Vol. 9 Nro. 2 2018

Smart Room: Criação de um quarto inteligente com preço acessível

Matheus Henrique Lopes BADOCO[1]

Mahezer Carvalho de MELO[2]

Leandro BORGES[3]

Resumo: Este artigo relata o desenvolvimento de um quarto inteligente de baixo custo, utilizando sistemas embarcados de *IoT* (Internet das coisas). O foco deste quarto inteligente é de controlar de maneira automática, sem qualquer intervenção humana, a temperatura e a umidade do quarto. Para isso, utilizamos sensores de baixo custo e boa performance para que eles verifiquem a temperatura, luminosidade e umidade atual do quarto, e acoplamos LEDs para indicar e simular o momento correto para que o ventilador e o umidificador liguem, de acordo com a necessidade do usuário, sem que ele se preocupe com isso. Este artigo também tem por objetivo servir de exemplo para futuros projetos automatizados de *IoT*, a fim de viabilizar este tipo de projeto, e torná-los mais próximos da sociedade.

Palavras-chave: Quarto Inteligente; *IoT (Internet of Things);* Sistemas embarcados; Baixo Custo; Arduino e NodeMCU.

Abstract: This article relates about the development of a Smart Room with low cost, by utilizing embedded systems of IoT. This smart room's focus is to control, automatically, without any human intervention, the temperature and humidity of the room. In order to do that, we used low cost sensors with good performance to read the actual temperature, luminosity and humidity of the room, and joined them together with some LEDs to indicate and simulate the right moment for the fan and the humidifier to turn on, based on the user needs, without making the user worry about it. This article also has the objective of being an example to future automatized projects of IoT, with the focus of viabilizing this kind of project, and make them closer to the society.

Keywords: Smart Room; Internet of Things; Embedded systems; Low Cost; Arduino and NodeMCU.

---

[1]Discente do curso de Bacharelado em Sistemas de Informação do Uni-FACEF
[2]Discente do curso de Bacharelado em Sistemas de Informação do Uni-FACEF
[3]Docente do curso de Bacharelado em Sistemas de Informação do Uni-FACEF

1. Introdução

Esse artigo tem como objetivo detalhar o desenvolvimento de um projeto de automatização domiciliar, mais especificamente de um quarto, tornando-o inteligente e independente, sem necessidade de qualquer intervenção humana.

1.1. Problema

Com a falta de autonomia dos itens domiciliares, as pessoas precisam fazer atividades repetitivas como ligar o ventilador quando está calor, ou apagar as luzes quando vão dormir. Tecnologias que utilizam Arduino para resolver este tipo de problema já existem, mas não são amplamente conhecidas pela população, e o custo destes equipamentos estão fora do orçamento de grande parte da população.

1.2. Justificativa

Este tema foi escolhido após reconhecermos a grande gama de problemas que poderiam ser solucionados ao utilizar o Arduino, e questionar a necessidade de precisar controlar manualmente cada componente da casa, enquanto os componentes poderiam ter alguma autonomia para agir. Além disso, o Arduino é uma tecnologia que abre um novo campo para pesquisa e aprendizado. Estes motivos aliados ao interesse dos pesquisadores, juntamente com a relação das tecnologias com tudo o que foi aprendido durante o curso, tornam este projeto relevante.

1.3. Motivação

Ao notar a distância entre este tipo de tecnologia e a grande maioria da população, percebemos a oportunidade de resolver problemas que a maioria das pessoas não imaginam que existam solução, além de podermos explorar melhor as possibilidades do Arduino em um ambiente prático, e servir de exemplo para futuros projetos ainda mais aprimorados.

1.4. Procedimentos Metodológicos

Para concluir este projeto iremos primeiramente adquirir e estudar os sensores necessários para controlar os componentes escolhidos no quarto. Para isso, vamos utilizar a documentação de cada um dos sensores, juntamente com a documentação da placa NodeMCU, para entender como será integrado cada sensor à placa. Com todos os sensores integrados, utilizaremos o protocolo *MQTT* para comunicar os estados de cada sensor para um servidor web, que irá decidir o que fazer para cada situação do sensor.

Embora o objetivo deste projeto seja fazer um quarto que possua autonomia para controlar seus componentes, em alguns momentos podem ocorrer exceções não

previstas no sistema. Para corrigir estes problemas, iremos criar um aplicativo de celular que envie um comando de ligar e desligar para a placa. Para isso, utilizaremos tecnologias mobile que irão se comunicar com o servidor web que recebe os dados dos sensores, e controlar os dispositivos da casa.

Ao definir esta estrutura, fica implícito que o ponto central de comunicação entre sensores, dispositivos e o aplicativo mobile será o servidor que irá orquestrar a autonomia dos dispositivos, assim como as exceções que o usuário poderá definir através do celular.

2. Referencial Teórico

Nosso projeto consiste na unificação de alguns sensores com a placa NodeMCU, tendo como maior diferencial tornar o funcionamento dessa placa autônoma, a fim de facilitar a vida do usuário. Para melhor compreensão do projeto como um todo, iremos explicar as tecnologias e os recursos que serão utilizados em nosso projeto.

2.1. Arduino

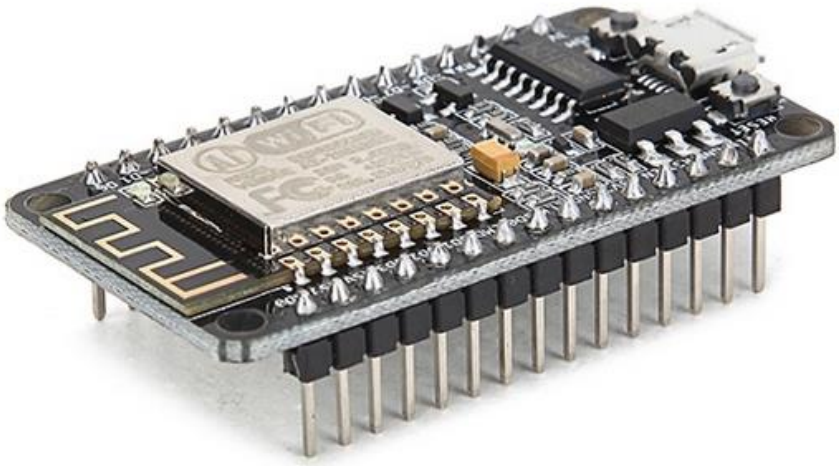
De acordo com Thomsen (2014), a criação do Arduino teve como objetivo “Elaborar um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de programar, sendo dessa forma acessível a estudantes e projetistas amadores”. Além disso, o Arduino possui um conceito de “*Hardware* Livre”, o que permite que qualquer um possa criar, montar ou melhorar, partindo desse mesmo hardware. Abaixo podemos verificar alguns tipos de Arduinos existentes na Figura 1, a seguir:

Figura 1 - Tipos de Arduino.

