacons para emitirem o sinal em um raio de 7 metros para se realizar os primeiros testes, os beacons foram dispostos em locais estratégicos nos equipamentos, o primeiro teste foi com aparelhos espalhados pelo ambiente (figura 40).

Figura 40 – Simulação de ambiente de academia 1.



Fonte: Do autor.

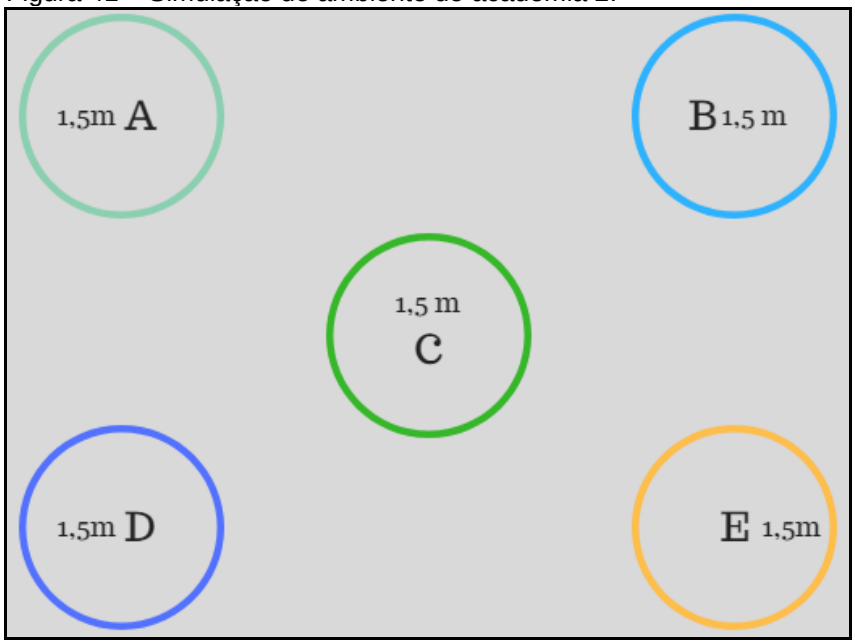
A figura 40 foi o primeiro caso de teste, os resultados não foram satisfatórios, cada letra representa um beacon e um aparelho de ginastica, na imagem podemos ver que o sinal dos beacons ficou soprepondo um ao outro, dando uma interferência de sinal, onde a aplicação não se comportou bem e exibia alertas simultaniamente.

Foi identificado que o problema se tratava do raio de emissão de sinal dos *beacons*, então foram feitos testes reduzindo a aréa de transmissão dos *beacons* para 3,5m e 1,5m respectivamenteveio, os resultados foram melhorando gradativamente.

Para o caso de teste com os beacons com transmissão de sinal para 3,5 metros já houve uma melhora significante, mas em determinados momentos ainda havia a interferência de sinal dos beacons, com uma frequencia bem inferior ao primeiro caso.

Com o terceiro caso os beacons conseguiram ter um aproveitamento de 99,99%, os alertas só eram demonstrados para determinado equipamento que possuia os beacons, mas foi possível observar que poderia ser melhorado a diposição dos equipamentos nas salas, assim melhorando o sinal dos beacons e o bom funcionamento da aplicação. Foi considerado como um bom cenário para a aplicação os aparelhos serem distribuidos conforme a figura 41.

Figura 41 – Simulação de ambiente de academia 2.



Fonte: Do autor.

A aplicação funcionou conforme o esperado, mostrando os alertas dos tutoriais somente quando o usuário adentrava na área de transmissão do *beacon*, sem houver interferência de sinal. Lembrando que a aplicação funciona também coma disposição dos equipamentos de outras formas, mas talvez possa ocorrer os problemas de interferência de sinal.

Os beacons são configurados para emitir sinal de 1,5m, o minimo possível através das configurações da estimote, dessa forma é interessante manter a distancia de no mínimo 1,5m de distancia de um equipamento para o outro, para que o protótipo funcione como desejado.

8.5 RESULTADOS OBTIDOS

Com o desenvolvimento do projeto de pesquisa deste trabalho, pôde-se obter uma aplicação de estudo para realizar a integração dos *beacons* com o conceito de Internet da Coisas, utilizando o *framework* AngularJs com o padrão MVC e Jersey para a criação dos serviços. Com isso foi possível disponibilizar uma ferramenta que funciona como um manual digital para ambientes de academia.

O desenvolvimento do protótipo foi realizado com o *framework* AngularJS, utilizando o conceito de MVC que é uma pratica recomendada para estrutura do projeto, este *framework* colaborou em grande escala para a produção da aplicação do lado *front-end*, além de diminuir o tráfico de dados da aplicação devido utilizar o conceito de *Sigle-Page-Application*, as demais tecnologias usadas foram para facilitar o desenvolvimento.

Para disponibilizar a ferramenta para os dispositivos móveis foi utilizado o IONIC, que é uma ferramenta *Cross Platform*, onde é possível programar utilizando as linguagens de programação web e no final fazer o *build* para as plataformas desejadas, como Android, iOS ou Windows.

A parte do desenvolvimento *back-end* foi desenvolvida utilizando o conceito de REST e serviços, onde é disponibilizado *endpoints* que podem ser consumidos por diversas aplicações, tornando as maleáveis e totalmente desacopladas, assim um serviço recebe os dados através de *endpoints* e os retorna quando necessários. Os resultados obtidos pela aplicação *back-end* foram simulados em conexões 3G e rede *WIFI,* todos os resultados foram satisfatórios, tendo uma resposta quase imediata dos *endpoints*.

Para se obter a localização do usuário no ambiente, foram utilizados *beacons,* os primeiros resultados foram ruins, havendo interferência de sinal, devido ao raio de rastreio dos dispositivos *beacons.* Os resultados posteriores foram positivos, tendo o resultado esperado pela aplicação, identificando a presença do dispositivo somente quando o usuário adentrava na região do *Beacon*.

Na integração dos projetos foi possível notar as vantagens em utilizar o conceito de se ter dois projetos distintos, um sendo responsável pela parte visual e de entrada de dados e outro responsável por armazená-las e consumi-las da melhor maneira possível, além de ser possível consumir estes dados em outros meios, como páginas web ou em outros protótipos.

