

# I JORNADA CIENTÍFICA

## DA FACULDADE META

### APLICATIVO PARA MONITORAMENTO DE SALAS E CONTROLE DE EQUIPAMENTOS ELETRÔNICOS UTILIZANDO CONCEITO DE IOT

Rainério Júnior da Silva Costa <sup>1</sup>

Victor Henrique de Araújo M. de Oliveira <sup>2</sup>

Hugo Pinheiro da Silva <sup>3</sup>

Ernesto Gomes Pereira Júnior <sup>4</sup>

Richardson Salomão de Araújo <sup>5</sup>

EIXO: ENGENHARIA DE COMPUTAÇÃO

**RESUMO:** O presente estudo tem como objetivo geral o desenvolvimento de um protótipo para monitoramento de salas e controle de equipamentos eletrônicos utilizando o conceito de Internet das coisas - IOT. Esta aplicação foi pensada observando a logística de monitoramento e acionamento de dispositivos nas salas da Faculdade de Tecnologia do Amapá - META. Assim, inicialmente foram realizadas pesquisas bibliográficas para embasamento teórico, e posteriormente, foi iniciada a construção do protótipo, onde utilizou-se de materiais como uma câmera, uma *Sonoff* em modo *DIY* (que significa Faça Você Mesmo, do inglês *Do It Yourself*) e ferramentas de desenvolvimento de software para a construção de uma API e uma interface *Web*. Com isso, obteve-se como resultado uma aplicação *Web* integrando recursos de vídeo do ambiente, bem como os dispositivos que podem ser controlados em cada sala de maneira remota, chegando assim ao objetivo deste estudo.

**Palavras-chave:** Automação; Internet das Coisas; IOT; Sonoff.

## INTRODUÇÃO

---

<sup>1</sup> Acadêmico do Curso de Engenharia de Computação

<sup>2</sup> Acadêmico do Curso de Engenharia de Computação

<sup>3</sup> Graduação em Engenharia Elétrica (UNIFAP, 2018), especialização em Gestão e Docência no Ensino Superior (FATECH, 2018)

<sup>4</sup> Graduação em Sistemas de Informação (Estácio, 2008), especialização em Análise da Tecnologia da Informação (Faculdade Meta, 2010), especialização em Análise e Sistemas de Banco de Dados (Atual, 2018)

<sup>5</sup> Graduação em Redes de Computadores (Estácio, 2006), graduação em Engenharia Elétrica (UNIFAP, 2018), mestrado em Engenharia Elétrica (UFPA, 2022)

A busca pelo desenvolvimento de tecnologias que auxiliem e facilitem a execução de atividades humanas é uma ideia que já vem sendo desenvolvida há muito tempo. No cenário tecnológico atual, é perceptível diversos recursos utilizados no cotidiano como assistentes virtuais, controle de dispositivos via internet, serviços automatizados, etc. Um conceito que ganhou força há alguns anos e se consolidou dentro deste modelo tecnológico foi o de Internet das coisas - *IOT*.

Na concepção de Santos *et al.* (2016, p. 2) a *IOT* pode ser entendida como,

[...] uma extensão da Internet atual, que proporciona aos objetos do dia-a-dia (quaisquer que sejam), mas com capacidade computacional e de comunicação, se conectarem à Internet. A conexão com a rede mundial de computadores viabilizará, primeiro, controlar remotamente os objetos e, segundo, permitir que os próprios objetos sejam acessados como provedores de serviços.

A *IOT* além de facilitar a conexão de dispositivos gerando praticidade em sua utilização, pode trazer benefícios relacionados a redução do desperdício de energia elétrica. Cada vez mais, as empresas buscam formas de reduzir esse desperdício e os custos gerados por ele, dessa forma, a *IOT* pode ser um agente importante a ser utilizado para monitoramento e controle desses recursos.

De acordo com Pereira *et al.* (2017 *apud* PÉREZ-LOMBARDA, p. 1),

Os setores que mais consomem energia elétrica no mundo são: indústria, transportes, agricultura, serviços e residências. No entanto, ultimamente, com o crescimento populacional, aumento do nível de conforto nas residências, e outros fatores, a quantidade de energia consumida nas residências e prédios comerciais tem se equiparado aos níveis de consumo da indústria e transportes.