Fonte: *Print Screen* do site oficial do Arduino (2018)

Por ser uma plataforma, o Arduino é composto por vários componentes: As placas Arduino, a linguagem de programação Arduino, e a IDE Arduino. Para este projeto, estamos utilizando o *NodeMCU.* Essa placa consegue interpretar código feito para placas Arduino, porém, possui funcionalidades adicionais. A Figura 2 logo abaixo demonstra como é a placa NodeMCU:

Figura 2 - Placa NodeMCU.



Fonte: FilipeFlop (2018)

Essa plataforma será responsável por coletar os dados sobre a luminosidade, umidade ou temperatura do quarto, permitindo que ele mostre essas informações ao usuário, e principalmente, tome decisões posteriormente a partir do resultado desta coleta de dados.

2.2. *Open source*

O Arduino é uma plataforma *open source,* termo que vem do inglês, e tem como significado “código livre”, o que se refere à um sistema que pode ser adaptado para diversos outros fins, por diversos usuários. De acordo com o site Canaltech, o termo *open source* foi criado em 3 de fevereiro de 1998 pela Open Source Initiative (Com Eric Raymound e outros fundadores) com o objetivo de apresentar um software livre para as empresas, e de uma maneira mais comercial, evitando discursos éticos e de direitos.

2.3. *I.D.E.*

A sigla em inglês IDE significa *Integrated Development Environment* que em português, é: Ambiente de desenvolvimento Integrado. De acordo com a PSafe (2014), *I.D.E.* é uma plataforma que une diversas ferramentas de utilização comum de muitos

programadores a fim de facilitar o processo de programação. Dentre estas ferramentas estão: Ferramentas de *debug*, compiladores, programação visual, suporte linguístico de algumas linguagens de programação, entre outros. A plataforma que utilizamos é a do próprio Arduino (Arduino IDE), e a linguagem de programação utilizada é C.

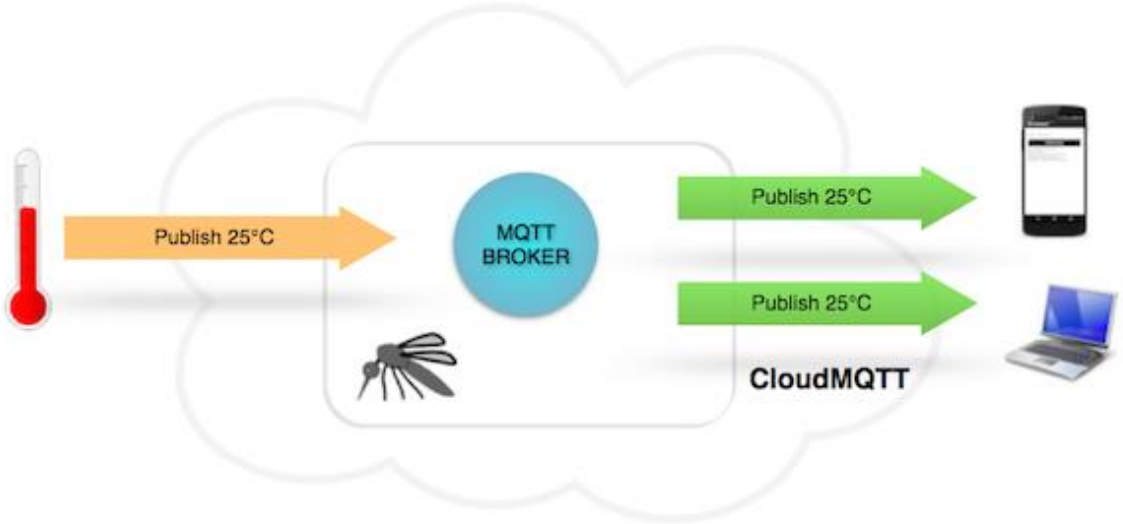
2.4. Protocolo *MQTT*

Para transmitir as mensagens do Arduino para o aplicativo, utilizamos o protocolo MQTT. Conforme Barros (2015), a sigla em inglês *MQTT* se refere a *Message Queue Telemetry Transport*, e ele foi criado pela empresa IBM no final da década de 90. Yuan (2017) define que "*MQTT* é um protocolo de rede leve e flexível que oferece o equilíbrio ideal para os desenvolvedores de *IoT" (**I**nternet **O**f **T**hings,* que na tradução em português significa Internet das Coisas). Nesse projeto, utilizamos um serviço que faz o uso do protocolo *MQTT* que é o CloudMQTT.

2.4.1 CloudMQTT

De acordo com a documentação disponibilizada no site oficial do CloudMQTT (2018), o CloudMQTT é um servidor em nuvem que gerencia o protocolo MQTT. Ele funciona como um intermediário, carregando uma mensagem utilizando o protocolo MQTT entre o usuário e o destino final ou vice-versa (No caso deste projeto, o destino é a placa NodeMCU), sem que eles precisem interagir com a "fila de mensagens" ao mesmo tempo: o CloudMQTT armazena essas mensagens até que o usuário ou o destino recebam, ou até chegar no tempo limite delas. A figura 3 ilustra como seria essa comunicação (Para melhor entendimento, o "broker" seria o intermediário CloudMQTT, o mosquito se refere ao protocolo MQTT [Que seria uma abreviação para *Mosquitto]*, e *Publish* significa "Publicar").

Figura 3 - Funcionamento do CloudMQTT



Fonte: CloudMQTT

2.5. Redis

Redis é o banco de dados utilizado nesse projeto. De acordo com a documentação disponibilizada no site oficial do Redis (2018), o Redis é um armazenamento de estrutura de dados em memória, que é utilizado como cache, banco de dados e intermediador de comunicação. Por trabalhar com informações armazenadas na memória RAM (*Random Acesss Memory*), os dados armazenados no Redis são voláteis. Em contrapartida, o Redis funciona de maneira extremamente performática, o que viabiliza sua utilização para o uso de dados em tempo real.