Com a junção de todos as etapas citadas anteriormente, foi possível disponibilizar um protótipo final que possibilitou aos usuários de academia, que pratiquem suas atividades físicas nos ambientes de academia sem a ajuda de um *Personal Trainer,* utilizando apenas o protótipo como um manual autoexplicativo. O protótipo disponibilizado possui uma parte administrativa para os *Personal Trainers* e a outra parte como um manual digital.

9 CONCLUSÃO

Devido à grande procura por manter a saúde em dia e a popularização da pratica de exercícios nas academias, além da alta demanda de dispositivos móveis, o projeto de pesquisa resultou-se em disponibilizar uma ferramenta para o auxílio na prática de exercícios físicos.

Com o desenvolvimento do projeto se obteve conhecimento em diversas áreas importantes, além de se obter o conhecimento em conceitos populares utilizados na construção do protótipo, assim se obteve maior facilidade no desenvolvimento dos projetos, em ambos os projetos foram utilizados o conceito de MVC.

Como resultado final para o usuário, se teve um ganho elevado em performance devido ao AngularJs utilizar o conceito de *Single-Page-Application*, tornando a experiencia do usuário agradável e convidativa. Além do *framework* ser recomendado por desenvolvedores e possuir grande quantidade de conteúdo na internet e o mesmo utilizar os melhores conceitos no desenvolvimento WEB.

A utilização dos *Beacons* para a identificação de presença de usuário próximo ao aparelho, resultou em uma aplicação capaz de auxiliar usuários na pratica de exercícios físicos e também abriu uma alta possibilidades de aplicações seguindo esse tipo de conceito.

A pesquisa resultou em diversas formas de comunicação entre aparelhos, ou seja, o conceito de Internet das Coisas. Portanto para trabalhos futuros, é possível disponibilizar também na ferramenta uma opção de relatórios, com o desempenho do usuário no equipamento levanto em consideração o tempo em que o usuário permaneceu na prática do exercício. Também pode ser utilizado o conceito de realidade aumentada para a exemplificação de como executar os movimentos nos equipamentos de maneira correta.

Outra possibilidade seria desenvolver uma aplicação semelhando utilizando o conceito de *Radio-Frequency Identification* (RFID), onde a identificação dos aparelhos é feita através dos RFID. Também pode-se utilizar este mesmo tipo de aplicação para manuais de carros ou até mesmo para se obter informações sobre uma peça de roupa desejada em determinada loja.

REFERÊNCIAS

ALMEIDA, Flávio. **Angular 2: o fim do two-way data binding?** 2016. Disponível em: <http://blog.caelum.com.br/angular-2-o-fim-do-two-way-data-binding/>. Acesso em: 26 out. 2017.

ALMEIDA, Hyggo. **Tudo conectado – Internet das Coisas**. Revista da Sociedade Brasileira de Computação, 29, 04/2015.

ASHTON, Kevin. That 'Internet of Things' thing. Publicano no RFID Journal, 2009. Disponível em <http://www.rfidjournal.com/article/view/4986> Acesso em 11 set. 2017.

BATISTA, Diogo T. C. **O impacto do HTML5 no desenvolvimento para a internet**. 2009. Disponível em: <www.diogocezar.com/files/html5/artigo_html5.pdf>. Acesso em: 15 set. 2014.

BRASIL. Ministério da Ciência e Tecnologia. Secretaria de Política de Informática e Automação. Evolução da Internet no Brasil e no Mundo. 2000. Disponível em: <http://www.dominiopublico.gov.br/pesquisa/DetalheObraForm.do?select_action=&co_obra=18037> Acesso em: 07 set 2017

CAELUM. **F11 - Java e Orientação a Objetos**. Disponível em: <https://www.caelum.com.br/download/caelum-java-objetos-fj11.pdf>. Acesso em: 03 out. 2017.

CAMPOS, Wander. **APLICAÇÕES MÓVEIS PARA EMPRESAS. SAIBA COMO AJUDAM A SEU NEGÓCIO.** Disponível em: <https://www.questionpro.com/blog/pt-br/como-ferramentas-e-aplicacoes-moveis-ajudam-as-empresas/>. Acesso em: 06 nov. 2017.

CANTARINO, Renato. **Injeção de dependência no AngularJS.** 2014. Disponível em: <https://trolldeveloper.wordpress.com/2014/08/15/angularjs-injecao-de-dependencia-no-angularjs/>. Acesso em: 26 out. 2017.

CARNEIRO, Conrado. **Beacon: o que é e quais suas utilizações mais inusitadas,** 2016. Disponível em: <http://usemobile.com.br/conheca-beacon/>. Acesso em 26 set. 2017.

CAROLINA, Ana. **Life Fitness aposta em gamificação, acessibilidade e tecnologia ao lançar seis novos produtos no Brasil.** 2017. Disponível em: <http://www.segs.com.br/info-ti/81677-life-fitness-aposta-em-gamificacao-

acessibilidade-e-tecnologia-ao-lancar-seis-novos-produtos-no-brasil.html>. Acesso em: 02 nov. 2017.

CHAMUSCA, A. **Domótica e Segurança Electrónica: a inteligência que se instala.** Portugal: Ingenium, 2006.

CHRISTENSEN, C. M. The ongoing process of building a theory of disruption. **The Journal of Product Innovation Management**, v. 23, n.1, 2006.

CONVERTTE. **Mobile Marketing: o futuro do marketing através dos aparelhos móveis.** 2014. Disponível em: <https://www.convertte.com.br/mobile-marketing/>. Acesso em: 06 nov. 2017.

CROCKFORD, Douglas. **JavaScript:** The Good Parts. 1. ed. Estados Unidos: Yahoo! Inc., 2008.

DAVID NIELD (Brasil). **Conheça todos os sensores do seu smartphone e como eles funcionam.** 2017. Disponível em: <https://gizmodo.uol.com.br/sensores-smartphones-guia/>. Acesso em: 04 jul. 2018.