No Brasil, de acordo com dados do balanço energético nacional de 2021, disponibilizado pela Empresa de Pesquisa Energética (EPE), o consumo de energia elétrica no setor comercial equivale a aproximadamente 50% de toda a energia consumida no país.

Esse alto consumo energético muito se dá também pelo alto índice de energia elétrica desperdiçada no país. Dados ABESCO alegam que entre os anos de 2015 a 2017, o desperdício de energia elétrica correspondeu a cerca de R\$ 61,7 bilhões de reais.

Nesse sentido, verificou-se um problema de logística e monitoramento de salas na Faculdade de Tecnologia do Amapá – META, onde diariamente, um funcionário deve se dirigir às salas onde ocorrerão as aulas para ligar ou desligar os equipamentos eletrônicos utilizados, principalmente centrais de ar e projetores. No entanto, esse trabalho manual leva

muito tempo e esforço humano, principalmente em horários de início e término das aulas, onde a demanda é maior.

Esse controle, apesar de funcional, mostra-se pouco eficiente pela demora para que sejam ligados/desligados todos os equipamentos das salas, além de dificultar que sejam identificadas situações onde os equipamentos estão ligados sem necessidade, como em casos onde os alunos saem mais cedo ou trocam de sala, pois exige que o funcionário esteja presente fisicamente no local. Este uso desnecessário dos equipamentos resulta em um grande desperdício de energia e por conseguinte alto custo financeiro para a instituição.

Nesta perspectiva, pode-se elencar possíveis alternativas focadas em eficiência energética que busquem minimizar tal problema, que vão desde campanhas de conscientização até o uso de tecnologias de controle e monitoramento. Na visão de SONEGO *et al.* (2016 *apud* Wu *et al.*) “o processo de eficiência energética está condicionado à integração de novas tecnologias, como por exemplo, as TIC, sendo a Internet das Coisas considerada ideal para este ambiente” (p. 81).

Assim, este estudo tem como objetivo geral desenvolver um protótipo de uma aplicação *web* onde será possível visualizar imagens de câmeras das salas e controlar dispositivos eletrônicos de maneira remota utilizando conceitos de *IOT*, simulando assim o ambiente de salas da instituição. Para esse fim, foram definidos os seguintes objetivos específicos a serem executados, sendo estes:

- Desenvolver uma API para controle e gerenciamento de fluxo de dados de câmeras e regras de negócio da aplicação.
- Configurar Sonoff (dispositivo usado para acionamento dos equipamentos eletrônicos) para operar em modo *DIY*<sup>6</sup>, para que seja possível controlá-lo pela aplicação.
- Desenvolver interface da aplicação e integrar com a API para utilização dos usuários.

## 1 REFERENCIAL TEÓRICO

A pesquisa tem como objetivo desenvolver um protótipo de uma aplicação *web* onde será possível visualizar imagens de câmeras das salas e controlar dispositivos eletrônicos de

---

<sup>6</sup> O modo *DIY* (*Do It Yourself*) é um recurso disponibilizado pela Sonoff, em alguns modelos, que permite sua utilização através de uma API baseada em HTTP, possibilitando a usuários e desenvolvedores, controlar o dispositivo por meio de requisições HTTP. (SONOFF, 2022).

maneira remota utilizando conceitos de *IOT*. Assim, buscou-se construir um embasamento teórico que pudesse agregar e interligar esse processo da teoria à prática.

## 1.1 MICROPROCESSADORES E MICROCONTROLADORES

Os microprocessadores e microcontroladores são a base dos sistemas computacionais que conhecemos hoje, estão diretamente ligados aos computadores atuais e sistemas de controle e automação, por exemplo. Nesse sentido, Lima e Villaça (2012) descrevem um microprocessador como:

um circuito integrado composto por inúmeras portas lógicas, organizadas de tal forma que permitem a realização de operações digitais, lógicas e aritméticas. Sua operação é baseada na decodificação de instruções armazenadas em uma memória programável. A execução das instruções é ditada por um sinal periódico (relógio - clock) (p.3).

Já os dispositivos microcontroladores na visão dos autores podem ser descritos como:

um sistema microprocessado com várias funcionalidades (periféricos) disponíveis em um único chip. Basicamente, um microcontrolador é um microprocessador com memórias de programa, de dados e RAM, temporizadores e circuitos de clock embutidos (p.9).