3. Empreendedorismo

Empreendedorismo, segundo o SEBRAE-SC (2018), é “a capacidade que uma pessoa tem de identificar problemas e oportunidades, desenvolver soluções e investir recursos na criação de algo positivo para a sociedade”; sendo assim, nem sempre um empresário é também um empreendedor. Além disso, de acordo com a teoria de Schumpeter (1982), o empreendedor tem um papel importantíssimo no processo de inovação. Conforme descreve Fuzetti (2009), o empreendedor é responsável por realizar “novas combinações”, que podem ser identificadas por: Introdução de um novo bem ou qualidade do bem; método de produção ou comercialização de um bem, abertura de novos mercados; conquista de novas fontes de oferta; e estabelecimentos de uma nova organização de qualquer indústria, abrangendo novas maneiras de se fazer algo existente.

3.1. Startup

De acordo com uma entrevista realizada pela Revista Exame, com o especialista em *Startups,* Gitahy (2018), *Startup* pode ser um grupo de pessoas trabalhando com uma idéia diferente e que consigam fazer dinheiro, como também uma empresa com custos de manutenção baixos e que conseguem gerar lucros cada vez maiores e de maneira rápida.

Gitahy define o conceito de startup como sendo:

- Um cenário de incerteza, onde é improvável o sucesso ou sustentação da empresa;
- O modelo de negócios da startup é o que a faz ganhar dinheiro;
- Ser repetível é ser capaz de entregar o mesmo produto em escala ilimitada, sem muitas customizações ou adaptações específicas para cada cliente;
- E por fim, a chave da Startup é ser escalável: ser capaz de crescer cada vez mais, sem que isso influencie no modelo de negócios.

3.1.1. Startup Enxuta

Conforme o site CODEMEC (2015) cita Alberone, para ele, uma startup enxuta possui 4 características:

- Desperdício: a startup deve sempre buscar reduzir o desperdício e a aprendizagem contínua;
- Modelo de negócios: O método usado no desenvolvimento do negócio para “se descobrir” e alcançar a maturidade pode ser aplicado em quase todas as organizações;
- Aprendizado contínuo: Construir um produto priorizando recursos e aprendizado, criar e seguir um determinado número de métricas que refletem bem a relação de causa e efeito, e coletar os dados gerados, a fim de aprender com eles e incorporam o aprendizado;
- Persistência: Assim que estabelecido o modelo de negócios, é importante que ele seja mantido até a empresa alcançar a maturidade necessária.

Também podemos notar essas características no livro de Ries (2012), onde o autor explica que o conceito de enxuta veio da revolução que Taiichi Ohno e Shigeo Shingo fizeram na Toyota, onde esse pensamento enxuto possuía como princípios: o aproveitamento do conhecimento e da criatividade de cada funcionário, juntamente com a redução dos tamanhos dos lotes e produções do tipo *Just in Time*, que ajudou no controle de estoque e aceleração do tempo de ciclo.

3.2. Canvas

Como Moura(2014) descreve:

> O Business Model Canvas é uma ferramenta de gerenciamento estratégico, que permite desenvolver e esboçar modelos de negócio novos ou existentes. É um mapa visual pré-formatado contendo nove blocos do modelo de negócios. O Business Model Canvas foi inicialmente proposto por Alexander Osterwalder baseado no seu trabalho anterior sobre Business Model Ontology.(MOURA, 2014, p. 2).

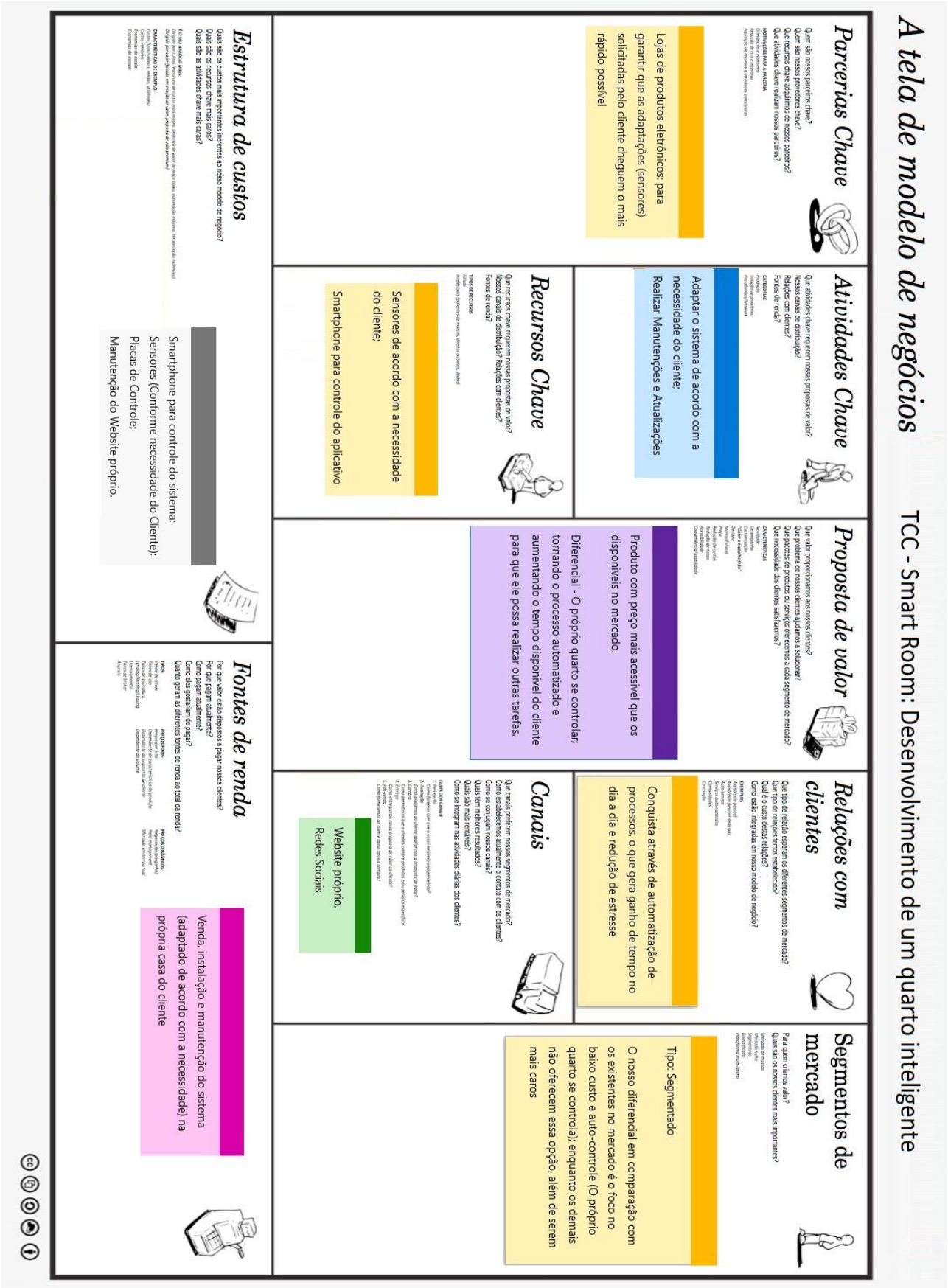
Este modelo de negócios é conhecido pela facilidade de visualizar vários aspectos do negócio em questão, facilitando o reconhecimento de oportunidades e ameaças. O canvas é composto em 9 partes: Segmentos de Clientes, Proposta de Valor, Canais, Relacionamento com Clientes, Fontes de Receita, Recursos Principais, Atividades-Chave, Parcerias Principais e Estrutura de Custo.