DARIN, Ticianne et al. **GreatRoom: Uma Aplicação Android Baseada em Proximidade para a Criação de Salas Virtuais Inteligentes.** 2016. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wfa/2016/004.pdf>. Acesso em: 13 nov. 2017.

DIETZ, Frederik. **Recipes with Angular.js.** Disponível em: <https://leanpub.com/recipes-with-angular-js/read>. Acesso em: 17 de out. 2017.

DOCS ANGULAR. API Guide. Disponivel em:<https://docs.angularjs.org/guide/>. Acesso em: 26 de out. 2017.

FELIZARDO, Andre. **O que é AngularJS.** 2015. Disponível em: <http://www.andrefelizardo.com.br/blog/o-que-e-angularjs/>. Acesso em: 19 out. 2017.

FERNANDES, Maria Cecília. **O que é um aplicativo móvel?**. 2016. Disponível em: <http://blog.stone.com.br/aplicativo-movel/>. Acesso em: 06 nov. 2017.

FISHER, Thomas. **AngularJS Scopes: An Introduction.** 2014. Disponível em: <https://blog.carbonfive.com/2014/02/11/angularjs-scopes-an-introduction/>. Acesso em: 26 out. 2017.

GREEN, Brad; SESHADRI, Shyam. **AngularJS.** Sebastopol,: O'Reilly Media, 2013. 196 p.

INDRUSIAK, Leandro Soares. **Linguagem Java.** 1996. Disponível em: <http://docshare01.docshare.tips/files/9809/98094922.pdf>. Acesso em: 02 out. 2017.

IONIC. **Advanced HTML5 hybrid mobile app framework.** Disponível em: <http://ionicframework.com>. Acesso em: 11 out. 2017.

KAPP, Karl. The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education. Pfeiffer, 2012.

LIMA, Stefano Costa Barroso; SILVA, Agatha. **Introdução ao CSS.** 2017. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Aprender/CSS/Introduction_to_CSS>. Acesso em: 26 out. 2017.

MCGEE, James V.; PRUSAK, Laurence. **Gerenciamento estratégico da informação**. Elsevier Brasil, 2004.

MILANI, André. **MySQL-guia do programador.** São Paulo: Novatec Editora, 2007. 398p.

MILETTO, E. M.; BERTAGNOLLI, S. D. C. **Desenvolvimento de Software II: Introdução ao Desenvolvimento Web com HTML, CSS, JavaScript e PHP**. Porto Alegre: Bookman, 2014.

MYSQL. **MySQL Workbench.** 2017. Disponível em: <https://dev.mysql.com/doc/workbench/en/>. Acesso em: 11 out. 2017.

NOGUEIRA, Rebecca. **6 erros comuns na musculação.** 2015. Disponível em: <http://corpoacorpo.uol.com.br/fitness/treino-na-academia/6-erros-comuns-na-musculacao/8475#>. Acesso em: 02 nov. 2017.

OLSSON, Tommy; O'BRIEN, Paul. **The ultimate CSS reference**. 1. ed. Estados Unidos: SitePoint Pty Ltd, 2008.

OPSERVICES (Porto Alegre). **Exemplos e aplicações de Internet das Coisas (IoT).** 2016. Disponível em: <https://www.opservices.com.br/exemplos-de-internet-das-coisas/>. Acesso em: 04 jul. 2018.

PAIVA, Nuno Alexandre Lemos de. **Desenvolvimento de uma APP Android e Plataforma Web para comunicação com Beacons.** 2016. Disponível em: <http://hdl.handle.net/10400.26/16691>. Acesso em: 13 nov. 2017.

PERRY, Steve. **Fundamentos da linguagem Java:** Programação orientada a objetos na plataforma Java. 2016. Disponível em: <https://www.ibm.com/developerworks/br/java/tutorials/j-introtojava1/index.html>. Acesso em: 09 out. 2017.

POWEL, Thomas.**HTML & CSS:** The Complete Reference, Fifth Edition. 5. ed.Estados Unidos: Mc Graw Hill, 2010.

RAFT. **O que é disrupção?** 2015. Disponível em: <https://projetodraft.com/verbete-draft-o-que-e-disrupcao/>. Acesso em: 04 jul. 2018.

REZENDE, Denis Alcides. Evolução da Tecnologia da Informação nos Últimos 45 anos. **Revista FAE Business**, n. 4, p. 42-46, 2002.

ROCHA, Carlos E. et al. **Framework para Aplicações em Plataformas Móveis Usando Panoramas com Camadas**. Rio de Janeiro, p. 1–77, 2014. Disponível em: <https://www.visgraf.impa.br/Data/RefBib/PS_PDF/tr-042014/tr-04-2014.pdf>. Acesso em 14 set. 2017.

ROCHA, Diego Tondim. **TÔ AQUI: APLICATIVO PARA GEORREFERENCIAMENTO EM AMBIENTES RESTRITOS.** 2015. Disponível em: <http://dsc.inf.furb.br/arquivos/tccs/monografias/2015_1_diego-tondim-rocha_monografia.pdf>. Acesso em: 13 nov. 2017.

RUEBBELKE, Lukas. **Awesome AngularJS Features.** Disponível em: <http://net.tutsplus.com/tutorials/javascript-ajax/5-awesome-angularjs-features/>. Acesso em: 03 jul. 2018.

SCHMITZ, Daniel; LIRA, Douglas. **AngularJs na prática.** 2014. Disponível em: <https://leanpub.com/livro-angularJS/read>. Acesso em: 19 out. 2017.

SILVANO, Claudia. **A importância de ler o manual de instruções.** 2017. Disponível em: <http://www.tribunapr.com.br/blogs/tribuna-do-consumidor/manual-de-instrucoes/>. Acesso em: 02 nov. 2017.

SIMOVA. **ENTENDA AS VANTAGENS DA TECNOLOGIA MOBILE PARA EMPRESAS.** Disponível em: <http://blog.simova.com.br/entenda-as-vantagens-da-tecnologia-mobile-para-empresas/>. Acesso em: 06 nov. 2017.