Aguiar (2018) acrescenta informações a respeito da estrutura do microcontrolador que, inicialmente, possuía basicamente funções de entrada e saída (I/O), sendo implementado a cada nova versão. Dentre os avanços, tem-se a implementação de memória RAM e memória EPROM para programas e dados e circuito de oscilador (*clock*). Também houveram implementações das interfaces de comunicação como serial e USB, e ainda interfaces de rede, (*Ethernet, WI-FI e Bluetooth*). A integração de todos esses recursos em um único dispositivo trouxe vantagens em sua utilização, pois o tornou mais acessível, principalmente para uso em sistemas de baixo custo.

Nesse ambiente, pode-se enfatizar as placas de desenvolvimento que segundo Lima (2021) se popularizaram e passaram a ser comercializadas impulsionadas pela criação e avanços da microinformática e seus meios de produção, após esse determinado acontecimento os valores dos componentes eletrônicos tiveram uma redução, fortalecendo setores tecnológicos, acarretando no crescimento da área da automação com foco na prototipagem eletrônica.

Dentre as placas de desenvolvimento, podemos frisar o *Raspberry Pi*, que é considerado um sistema embarcado de baixo custo, com tamanho reduzido, executado no sistema *Raspbian* que é baseado no GNU/Linux Debian. Pode executar instruções, pois possui processador incorporado e possui *interface General Purpose Input/Output* (GPIO), que permite

utilização de diversos sensores e módulos, além de ser possível utilizar módulos *WI-FI* (SANTOS E ZAMBERLAN, 2021).

Para Lima (p.54, 2021) “Devido ao grande sucesso dessas placas, ocorreu a popularização da chamada internet das coisas, onde equipamentos agora podem ser controlados e monitorados remotamente.”.

Há ainda o conceito de sistemas embarcados, onde Chase (p.3, 2007) explica ser um sistema “dedicado a uma única tarefa e interage continuamente com o ambiente a sua volta por meio de sensores e atuadores”.

Para Oliveira (p.40, 2017) os sistemas embarcados “começaram a se desenvolver e ocupar tarefas que antes eram feitas de forma manual. Alguns desses componentes começaram a se destacar na automatização dos dispositivos e sistemas por agregar mais funcionalidade no próprio componente”.

Na visão de Lima (2021) um sistema embarcado pode ser visto como uma integração de hardware e software que pode ser modificada por meio de softwares, elencando *Raspberry pi*, Arduino e FPGA como os principais sistemas embarcados empregados na engenharia.

## 1.2 AUTOMAÇÃO

Dentro do contexto da automação, seja industrial, residencial ou comercial, o conceito de sensores e atuadores é fundamental, pois por meio destes é possível captar sinais do ambiente e assim fazer a interação com os sistemas. De maneira mais técnica os sensores podem ser conceituados como

dispositivos que detectam estímulos, medem e monitoram grandezas físicas e eventos (temperatura, umidades etc.), convertendo-as em um valor passível de manipulação por sistemas computacionais. São eles que encaminham as informações aos controladores sobre algum evento, para que os controladores possam enviar os comandos adequados para os atuadores (ACCARDI E DODONOV, p. 157, 2012 *apud* ALMEIDA, 2009).

Na perspectiva de Soares (p.19, 2012) os “atuadores podem atuar ativamente no ambiente físico, em resposta a situações indicadas pelos dados coletados pelos nós sensores.”. Na visão do mesmo, atualmente os dispositivos atuadores estão sendo utilizados para monitoramento de dados, e através desse monitoramento os atuadores são capazes de tomar decisões sobre uma determinada situação.

Outro conceito bastante importante no âmbito da automação e que tem grande atuação dentro da ideia do projeto são os controladores *On-Off*, que são controladores simples,

possuindo apenas dois estados possíveis, 0 e 1. Pode-se citar como exemplo os relés, que de acordo com Santos e Zamberlan (p.3, 2021)

são dispositivos elétricos que tem como função produzir modificações repentinas, mas predeterminadas em um ou mais circuitos elétricos de saída. O relé tem um circuito de comando, que no momento em que é alimentado por uma corrente, aciona um eletroímã (bobina) que faz a mudança de posição de outro par de contadores, que estão ligados a um circuito ou comando secundário.