Osterwalder e Pigneur (2011) explicam cada um destes blocos da seguinte maneira:

- **Segmentos de Clientes**: Uma organização serve a um ou diversos segmentos de clientes.
- **Proposta de Valor**: Busca resolver os problemas do cliente e satisfazer suas necessidades, com propostas de valor.
- **Canais:** As propostas de valor são levadas aos clientes por Canais de comunicação, distribuição e vendas.
- **Relacionamento com clientes:** O relacionamento com clientes é estabelecido e mantido com cada segmento de clientes.
- **Fontes de Receita:** As fontes de receita resultam de propostas de valor oferecidas com sucesso aos clientes.
- **Recursos Principais:** Os recursos principais são os elementos ativos para oferecer e entregar os elementos previamente descritos.
- **Atividades-Chave:** ao executar uma série de Atividades-Chave.
- **Parcerias Principais:** Algumas atividades são terceirizadas e alguns recursos são adquiridos fora da empresa.
- **Estrutura de Custo:** Os elementos do Modelo de Negócios resultam na estrutura de custo.

A Figura 4 ilustra o modelo de negócios (Canvas) aplicando os tópicos mencionados acima para este projeto:

Figura 4 - Modelo de negócios TCC Smart Room: Desenvolvimento de um quarto inteligente



Fonte: Elaborado pelos autores

Explicando de maneira mais detalhada cada um dos tópicos descritos na figura anterior, aplicados para o nosso modelo de negócios descrito neste projeto, e que possui como o principal diferencial do nosso negócio diante dos concorrentes é a automatização sem interferência humana e preço baixo:

- **Proposta de valor:** Nosso objetivo é trazer um produto com um preço mais acessível que os já disponíveis no mercado, tendo como diferencial o fato do quarto se controlar sozinho, sem a necessidade de intervenção humana.
- **Relações com os clientes:** Iremos conquistar nossos clientes através da automatização dos processos, reduzindo o estresse e aumentando o tempo disponível do cliente; além de permitir que ele controle o seu quarto inteiro usando apenas o seu celular.
- **Segmentos de mercado - Tipo Segmentado:** Nosso maior diferencial em relação a concorrência é o baixo custo e auto-controle (sem necessidade de qualquer intervenção humana). Os demais além de caros, apenas permitem que o usuário controle usando um celular.
- **Canais:** Os canais que nossos clientes costumam utilizar são as redes sociais (Facebook, Instagram, Twitter…). Um diferencial seria ter o nosso próprio website, para descrever melhor os produtos e facilitar a divulgação do produto.
- **Atividades Chave:** Como principal atividade, teremos que adaptar o sistema de acordo com a necessidade de cada cliente; além de realizar manutenções (tanto no sistema utilizado no celular, quanto no aparelho físico instalado no quarto do cliente) e atualizações sempre que necessário.
- **Recursos Chave:** Placas de controle NodeMCU e sensores (De acordo com a necessidade do cliente); Smartphone (Para controlar o aplicativo).
- **Parcerias Chave:** Lojas de produtos eletrônicos (que trabalhem com os sensores adaptáveis para a placa NodeMCU). Tendo parceria com essas lojas, podemos reduzir o tempo de entrega do sistema personalizado do cliente, evitando o tempo de espera até a chegada dessas peças.
- **Estrutura de custos:** Placa de controle NodeMCU, sensores (De acordo com a necessidade de cada cliente), Smartphone (para controlar o sistema via aplicativo), e manutenção do website próprio.
- **Fontes de renda:** A renda nesse projeto vem da venda do sistema pronto (Aplicativo e Placa NodeMCU personalizada de acordo com a necessidade do cliente), instalação no quarto do cliente, e manutenção (Seja por defeito ou alteração).

4. Proposta de Solução

Este projeto está atrelado ao desenvolvimento de um controlador físico dos aparelhos de um quarto utilizando Arduino (neste caso, será utilizada a placa NodeMCU), e também a um sistema *mobile* para verificar o status desses aparelhos, bem como controla-los manualmente.

O controlador NodeMCU funciona por conta própria, e dá os comandos necessários para cada aparelho do quarto sem a necessidade de intervenção humana. O aplicativo *mobile* permite controlar cada um desses aparelhos manualmente, sem a necessidade de ir até o aparelho; além de mostrar o *status* do quarto, como temperatura e umidade atual.

4.1. Funcionalidades

O objetivo do sistema é fazer com que o usuário não tenha que se preocupar, portanto, ele deve necessariamente controlar a si mesmo (sendo este o principal diferencial do sistema), além de receber ordens do usuário quando necessário. A todo instante, deveram ser feitas análises em tempo real sobre como está a temperatura e umidade do ar e mostrar ao usuário o status atual do quarto, e sempre que necessário, tomar uma atitude.

Em relação ao que deve ser feito automaticamente: para o controle da temperatura, sempre que o quarto atingir uma temperatura determinada pelo próprio usuário, o controlador físico deverá ligar o ventilador (ou ar condicionado) automaticamente, e só desligar quando a temperatura estiver abaixo da que foi determinada pelo usuário. Já para o controle da umidade, sempre que a porcentagem de umidade do ar estiver abaixo da que foi determinada pelo próprio usuário, o controlador físico deverá ligar o umidificador automaticamente, e só desligar quando atingir a porcentagem de umidade definida anteriormente.