SINGER, Talyta. Tudo conectado: conceitos e representações da internet das coisas. **Simpósio em tecnologias digitais e sociabilidade**. Disponível em: <http://www.simsocial2012.ufba.br/modulos/submissao/Upload/44965.pdf> Acesso em: 11 set. 2017.

STECZKIEWICZ, Agnieszka. **How to modify UUID, Major, and Minor values?** Disponível em: <https://community.estimote.com/hc/en-us/articles/200868188-How-tomodify-UUID-Major-and-Minor-values->. Acessado em: 14 set. 2017.

SUPPORT APPLE. Disponível em: <https://support.apple.com/en-us/HT202880>. Acesso em: 23 de agosto. 2017.

TEIXEIRA, Fabrício. **Tudo o que você precisa saber para começar a brincar com iBeacons**, 2014. Disponível em: <https://brasil.uxdesign.cc/tudo-o-que-voc%C3%AA-precisa-saber-para-come%C3%A7ar-a-brincar-com-ibeacons-fdf5847e640b>. Acesso em 14 set. 2017.

TOTALCROSS. **Entenda a importância do conteúdo nos aplicativos móveis.** Disponível em: <http://www.totalcross.com/blog/entenda-a-importancia-do-conteudo-nos-aplicativos-moveis/>. Acesso em: 06 nov. 2017.

UNIPÊ (João Pessoa). **Tecnologia da Informação: tudo que você precisa saber.** 2018. Disponível em: <http://blog.unipe.br/graduacao/tecnologia-da-informacao-tudo-que-voce-precisa-saber>. Acesso em: 04 jul. 2018.

VENKATESH, A. Computers and other interactive technologies for the home. **Communications of the ACM**, v. 39, n. 12, 1996.

WERBACH, Kevin; HUNTER, Dan. For The Win: How Game Thinking Can Revolutionize Your Business. Filadélfia, Pensilvânia: Wharton Digital Press, 2012.

ZAKAS, Nicholas C. **Professional JavaScript for Web Developers**. 3. ed. Estados Unidos: Wrox, 2012.

ZAMBARDA, Pedro. **Internet das Coisas: entenda o conceito e o que muda com a tecnologia.** 2014. Disponível em: <http://www.techtudo.com.br/noticias/noticia/2014/08/internet-das-coisas-entenda-o-conceito-e-o-que-muda-com-tecnologia.html>. Acesso em: 04 jul. 2018.

APÊNDICE(S)

APÊNDICE A – Artigo do Trabalho de Conclusão de Curso

Utilização do conceito de Internet das Coisas associada à tecnologia de iBeacons como ferramenta computacional aplicada à instrução de usuários na utilização de aparelhos de ginástica

Maikeu P. Duarte[1], Gustavo Bisognin[2]

[1] Universidade do Extremo Sul Catarinense (UNESC) Caixa Postal 3167 – 88806-00- Criciúma – SC - Brasil

[2] MSc. Professor do Curso de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) Caixa Postal 3167 – 88806-00 - Criciúma – SC - Brasil

maikeuduarte@gmail.com, gbisog@gmail.com

Abstract. *With the constant search for health, gym environments are gaining space for physical exercise, with the great demand and the limited number of professionals trained to instruct users in the use of the devices, some users choose to avoid the orientations and end up executing incorrect movements in the devices being possible to occur injuries. Due to the popularity of gadgets it is possible to create an application that works as a sort of digital manual where it assists the user. The development of the prototype was performed with the REST architecture for the server side and with the AngularJS framework for the development of the client side, having as the two main programming languages JAVA and JavaScript.*

Resumo. *Com a constante procura por manter a saúde em dia, os ambientes de academia vêm ganhando espaço para a pratica de exercício físico, com a grande demanda e a quantidade limitada de profissionais capacitados para instruir os usuários na utilização dos aparelhos, alguns usuários optam por evitar as orientações e acabam executando movimentos incorretos nos aparelhos sendo possível ocorrer lesões. Devido a popularidade dos gadgets é possível criar uma aplicação que funciona como uma espécie de manual digital, onde auxilia o usuário. O desenvolvimento do protótipo foi realizado com a arquitetura REST para o lado server e com o framework AngularJS para o desenvolvimento do lado client, tendo como as duas principais linguagens de programação o JAVA e o JavaScript.*

1. INTRODUÇÃO

A crescente evolução da tecnologia computacional aplicada ao conceito de internet das coisas vem apresentando diversas soluções que facilitam a vida dos usuários nos mais diversos níveis. Desta forma, algumas aplicações práticas destacam-se no mercado em geral, dentre elas, podemos citar as tecnologias interativas que criam uma conexão do usuário com as coisas cotidianas, como refrigeradores interativos, aparelhos de TV, pisos inteligentes e a domótica em geral

Um dos objetos de estudo deste trabalho, aborda a Internet das Coisas, por ser uma tecnologia que vem ganhando espaço no mercado, com possibilidades infinitas de negócios. Como está tecnologia é uma tecnologia recente, ela possui um amplo mercado, empresas como Google, Microsoft entre outras estão trabalhando firme nesse conceito, com o intuito de revolucionar a tecnologia e o nosso dia-a-dia. Outra constatação importante, é a associação do conceito de *Internet of Things* (*IoT)*

com os dispositivos beacons, através deles é possível recolher dados e informações que nos permitiriam criar aplicações que nos possibilitariam interagir com os objetos do nosso dia-a-dia.

Com a crescente procura por soluções referentes à qualidade de vida destaca as academias como sendo um dos principais recursos utilizados para a prática de exercícios associados à saúde e bem-estar. A maioria dos usuários iniciantes tem dificuldade em associar o nome ao exercício, e qual o equipamento usar para a prática do mesmo, nesses casos, torna-se necessário à ajuda de um personal trainer ou profissional capacitado, para que o mesmo lhe mostre qual o exercício e o equipamento que pode ser utilizado para a pratica.

O presente trabalho pretende abordar o estudo da internet das coisas com a junção de algumas tecnologias como AngularJs, Ionic dentre outras na implementação de um protótipo que visa à resolução do problema na falta de conhecimento para determinadas práticas de exercícios nos ambientes de academia e também a falta de personal trainers disponíveis. Além disso, visa o estudo da tecnologia Beacon utilizando um aplicativo móvel que detecta a região que o usuário está contido e a localização do Beacon, e fornece informações ao usuário através do web service, como quais exercícios podem ser executados em determinados aparelhos e um breve vídeo de como executar o movimento.