Atualmente, outro controlador muito utilizado no mercado é a *Sonoff*. Este dispositivo é um tipo de relé que pode ser conectado a redes sem fio, operando em tensões de 100 ou 220 Volts (Bivolt), ligado diretamente à rede elétrica, sendo um interruptor versátil, sem fio e de baixo custo, segundo os autores.

A partir desse embasamento sobre alguns componentes que fazem parte do conceito de automação, pode-se definir automação como “dotar um equipamento de meios que lhe permitam realizar seu controle automaticamente, sem a intervenção humana.” (BAYER; ECKHARDT; MACHADO. p.11, 2011). Este conceito por ser amplo, pode ser subdividido em categorias, sendo elas, residencial ou predial, comercial e industrial.

Vale destacar a automação residencial e predial que nos últimos anos vem sendo muito utilizada, principalmente pela capacidade de reduzir custos com energia, melhorar o controle e monitoramento de recursos, trazendo ganhos financeiros e conforto para as pessoas. Isso é possível devido ao uso de sensores que captam dados do ambiente gerando informações que são utilizadas para realizar ações dentro do ambiente automatizado. (RÊGO, 2020).

### 1.3 PROTOCOLOS DE REDE

O conceito de protocolo de rede é proposto por Elias e Lobato (p.45, 2013) como

um conjunto de regras que controla a interação de duas máquinas ou dois processos semelhantes ou com funções semelhantes. Para que dois computadores se comuniquem, é necessário que usem o mesmo protocolo. Sem protocolos, o computador não pode reconstruir, no formato original, a sequência de bits recebida de outro computador.

No modelo atual de comunicação de rede, utiliza-se uma arquitetura de organização e padronização chamada arquitetura TCP/IP, onde os autores explicam, ser formada por 4 camadas, sendo elas: aplicação, transporte, rede e interface de rede (rede física), responsáveis por cada etapa da troca de informações.

Neste sentido, e com foco nos objetivos propostos, serão abordados a seguir, os principais protocolos que estarão sendo utilizados durante a execução da aplicação, sendo eles: TCP, UDP, HLS e HTTP.

Os protocolos TCP e UDP, dentro da arquitetura TCP/IP, fazem parte da camada de transporte. O TCP segundo Rios (2012), é um dos principais utilizados nas aplicações da internet e tem como características a robustez, um alto controle de fluxo, garantia de entrega e organização dos pacotes enviados. Já o UDP não garante a entrega dos pacotes nem a ordem em que são entregues, no entanto, tem uma taxa de entrega mais rápida que o TCP, sendo muito utilizado em sistemas peer-to-peer.

Na comparação entre os dois protocolos Kauffmann (p.2, 2020) explana,

UDP é um serviço de rede simplificado onde os pacotes são entregues o mais rápido possível, mas correm o risco de ser perdidos (dropados). O UDP oferece suporte à entrega unpara-muitos usando multicast. Por outro lado, o TCP fornece controle de fluxo e garante a entrega, mas o custo disso é a entrega limitada.

Outro protocolo fundamental para a funcionalidade da aplicação é o HTTP Live Stream (HLS) que segundo Kauffmann (p.4, 2020)

é um protocolo desenvolvido pela Apple para fornecer streaming usando um servidor da web padrão. O stream de vídeo é dividido em “pedaços” de alguns segundos cada. O decodificador baixa os pedaços como arquivos do servidor web com transações HTTP padrão, usando uma lista de reprodução chamada “manifesto”.

Ainda segundo o autor, “o HLS tem uma latência muito alta, 3-4x o tamanho do bloco, normalmente de 2 a 30 segundos. Também é altamente robusto e a escalabilidade pode ser feita usando servidores *web* externos”.

O protocolo HTTP faz parte da camada de aplicação e tem como função tratar pedidos e respostas entre cliente e servidor (RIOS, 2012). O autor também comenta sobre a utilização do HTTP, que é bem comum no dia a dia. Este protocolo é utilizado no acesso à internet, por exemplo, no acesso de sites através de um navegador (Chrome, Firefox, Safari) ou outros serviços por meio de aplicativos em diversos dispositivos.

#### 1.4 JAVASCRIPT/TYPESCRIPT (NODE JS E REACT JS)

A linguagem de programação JavaScript segundo o site Mdn (2020), “é uma linguagem de script orientada a objeto, multiplataforma. É uma linguagem pequena e leve. Dentro de um ambiente de host (por exemplo, um navegador *web*) o JavaScript pode ser ligado aos objetos deste ambiente para prover um controle pragmático sobre eles”.