Quanto ao que o usuário pode fazer manualmente: ligar o modo automático, ligar o ventilador, ligar o umidificador e controlar as luzes do quarto através do aplicativo.

4.2. Levantamento dos Requisitos

Para o desenvolvimento do controlador físico utilizando NodeMCU, é necessário conexão via *Wi-Fi* para integrar os sensores, conectar-se a internet para utilizar o protocolo *MQTT* e comunicar com o aplicativo; um sensor de luz, para verificar as luzes do quarto e controlá-las utilizando o aplicativo; um sensor de temperatura, para verificar

em tempo real o *status* da temperatura do quarto e informar no aplicativo; e um sensor de umidade, para verificar em tempo real o *status* da umidade do quarto e informar no aplicativo.

Para a utilização do aplicativo mobile (que visa alterações manuais a serem feitas pelo usuário, bem como a correção de eventuais falhas), é necessário um aparelho celular com comunicador *Wi-Fi* para receber as informações dos sensores, e enviar comandos para o controlador físico.

A documentação de engenharia de software referente aos demais requisitos (Como BPMN, Casos de uso e Matriz de Rastreabilidade) é demasiadamente extensa para compor este arquivo. Dessa forma, os arquivos referentes a esta documentação poderão ser acessados online.

4.3. Custos do projeto

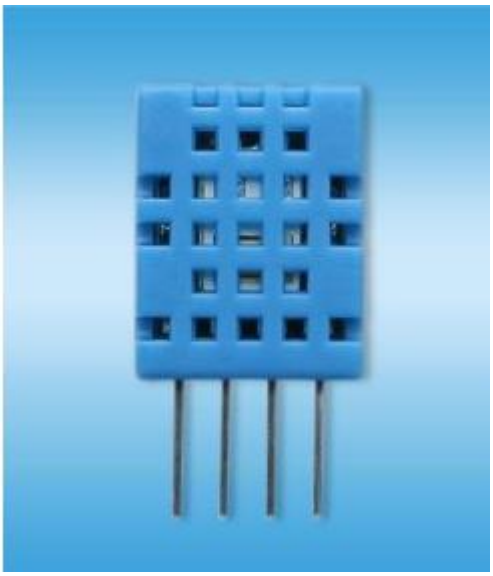
Um dos principais pontos do nosso projeto é a viabilização dele através do baixo custo, comprando materiais e sensores que tenham uma performance boa e um preço baixo. Para isso, utilizamos o sensor de Luz da marca SunRoom, que está ilustrado na Figura 5. O preço pago nele foi de R$ 13,00.

Figura 5 - Sensor de Luz.

Fonte: SunRoom

Utilizamos o sensor de umidade da marca Aosong, conforme ilustrado na Figura 6. O preço pago nele foi de R$ 10,80. Este sensor também possui a função de medir temperatura, mas neste projeto, ele é utilizado apenas para medir a umidade. A precisão de umidade dele varia aproximadamente 5% quando em temperaturas de 25ºC.

Figura 6 - Sensor de Umidade.



Fonte: Aosong

Utilizamos o sensor de temperatura da marca National Semiconductor, conforme ilustrado na Figura 7. O preço pago nele foi de R$ 11,99. A precisão de temperatura dele varia aproximadamente ¾ºC em temperaturas entre –55ºC e 150ºC.

Figura 7 - Sensor de temperatura.

Fonte: National Semiconductor

O NodeMCU utilizado (Ilustrado na Figura 2) é fabricado pela Espressif. O preço pago nele foi de R$ 28,00.

5. Desenvolvimento

Conforme mencionado no tópico 4.1. sobre os custos deste projeto, optamos por utilizar um sensor de temperatura, um de umidade, e um de luz. Todos esses sensores foram escolhidos com base na boa performance deles, em comparação com o baixo custo no mercado. Apesar do sensor de Umidade possuir a funcionalidade de medir a

temperatura também, optamos por deixar um sensor dedicado para essa função, a fim de trazer dados mais precisos, e evitar possíveis erros de leitura desses dados

No desenvolvimento do sensor de umidade, utilizamos o seguinte código da figura 8 para ler a umidade:

Figura 8 - Leitura de Umidade

```
digitalWrite(energy_dht11, HIGH);
delay(200);
float h = dht.readHumidity();
if (isnan(h)) {
  Serial.println("Failed to read from DHT sensor!");
} else {
   // Serial.print("DHT11: ");
   // Serial.println(h);
}
if (h < dht11_config) {
  digitalWrite(light_dht11, HIGH);
} else {
  digitalWrite(light_dht11, LOW);
}
snprintf(mqttMessage, 40, "%.2f", h);
if (client.publish("dht11", mqttMessage)) {
  //Serial.print("DHT11 = ");
  //Serial.println(mqttMessage);
} else {
  client.connect(mqttClientId, mqttUser, mqttPassword);
  delay(10);
  client.publish("dht11", mqttMessage);
}
digitalWrite(energy_dht11, LOW);
```

Fonte: Elaborado pelos Autores

Em relação ao desenvolvimento do sensor de temperatura, utilizamos o código da Figura 9 para ler a temperatura atual do quarto:

Figura 9 - Leitura de Temperatura

```
digitalWrite(energy_lm35, HIGH);
leitura = ((analogRead(sensor)/1024.0)*5000)/10;
if (leitura > lm35_config) {
  digitalWrite(light_lm35, HIGH);
} else {
  digitalWrite(light_lm35, LOW);
}
//Serial.print("LM35 - ");
//Serial.println(leitura);
snprintf(mqttMessage, 40, "%.2f", leitura);
if (client.publish("lm35", mqttMessage)) {
  //Serial.print("LM35 = ");
  //Serial.println(leitura);
} else {
  client.connect(mqttClientId, mqttUser, mqttPassword);
  delay(10);
  client.publish("lm35", mqttMessage);
}
digitalWrite(energy_lm35, LOW);
```

Fonte: Elaborado pelos autores.