2. DESENVOLVIMENTO DO PROTÓTIPO

O trabalho tem como objetivo proporcionar a criação de uma ferramenta computacional, para o auxílio de praticantes de exercícios físicos no ambiente de academia, onde será possível acompanhar a sua ficha de treino, além de exibir tutorias de como executar o movimento no aparelho, que será exibido assim que o usuário se aproximar do mesmo.

2.1. INICIALIZAÇÃO

Com a popularidade dos dispositivos móveis, foi realizado um levantamento de requisitos através de pesquisas bibliográficas para disponibilizar uma ferramenta capaz de auxiliar indivíduos na prática de exercícios físicos no ambiente de academias, por meio de um manual digital gamificado ou através de vídeos explicativos.

Com isso foi levantado alguns requisitos funcionais para dar início na elaboração do protótipo. Assim foi levantado todas a telas da aplicação e o fluxo de cadastro e além do fluxo entre as telas.

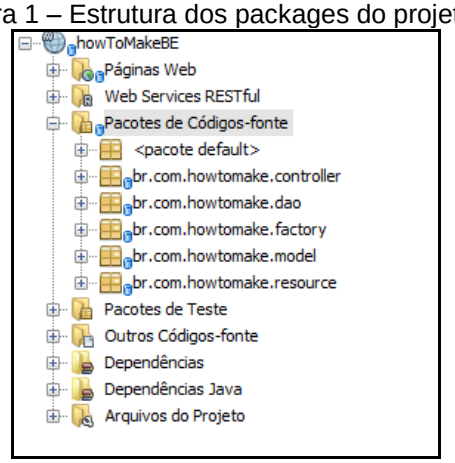
Com o levantamento dos requisitos, se tornou importante a criação da modelagem de dados, para auxiliar em uma visão geral do contexto para construir a ferramenta. Após está etapa, ficou explícito a necessidade de escolher as tecnologias a serem utilizadas para a criação da aplicação.

2.2. PROJETO BACK-END

Para o desenvolvimento do projeto foi utilizado a IDE Netabens, que é disponibilizada gratuitamente no site do fabricante. A IDE foi escolhida devido a possuir integração com diversos servidores, como Tomcat e Glassfish e facilitar na criação de projetos em diversas linguagens, além de ser a ferramenta utilizada em aulas durante o curso, tendo assim um domínio maior sobre a mesma.

Após toda a parte de configuração do projeto, foi dado início no desenvolvimento. O primeiro passo foi de estruturar o projeto, criando os packages, ficando conforme a figura 1.

Figura 1 – Estrutura dos packages do projeto.



Fonte: Do autor.

O projeto foi dividido em quatro *packages,* neles estão contidas todas as classes JAVA do projeto. O *package br.com.howtomake.controller* é responsável por amarzenar todos os *controllers*, um controle tem literalmente a função de controlar a aplicação. *Controller* tem a função de controlar a comunicação dos *resources* com a camada DAO que veremos a seguir. Neles se encontram todos os métodos que serão consumidos pelos *resources*.

No *package br.com.howtomake.dao* estão contidas todas as classes DAO, elas são encarregadas por todo o CRUD da aplicação, efetuando consultas ou inserindo registros no banco de dados. Todas as classes DAO extendem a classe *ConnectionFactory* que é responsável por toda a configuração de banco de dados, como criar uma conexão com o banco e fechar, veremos sobre ela a frente. Para a criação dos SQL é utilizado o *prepareStatement* que seria como um ajudante na criação dos scripts. Após a criação do SQL, é necessário executar ele através do *executeQuery*. Todos estes métodos pertencem a API do *Java Database Connectivity* (JDBC).

O *package br.com.howtomake.factory* é designado por alocar todos os serviços em comum da aplicação. Nele temos a classe *ConnectionFactory* que é responsável por criar a conexão com o banco de dados *MySql* e também encerrar a mesma.

O *package br.com.howtomake.model* contem todos os modelos, entidades da aplicação, todos os objetos representacionais com que a aplicação ira controlar e gerenciar dados, cada modelo representa uma tabela do banco, quais aributos ela tem e quais os tipos de dados suportados por ela. Todas as entidades possuem seus atributos e tipos de dados, além de possuir dois contrutores, um padrão e outro recebendo todas as informações para instanciar um novo objeto. Além disso possuem os metodos *getters* e *setters* que são necessarios para buscar e setar as informações das propriedades das entidades respectivamente.

O ultimo *package* é o *br.com.howtomake.resource* nele contem todos os serviços da aplicação, estes serviços serviram como uma espécie de ponte para a comunicação entre o projeto BE e FE. Para criar os serviços foi utilizado a bliblioteca *Jersey*, que é uma implementação do padrão REST.

Em um primeiro momento foi necessário configurar o *Jersey*, no arquivo *web.xml* da aplicação, para que o *Servelet* do *Jersey* consiga gerenciar os serviços criados (figura 2).

Figura 2 – Configuração do Jersey.

```
<servlet>
        <servlet-name>Jersey REST Service</servlet-name>
        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
        <init-param>
            <param-name>com.sun.jersey.config.property.packages</param-name>
            <param-value>br.com.howtomake.resource</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
        <servlet-name>Jersey REST Service</servlet-name>
        <url-pattern>/*</url-pattern>
</servlet-mapping>
```

Fonte: Do autor.

No parametro *com.sun.jersey.config.property.packages* é especificado onde estarão os serviços e na configuração do *Servelet* foi definida a URL de como acessar estes serviços, nesse caso através do "/*", sendo assim todos os diretórios.

Depois de configurado, foi criado cada serviço no pacote definido na configuração. Todos os serviços possuem um caminho, que é identificado através da notação *@path,* que será utilizada na hora de consumir esse serviço na aplicação FE. Também contém o tipo de requisição que é feito para acessar, podendo ser as seguintes notações *@POST, @GET, @PUT, @DELETE* entre outros. Além disso também é especificado o tipo de mídia que o *endpoint* irá consumir e produzir, sendo respectivamente configurados pelas notações *@Consumes* e @Produces (figura 3).