O Typescript, segundo seu site oficial, possui uma sintaxe extra para JavaScript para suportar uma integração mais rigorosa com o editor de código (TYPESCRIPT, 2022). Este recurso adicional permite uma melhor leitura de código e integração entre equipe de desenvolvimento utilizando tipagem mais rígida e recurso de intellisense<sup>7</sup> do editor.

Esta linguagem por ser altamente popular, possui diversos frameworks e bibliotecas disponíveis, na qual pode-se citar o React Native e Node Js, sendo possível a utilização tanto no lado do cliente, quanto no lado do servidor.

Segundo Silva e Cardoso (p.5, 2021 *apud* FLANAGAN 2011),

Node é um rápido interpretador JavaScript baseado em C++ com ligações às APIs de baixo nível do Unix para trabalhar com processos, arquivos, soquetes de rede e também para APIs de clientes e servidores HTTP. Com exceção de alguns métodos síncronos, especialmente nomeados, os vínculos do Node são todos assíncronos e por padrão os programas Node nunca são bloqueados, o que significa que eles normalmente são dimensionados adequadamente e lidam com cargas altas de maneira eficaz.

Com o NodeJs é possível utilizar JavaScript no lado do servidor na construção de diversos serviços e APIs. Segundo Peters e McClennen (p.4, 2015), as APIs

forneem um conjunto de protocolos e ferramentas para construção de software. Dentro no contexto de bancos de dados, uma API é uma especificação de como fazer solicitações remotas para dados (através de um protocolo padrão, como HTTP) usando uma semântica que não requer qualquer conhecimento do software de banco de dados e que retorna dados formatados de maneiras que não são específicos para qualquer uso final.

Corroborando com este conceito de API e APIs *Rest*, o site oficial da AWS (2022), descreve que “REST significa Transferência Representacional de Estado. REST define um conjunto de funções como GET, PUT, DELETE e assim por diante, que os clientes podem usar para acessar os dados do servidor. Clientes e servidores trocam dados usando HTTP”.

Para a construção da interface de aplicativos *web* pode-se utilizar o React JS que é um framework lançado pela equipe de desenvolvimento do Facebook (atual Meta) em 2013, sendo um dos mais utilizados atualmente na criação de interfaces *web*. Tem como características ser escrito em linguagem JavaScript e sua utilização é baseada em componentes. (REACT JS, 2022).

## 1.5 INTERNET DAS COISAS - IOT

---

<sup>7</sup> “IntelliSense é um termo geral para vários recursos de edição de código, incluindo: conclusão de código, informações de parâmetros, informações rápidas e listas de membros. Os recursos IntelliSense às vezes são chamados por outros nomes, como "conclusão de código", "assistência de conteúdo" e "dica de código.” (VISUAL STUDIO CODE, 2022).

Segundo Santos e Sales (*apud* ATZORI; IERA; MORABITO, 2010),

a Internet das Coisas - IoT advém do conceito de presença generalizada em torno das pessoas e de uma variedade de coisas ou objetos, através de Radio Frequency IDentification - RFID, sensores, atuadores, gadget como smartphones, tablet, televisores, pulseiras e relógios inteligentes, etc., por meio de esquemas de endereçamento exclusivos que são capazes de interagir uns com os outros e cooperar com os seus vizinhos para alcançar objetivos comuns.

Este conceito é corroborado por Magrani no livro *A internet das coisas* (2018), que visualiza a *IOT* como uma infraestrutura global capaz de prover serviços avançados através de conexões entre dispositivos físicos e virtuais, apoiado em tecnologias de informação e conceitos de interoperabilidade em constante evolução.

O autor ainda acrescenta algumas das diversas aplicações de sistemas de *IOT*, como por exemplo, a utilização na área da saúde buscando aprimorar o monitoramento e a interação entre paciente e médico ou sistemas de automação residencial onde o usuário pode controlar equipamentos para realizar ações e preparar ambientes mais confortáveis antes de chegar em casa. Pode-se citar também a aplicação voltada à eficiência energética, explorada por Sonego et al. (2016), na qual busca-se monitorar ambientes, coletar dados e aplicar soluções em *IOT* com objetivo de obter uma maior eficiência energética.