Quanto ao desenvolvimento do sensor de luz, utilizamos o código descrito na Figura 10 para verificar a luminosidade atual do quarto:

Figura 10 - Leitura de Luminosidade

```
digitalWrite(energy_ldr, HIGH);
leitura = analogRead(sensor);
if(leitura > ldr_config) {
  digitalWrite(light_ldr, HIGH);
} else {
  digitalWrite(light_ldr, LOW);
}
delay(50);
snprintf(mqttMessage, 40, "%.2f", leitura);
if (client.publish("ldr", mqttMessage)) {
//  Serial.print("LDR = ");
//  Serial.println(leitura);
} else {
  client.connect(mqttClientId, mqttUser, mqttPassword);
  delay(10);
  client.publish("ldr", mqttMessage);
}
digitalWrite(energy_ldr, LOW);
```

Fonte: Elaborado pelos autores

Para que os dados capturados pelos sensores pudesse ser mostrado pela tela de configuração, o servidor orquestrador se conectou ao CloudMQTT como um receptor de mensagens. Assim que ele recebe os novos valores dos sensores, o servidor armazena estes dados no Redis para que a tela de configuração possa acessar estes dados com alta disponibilidade e performance.

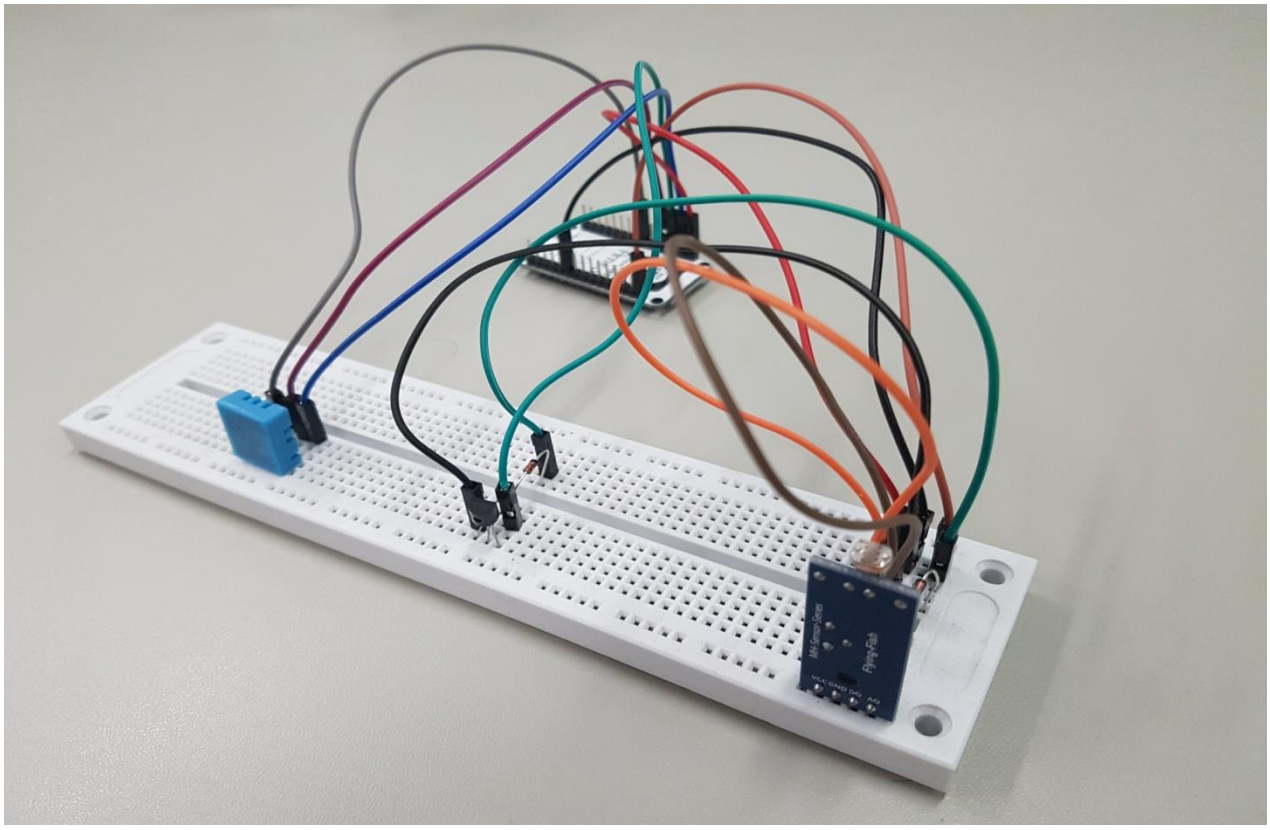
Da mesma maneira, para que os LEDs saibam quando eles deveriam ativar os seus respectivos componentes, a tela envia para o servidor os dados de configuração, que são salvos no Redis. A cada iteração do código da placa NodeMCU, ele busca os dados de configuração armazenados, e compara com os valores capturados pelos sensores. Quando os valores capturados pelos sensores ultrapassar os limites anteriormente definidos na tela de configuração, os LEDs são acionados.

Vale lembrar que os comandos para que o usuário ligue algo manualmente não estão totalmente integrados com o servidor. Para formar a parte gráfica da tela do aplicativo, utilizamos o framework de estilo Material Design, desenvolvido pelo Google. Para o funcionamento básico da tela, utilizamos a biblioteca de código aberto ReactJS, construída pelo Facebook para a linguagem JavaScript.

6. Análise dos Resultados

Após a conclusão do desenvolvimento, passamos os código para a placa NodeMCU para que ela trabalhe de forma independente. A Figura 11 mostra a fotografia da placa montada:

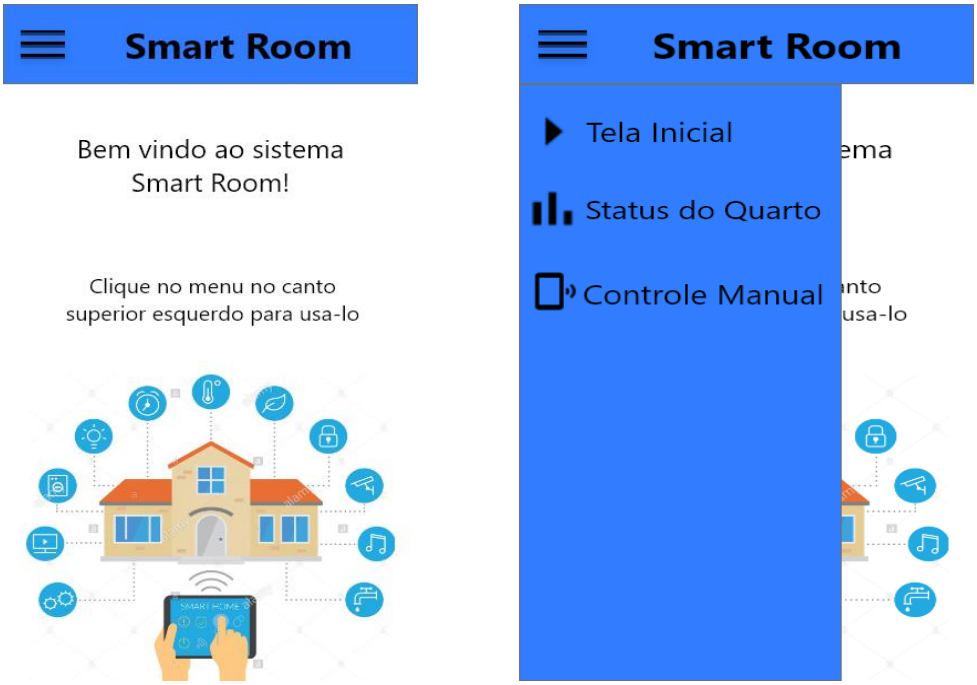
Figura 11 - Placa NodeMCU montada



Fonte - Elaborado pelos autores

Os sensores de umidade, temperatura e luminosidade funcionam em conjunto sem nenhum conflito, lendo os dados na maior precisão possível. Na Figura 12 é possível observar o protótipo de como ficaria o aplicativo no celular do usuário, juntamente com cada funcionalidade que ele oferece.

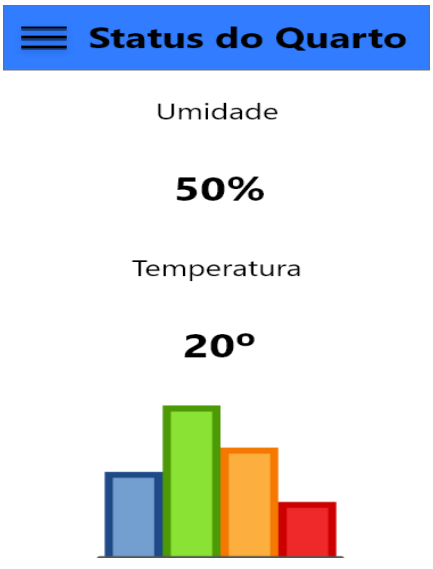
Figura 12 - Tela inicial com menu



Fonte: Elaborado pelos autores

Já na figura 13, verificam-se as informações que são lidas pela placa:

Figura 13 - Informações sobre temperatura, umidade e luminosidade



Fonte: Elaborado pelos autores

Na Figura 14, geramos o protótipo da tela no qual o usuário poderia ligar o ventilador, umidificador ou a luz do quarto de forma manual:

Figura 14 - Botões de comando do aplicativo

Fonte: Elaborado pelos autores

7. Considerações Finais

O objetivo desse projeto era de entregar uma solução para os problemas relacionados a automatização domiciliar (tendo o quarto como principal parte), e com foco no baixo custo e boa performance. Antes de iniciar esse projeto, tínhamos como objetivos automatizar mais partes da casa, como por exemplo:

- **Cozinha:** Tornar a geladeira inteligente, informando ao usuário tudo o que tinha dentro dela, e criando uma lista de compras automáticas, com base na preferência de cada usuário; Ter um sensor de gás próximo ao fogão, para evitar riscos de incêndios e explosões.
- **Jardim:** Possuir sensores de umidade nas plantas a fim de lembrar o usuário de regá-las sempre que necessário.
- **Animais:** Possuir um sensor dedicado para controlar o nível da água e da comida dos animais da casa, para lembrar o usuário sempre que ele precisasse colocar mais comida ou água.
- **Energia:** Ter sensores dedicados para controlar o gasto com luzes acesas pela casa toda, e avisar o usuário sempre que ele esquecesse alguma luz ligada; Sensores

dedicados para aparelhos que consomem muita energia (Como ferro de passar roupas, microondas…) a fim de lembrar o usuário sempre que ele esquecesse algum deles ligado.

- **Segurança:** Mostrar via aplicativo o status de todas as portas da casa (Aberta, fechada ou trancada), para evitar que usuário esqueça alguma porta destrancada de forma indesejada, a fim de evitar possíveis furtos enquanto ele dorme ou viaja, por exemplo.

Devido ao curto tempo para desenvolvimento, custo para comprar todas as peças necessárias, e dificuldade para aprender sobre a linguagem e todas essas peças, optamos por focar apenas na automatização de um quarto: os comandos para ligar e desligar os aparelhos de forma manual existem, mas não foi possível finalizar a integração desses comandos com o servidor. A partir deste projeto, acreditamos que todas as ideias dos tópicos mencionados anteriormente possam ser concretizadas.

O nosso principal desafio na parte do desenvolvimento foi para aprender sobre como cada um dos sensores funcionam, e como fazer eles funcionarem em conjunto; além de aprender sobre a linguagem de programação C para ser utilizada na placa NodeMCU para que isso tudo acontecesse da maneira correta. Além disso, foi necessário aprender como conectar o Javascript utilizado tanto para criar a tela de informações quanto no servidor que orquestra a comunicação entre a tela e a placa NodeMCU com o protocolo MQTT, e salvar os dados coletados em um banco de dados Redis, que é voltado para acessos de alta performance em consultas simples.

7.1. Pesquisa

Quando realizamos uma pesquisa para a identificação dos principais problemas, entrevistamos diversas pessoas para entender a necessidade de cada um, e pedir sugestões sobre o que gostariam de automatizar em suas casas. Aqui estão os 5 principais entrevistados:

A primeira pessoa entrevistada não possuía um problema prioritário, mas tinha como adversidades: Preguiça de limpar a casa, esquecer o ferro de passar roupas ligado, esquecer o gás da cozinha acesso.

A segunda pessoa, tinha como seu maior problema a manutenção da casa. Para essa pessoa, o sistema deveria: Avisar se há algum vazamento na casa; Avisar quando há alguma luz queimada; Avisar quando o gás estiver vazando; Avisar quando as portas

estiverem abertas ou destrancadas; Avisar quando o chuveiro queimar; Avisar quando a comida e a água dos animais estiverem abaixo de um certo nível.

Já a terceira pessoa, tinha como seus maiores problemas os filhos. Para essa pessoa, o sistema deveria: Possuir uma etiqueta de radiofrequência *(RFID)*, que assim que o filho chegasse em casa (Com a etiqueta dentro da mochila, por exemplo), o sistema avisaria os pais via *SMS;* Possuir um detector de ruídos, para alertar quando o bebê estiver chorando.

O quarto entrevistado, tinha como seu maior problema identificar os custos da casa. Sendo assim, o sistema deveria calcular os gastos de energia e água consumida no mês, separados por cada cômodo da casa, a fim de identificar com precisão qual parte da casa consumia mais, para facilitar a redução dos gastos.