Figura 3 – Exemplo de classe resource.

```
@Path("/aparelhos")
public class AparelhosResource {
        @GET
        @Path("/listarTodos")
        @Produces(MediaType.APPLICATION_JSON)
        public ArrayList<Aparelhos> listarTodos() {...3 linhas }

        @GET
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/id/{id}")
        public Response getById(@PathParam("id") Long id) {...10 linhas }

        @POST
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/salvar")
        public Response salvarAparelhoJson(Aparelhos aparelho) {...11 linhas }

        @PUT
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/atualizar")
        public Response altualizarAparelho(Aparelhos aparelho) {...11 linhas }

        @POST
        @Consumes(MediaType.APPLICATION_JSON)
        @Produces(MediaType.APPLICATION_JSON)
        @Path("/deletar")
        public Response deletarAparelho(Aparelhos aparelho) {...15 linhas }
}
```

Fonte: Do autor.

Na figura 3, existem cinco *endpoints* que podem ser consumidos futuramente pela aplicação FE. O serviço acima será usado para salvar os dados dos aparelhos, para salvar um registo será enviado um *POST* para a URL "*/aparelhos/salvar"*, está

URL estará esperando por uma mídia do tipo *JSON* e nos retornará um *JSON* com a resposta.

2.3. PROJETO FRONT-END

Após finalização do projeto BE, assim liberando os recursos necessários pela aplicação FE, pode se dar início no desenvolvimento da próxima etapa. Para o desenvolvimento do projeto foi utilizado a IDE Visual Code, que é disponibilizada gratuitamente no site do fabricante.

Para o desenvolvimento FE foi decidido utilizar *IONIC*, que é uma ferramenta *Cross Platform,* onde pode ser utilizado linguagens WEB conhecidas, desenvolvendo toda a aplicação com base nos conhecimentos WEB. *IONIC* tem suporte ao cordova, que nos permite acessar recursos nativos do dispositivo, como Câmera, *Bluetooth* entre outros.

Para o denseolvimento das telas foi utilizado o *framework AngularJS* e HTML, tornando o desenvolvimento ágil e rápido, ganhando em produtividade. Para desenvolver uma aplicação em *AngularJS* é necessário identificar que a mesma será uma aplicação *AngularJS* através do *ng-app="howToMake",* onde o *howToMake* é o nome da aplicação, e também o nosso módulo principal. Estão disponíveis dois menus, uma versão para administradores e outra para clientes, onde cada item do menu possui dois subitens, Ínicio onde é exibida a listagem e Novo por onde é feita a entrada de novos registros para a aplicação.

Um dos itens mais importantes da aplicação é o cadastro de aparelhos, conforme a imagem 4 demostra a listagem de aparelhos e também a parte de novos aparelhos, onde é possível observar os dados de configurações dos Beacons. identificador, que é um código unico por beacon, Id Beacon (Region) que é um ID de região, que especifica um tipo de rede de beacons, onde você pode fazer vários beacons trabalhar em uma mesma região ou em regiões diferentes, e por último os atributos Major e Minor, que são únicos de ibeacon para ibeacon, que são uados para diferenciar os ibeacons, por meio de localização ou conforme desejado (figura 4).

Figura 4 – Telas de listagem e cadastro de Aparelhos.

Fonte: Do autor.

Todas as telas, possuem um controller, um service e seus respectivos templates. A figura 5 demonstra um controller, o de listagem de dados da tela de aparelhos, que é semelhante em todas as demais telas.

Figura 5 – *Controller* da tela de listagem de aparelhos.

```
angular.module('starter.controllers.aparelhos', [])

    .controller('AparelhosCtrl', ['aparelhoService', 'loaderService', '$state', '$ionicPopup',
    function (aparelhoService, loaderService, $state, $ionicPopup) {
        var that = this;
        that.init = function () {
            that.listaAparelhos = [];

            loaderService.show();
            aparelhoService.getAll().then(function (result) {
                if(result.data){
                    if(angular.isArray(result.data.aparelhos)){
                        that.listaAparelhos = result.data.aparelhos;
                    }else{
                        that.listaAparelhos.push(result.data.aparelhos);
                    }
                }
            }).finally(function () {
                loaderService.hide();
            });
        };
        that.init();

        that.editar = function (id) {···
        };

        that.remover = function (aparelho) {···
        };
    }]);
```

Fonte: Do autor.

O *AngularJS* tem uma ferramenta muito forte que é a parte de injeção de dependências, onde você pode dividir o seu código em modulos, e depois injetalos e reusalos em toda a aplicação. Nesse *controller* estamos injetanto *aparelhoService*, que é responsavel por todas as nossas chamadas *REST(POST, GET, DELETE, PUT)*, em seguida o nosso serviço de *loader*, que é responsavel por colocar o *loading* nas páginas, *$state* responsável de controlar as nossas rotas, conforme vimos anterioremente, *$ionicPopup* um componente que o IONIC nos fornece, que tem algumas funcionalidades interessantes, no nosso contexto foi usado para a confirmação ao excluir registros.

A figura 6 exemplifica um *service*, o serviço abaixo é o responsável pela tela de aparelhos, neles estão todos os métodos que são responsáveis por salvar um registros, atualizar, deletar, listar todos e também buscar um registro específico. Para executar as chamadas dos métodos foi utilizado o serviço *$http,* que é um serviço fornecido pelo *AngularJS.*

Figura 6 – Service Aparelhos.

```
angular.module('starter.services.aparelhos', [])
    .service('aparelhoService', aparelhoService);

aparelhoService.$inject = ['$http', '$q'];

function aparelhoService($http, $q) {
    var baseService = 'http://192.168.0.102:8084/howToMakeBE/aparelhos';
    this.save = function (aparelho) {
        return $http.post(baseService + '/salvar', aparelho);
    }

    this.put = function (aparelho) {
        return $http.put(baseService + '/atualizar', aparelho);
    }

    this.delete = function (aparelho) {
        return $http.post(baseService+'/deletar', aparelho);
    }

    this.getAll = function () {
        return $http.get(baseService + '/listarTodos');
    }

    this.get = function (id) {
        return $http.get(baseService + '/id/' + id);
    }
}
```

Fonte: Do autor.

As tags *<ion-view></ion-view>* demostram o conteúdo que será exibido entre o menu, *<ion-nav-buttons></ion-nav-buttons>* demarcam os ícones que apareceram no menu, como voltar para a tela anterior entre outros. A tag *<ion-content></ion-content>* exibem o conteúdo principal da tela, como campos de entrada ou listagem de dados. Com a utilização das diretivas do *AngularJS* tornou-se mais fácil a criação de rotinas como listagem de dados, editar e excluir registros. O atributo *ng-repeat* nos da a possibilidade de criar uma lista de registros com pouco trecho de código, onde ele recebe um *array* e percorre todos os itens tornando o trabalho mais fácil e ágil. O atributo *ng-click* é usado para executar eventos quando o usuário clica em algum elemento com o atributo, na imagem acima ele foi utilizado para a edição de registros e exclusão.

Após a construção da aplicação *front-end*, foi dado inicio na utilização dos beacons na aplicação, o primeiro passo foi efetuar a configuração dos beacons. Os dispositivos utilizados são da estimote, uma distribuidora desse tipo de equipamento. A configuração dos beacons foi feita atraves da ferramente disponibilzada pela estimote tanto para iOS como Android. O próximo passo foi configurar a aplicação para dar suporte aos *Beacons*, para tal comportamento foi adicionado o plugin do *Cordova* para *Beacons.*

Para que a aplicação seja capaz de detequitar a presença dos dispositivos móveis em determinadas localizações, é necessário que você inicie o monitoramento dos *beacons* na aplicação, no contexto da aplicação cada *beacon* representa um equipamento da academia. Após o usuário logar na aplicação, é dado ínicio na busca dos treinos disponíveis do usuário e criado um *array* com os itens, após finalizar esse processo em seguida é chamado o método “iniciarBeacons” que é responsável por criar as regiões de monitoramento. O método é responsável por percorrer todos os itens da lista e também é responsável por criar um novo objeto que vincula o identificar do *Beacon* com o exercício.

Com a criação das regiões, é necessário identificar quando um dispositivo móvel entra na região e em seguida exibir o alerta com a informação de que existe um vídeo de instruções de uso do equipamento (figura 7).

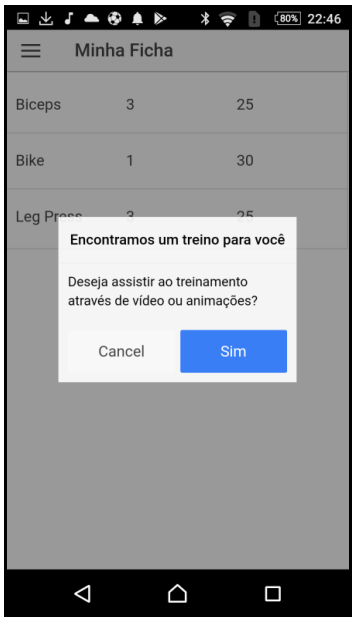
Figura 7 – Identifica quando um Beacon entra na região.

```
$rootScope.$on("$cordovaBeacon:didEnterRegion", function (event, pluginResult) {
    if (pluginResult.region.identifier != atualBeacon) {
        encontrouTreino(pluginResult.region.identifier);
    }

    atualBeacon = pluginResult.region.identifier;
    $scope.$apply();
});
```

Fonte: Do autor.

O código demostrado na figura 7 é chamado quando a aplicação identifica que um dispositivo móvel entrou em uma determinada região, o método é incumbido de identificar se é uma nova região e então chamar o método responsável por exibir o alerta de identificação de treinos para o equipamento (figura 8).

Figura 8 – Alerta de identificação de treino para equipamento.



Fonte: Do autor.

Se o usuário confirmar que deseja visualizar os vídeos e animações disponíveis para o equipamento, em seguida a aplicação abre a tela demostrando as informações do equipamento além de um vídeo autoexplicativo.

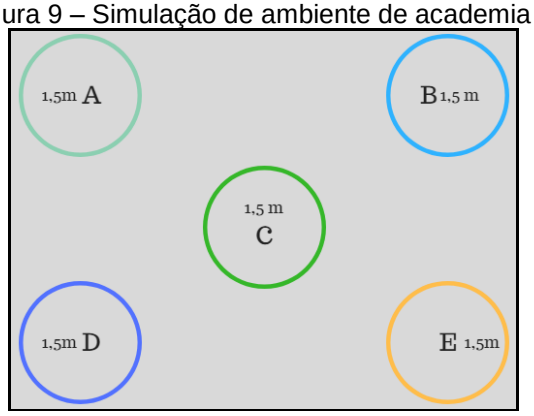
Após a finalização das etapas anteriores, a próxima etapa foi fazer todas as ferramentas trabalharem em conjunto, desde das aplicações *back-end* e *front-end* e a interação com os beacons. Foram simulados alguns ambientes de teste para comparar e verificar o desempenho da ferramenta. Primeiro foi feito um teste se a aplicação estava identificando os beacons, para isso é necessário que o dispositivo permita que o protótipo use a sua localização, então ao abrir o protótipo o dispositivo solicita a permissão para utilizar esse recurso.

Com o sucesso na primeira etapa, foi configurado os beacons para emitirem o sinal em um raio de 7 metros para se realizar os primeiros testes, os beacons foram dispostos em locais estratégicos nos equipamentos, o primeiro teste foi com aparelhos espalhados pelo ambiente. O primeiro caso de teste, os resultados não foram satisfatórios, cada letra representa um beacon e um aparelho de ginastica, na imagem podemos ver que o sinal dos beacons ficou sobrepondo um ao outro, dando uma interferência de sinal, onde a aplicação não se comportou bem e exibia alertas simultaneamente.