O quinto entrevistado tinha como seu maior problema os custos com suprimentos e comida. Para essa pessoa, o sistema deveria realizar um inventário em tempo real de todos os suprimentos e comidas existentes na casa, e gerar uma lista de compras, informando o que deveria ser comprado juntamente com a quantidade, com base nos valores definidos pelo usuário para cada item.

Além das necessidades identificadas pelos nossos 5 principais entrevistados, e também para fins de eventos futuros, sugerimos a criação de um comando por voz para executar os comandos, a fim de facilitar o uso das pessoas que possuem alguma deficiência de visão ou de movimento das mãos, ou qualquer outra deficiência que dificulte a operação do nosso sistema.

8. Conclusão

Iniciamos o projeto tendo como objetivo automatizar um quarto (tendo um preço acessível em mente), fazendo com o que o usuário não se preocupe com as temperaturas nem a umidade dele (Além de poder verificar se a luz do quarto está acesa). Para a correção de eventuais exceções não previstas no sistema, criamos a funcionalidade de ligar e desligar o ventilador, umidificador ou luz acoplada, porém, a integração dessa funcionalidade com as peças acopladas e o servidor não foi finalizada.

Em relação aos custos, somando todos os gastos do projeto, temos o total de R$ 63.79. Este é um custo consideravelmente baixo em relação às demais soluções

oferecidas no mercado, e que está dentro do orçamento da maioria da população. Sendo assim, o objetivo de baixo custo foi alcançado.

Outro de nossos objetivos era de servir de exemplo para facilitar e ajudar futuros projetos em relação a automatização domiciliar, que hoje, é algo considerado raro e distante da maioria da população ainda. Atingimos o objetivo de automatizar um quarto com custo baixo, e descrevemos como todo esse processo foi realizado; além de mencionar diversas outras soluções que podem surgir a partir desse mesmo projeto .

Referências

AOSONG. DHT11 Manual. Disponível em: <https://akizukidenshi.com/download/ds/aosong/DHT11.pdf>. Acesso em 11 de Agosto de 2018.

BADOCO, Matheus. MELO, Mahezer. Documentação de Engenharia de Software. Disponivel em: <https://github.com/SirFlyann/smart-room> Acesso em 20 de Setembro de 2018.

BARROS, Marcelo. MQTT - Protocolos para IoT. Disponivel em: <https://www.embarcados.com.br/mqtt-protocolos-para-iot/> Acesso em 7 de Setembro de 2018.

BLOG SEBRAE-SC. O que é Empreendedorismo? Disponível em: <http://blog.sebrae-sc.com.br/o-que-e-empreendedorismo/>. Acesso em 10 de Junho de 2018.

CANALTECH. O que é Open Source? Disponivel em: <https://canaltech.com.br/produtos/O-que-e-open-source/>. Acesso em 7 de Setembro de 2018.

CODEMEC. 4 características de uma startup enxuta. Disponivel em: <http://ibmec.org.br/geral/4-caracteristicas-de-uma-startup-enxuta/>. Acesso em 10 de Junho de 2018.

FUZETTI, Diana. Fuzetti. Empreendedorismo Na Visão Schumpeteriana Como Fator De Estratégia De Inovação Empresarial: Estudo Em Uma Metalúrgica. Disponível em: <http://www.unimep.br/phpg/mostraacademica/anais/7mostra/3/131.pdf>. Acesso em 10 de Junho de 2018.

GITAHY, Yuri. Resposta em entrevista dada à revista Exame, O que é uma Startup?. Disponível em: <https://exame.abril.com.br/pme/o-que-e-uma-startup/>. Acesso em 10 de Junho de 2018.

MOURA, Luiz. Memoria Seminario sobre Canvas Model com Alexander Osterwlader. Disponivel em: <http://www.academia.edu/9592727/Memoria_Seminario_sobre_Canvas_Model_com_Alexander_Osterwlader_-_by_Luiz_Rolim>. Acesso em 10 de Junho de 2018.

NOVAES, Rafael. O que é e para que serve IDE? Disponível em: <https://www.psafe.com/blog/o-que-serve-ide/> Acesso em 4 de Setembro de 2018.

OSTERWALDER, Alexander.; PIGNEUR, Yves. Business Model Generation - inovação em modelos de negócios: um manual para visionários, inovadores e revolucionários. Alta Books, 2011.

RIES, Eric. A startup enxuta. Leya, 2012. Disponivel em: <https://books.google.com.br/books?id=vLiUj1h5fhkC&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>. Acesso em 10 de Junho de 2018.

SCHUMPETER, Joseph, conforme citado por FUZETTI, Diana. Disponível em: <http://www.unimep.br/phpg/mostraacademica/anais/7mostra/3/131.pdf>, no capítulo 1 - Introdução, paragrafo 2. Acesso em 10 de Junho de 2018.

SITE ARDUINO. Arduino Products. Disponivel em: <https://www.arduino.cc/en/Main/Products>. Acesso em 11 de Agosto de 2018.

SITE CLOUDMQTT. CloudMQTT Overview. Disponível em: <https://www.cloudmqtt.com/images/cloudmqtt_overview.png>. Acesso em 14 de Setembro de 2018.

SITE CLOUDMQTT. Documentation. Disponivel em: <https://www.cloudmqtt.com/docs.html>. Acesso em 14 de Setembro de 2018.

SITE FILIPEFLOP. LM35. Disponivel em: <https://uploads.filipeflop.com/2017/07/LM35.jpg> . Acesso em 11 de Agosto de 2018.

SITE FILIPEFLOP. NodeMCU. Disponivel em: <https://uploads.filipeflop.com/2017/07/218fd7d9-8002-b5bc-6d12-fb2ee9837ee4.jpg>. Acesso em 11 de Agosto de 2018.

SITE REDIS. Introduction to Redis. Disponivel em: <https://redis.io/topics/introduction>. Acesso em 16 de Setembro de 2018.

SITE SUNROM. Documentation. Disponivel em: <https://www.sunrom.com/get/443700>. Acesso em 11 de Agosto de 2018.

THOMSEN, Adilson. O que é Arduino? Disponível em: <https://www.filipeflop.com/blog/o-que-e-arduino/> Acesso em 27 de Agosto de 2018.

YUAN, Michael. Conhecendo o MQTT. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html> Acesso em 7 de Setembro de 2018.