Foi identificado que o problema se tratava do raio de emissão de sinal dos *beacons*, então foram feitos testes reduzindo a área de transmissão dos *beacons* para 3,5m e 1,5m respectivamente, os resultados foram melhorando gradativamente.

Para o caso de teste com os beacons com transmissão de sinal para 3,5 metros já houve uma melhora significante, mas em determinados momentos ainda havia a interferência de sinal dos beacons, com uma frequência bem inferior ao primeiro caso. Com o terceiro caso os beacons conseguiram ter um aproveitamento de 99,99%, os alertas só eram demonstrados para determinado equipamento que possuía os beacons, mas foi possível observar que poderia ser melhorado a disposição dos equipamentos nas salas, assim melhorando o sinal dos beacons e o bom funcionamento da aplicação. Foi considerado como um bom cenário para a aplicação os aparelhos serem distribuídos conforme a figura 9.

Figura 9 – Simulação de ambiente de academia 2.



Fonte: Do autor.

A aplicação funcionou conforme o esperado, mostrando os alertas dos tutoriais somente quando o usuário adentrava na área de transmissão do *beacon*, sem houver interferência de sinal. Lembrando que a aplicação funciona também coma disposição dos equipamentos de outras formas, mas talvez possa ocorrer os problemas de interferência de sinal.

Os beacons são configurados para emitir sinal de 1,5m, o mínimo possível através das configurações da estimote, dessa forma é interessante manter a distância de no mínimo 1,5m de um equipamento para o outro, para que o protótipo funcione como desejado.

3. RESULTADOS OBTIDOS

Com o desenvolvimento do projeto de pesquisa deste trabalho, pôde-se obter uma aplicação de estudo para realizar a integração dos *beacons* com o conceito de Internet da Coisas, utilizando o *framework* AngularJs com o padrão MVC e Jersey para a criação dos serviços. Com isso foi possível disponibilizar uma ferramenta que funciona como um manual digital para ambientes de academia.

O desenvolvimento do protótipo foi realizado com o *framework* AngularJS, utilizando o conceito de MVC que é uma pratica recomendada para estrutura do projeto, este *framework* colaborou em grande escala para a produção da aplicação do lado *front-end*, além de diminuir o tráfico de dados da aplicação devido utilizar o conceito de *Sigle-Page-Application*, as demais tecnologias usadas foram para facilitar o desenvolvimento.

Para disponibilizar a ferramenta para os dispositivos móveis foi utilizado o IONIC, que é uma ferramenta *Cross Platform*, onde é possível programar utilizando as linguagens de programação web e no final fazer o *build* para as plataformas desejadas, como Android, iOS ou Windows.

A parte do desenvolvimento *back-end* foi desenvolvida utilizando o conceito de REST e serviços, onde é disponibilizado *endpoints* que podem ser consumidos por diversas aplicações, tornando as maleáveis e totalmente desacopladas, assim um serviço recebe os dados através de *endpoints* e os retorna quando necessários. Os resultados obtidos pela aplicação *back-end* foram simulados em conexões 3G e rede *WIFI,* todos os resultados foram satisfatórios, tendo uma resposta quase imediata dos *endpoints*.

Para se obter a localização do usuário no ambiente, foram utilizados *beacons,* os primeiros resultados foram ruins, havendo interferência de sinal, devido ao raio de rastreio dos dispositivos *beacons.* Os resultados posteriores foram positivos, tendo o

resultado esperado pela aplicação, identificando a presença do dispositivo somente quando o usuário adentrava na região do *Beacon*.

Na integração dos projetos foi possível notar as vantagens em utilizar o conceito de se ter dois projetos distintos, um sendo responsável pela parte visual e de entrada de dados e outro responsável por armazená-las e consumi-las da melhor maneira possível, além de ser possível consumir estes dados em outros meios, como páginas web ou em outros protótipos.

Com a junção de todos as etapas citadas anteriormente, foi possível disponibilizar um protótipo final que possibilitou aos usuários de academia, que pratiquem suas atividades físicas nos ambientes de academia sem a ajuda de um *Personal Trainer,* utilizando apenas o protótipo como um manual autoexplicativo. O protótipo disponibilizado possui uma parte administrativa para os *Personal Trainers* e a outra parte como um manual digital.

4. CONCLUSÃO

Devido à grande procura por manter a saúde em dia e a popularização da pratica de exercícios nas academias, além da alta demanda de dispositivos móveis, o projeto de pesquisa resultou-se em disponibilizar uma ferramenta para o auxílio na prática de exercícios físicos.

Com o desenvolvimento do projeto se obteve conhecimento em diversas áreas importantes, além de se obter o conhecimento em conceitos populares utilizados na construção do protótipo, assim se obteve maior facilidade no desenvolvimento dos projetos, em ambos os projetos foram utilizados o conceito de MVC.

Como resultado final para o usuário, se teve um ganho elevado em performance devido ao AngularJs utilizar o conceito de *Single-Page-Application*, tornando a experiencia do usuário agradável e convidativa. Além do *framework* ser recomendado por desenvolvedores e possuir grande quantidade de conteúdo na internet e o mesmo utilizar os melhores conceitos no desenvolvimento WEB.

A utilização dos *Beacons* para a identificação de presença de usuário próximo ao aparelho, resultou em uma aplicação capaz de auxiliar usuários na pratica de exercícios físicos e também abriu uma alta possibilidades de aplicações seguindo esse tipo de conceito.

A pesquisa resultou em diversas formas de comunicação entre aparelhos, ou seja, o conceito de Internet das Coisas. Portanto para trabalhos futuros, é possível disponibilizar também na ferramenta uma opção de relatórios, com o desempenho do usuário no equipamento levanto em consideração o tempo em que o usuário permaneceu na prática do exercício. Também pode ser utilizado o conceito de realidade aumentada para a exemplificação de como executar os movimentos nos equipamentos de maneira correta.

Outra possibilidade seria desenvolver uma aplicação semelhando utilizando o conceito de *Radio-Frequency Identification* (RFID), onde a identificação dos aparelhos é feita através dos RFID. Também pode-se utilizar este mesmo tipo de aplicação para manuais de carros ou até mesmo para se obter informações sobre uma peça de roupa desejada em determinada loja.