## **2 MATERIAL E MÉTODO**

Este estudo se trata de uma pesquisa aplicada apoiada em levantamentos bibliográficos e procedimentos experimentais. A pesquisa aplicada, segundo Andrade (p.110, 2010), “visa às aplicações práticas, com objetivo de atender as exigências da vida moderna. Nesse caso, sendo o objetivo contribuir para fins práticos, pela busca de soluções para problemas concretos.”, e se adequa ao tipo de pesquisa por se tratar do desenvolvimento de uma aplicação a fim de solucionar ou minimizar a problemática proposta.

O foco central da ideia é operar no ambiente residencial e predial de pequeno porte, pois estes possuem um quantitativo de salas e equipamentos (centrais de ar, iluminação) favoráveis à implementação da aplicação, no entanto, para esta etapa elaborou-se um protótipo para realização de testes em pequena escala.

### **2.1 TRABALHOS CORRELATOS**

Nesta etapa foram apresentados alguns projetos e estudos que aplicam conceitos de *IOT* e automação, colaborando com a proposta apresentada neste estudo.

### **2.1.1 Sistema de controle de acesso e acionamento da iluminação e ar-condicionado das salas de aula**

Este estudo, trata-se de um trabalho de conclusão de curso com tema “Desenvolvimento de um sistema de controle de acesso e acionamento da iluminação e ar-condicionado das salas de aula, utilizando internet das coisas (iot)”, no curso de Engenharia Elétrica pela Universidade do Estado do Amazonas, apresentado por Sousa (2019).

Sua pesquisa apresentou como problemática a dificuldade do gerenciamento manual de uma grande quantidade de salas de sua faculdade, em relação à abertura de salas e ligação dos equipamentos utilizados nas aulas.

Com isso, o estudo teve como objetivo

desenvolver e implementar um protótipo de um sistema de controle automatizado, utilizando Internet das coisas (IoT), nas salas de aula da EST, capaz de realizar remotamente a abertura e fechamento das salas, além de realizar remotamente o desligamento e acionamento da refrigeração e iluminação das mesmas, utilizando Internet das Coisas (IoT). (SOUSA, 2019).

Então, foi desenvolvido um sistema de fechadura capaz de ser acionado por meio de um cartão RFID, utilizando DOIT ESP32 Dev Kit V1, um microcontrolador que tem a capacidade de comunicação sem fio através do wireless e Bluetooth. Para este fim, foram instaladas fechaduras com capacidade de leitura de RFID.

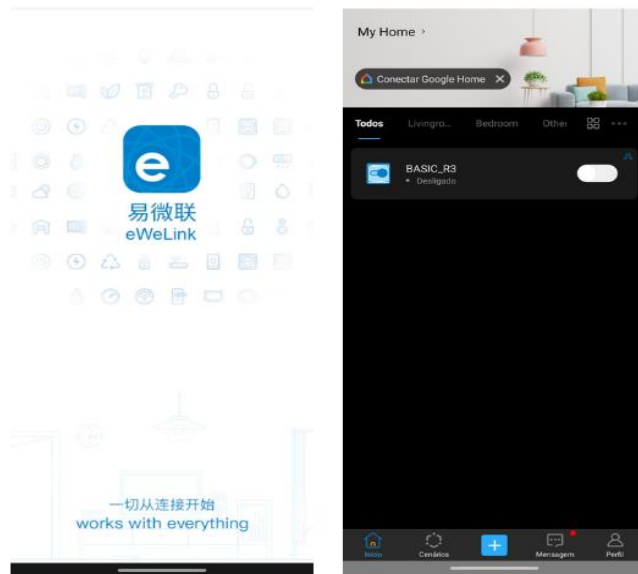
Em sua utilização, cada professor recebeu seu cartão para que o próprio pudesse abrir e fechar sua sala nos horários de aula. Após a abertura da sala, um sinal era enviado ao microcontrolador para acionar de forma automática os sensores de atuadores responsáveis por ligar e desligar as lâmpadas e centrais de ar da sala.

### **2.1.2 eWeLink**

Este projeto, segundo seu site oficial, se trata de uma aplicação onde é possível emparelhar diferentes dispositivos que tenham suporte ao eWeLink, sendo uma solução para a interoperabilidade de diversas marcas (EWELINK, 2022).

Possui suporte a dispositivos móveis, além de disponibilizar uma plataforma *web* em seu plano premium. Essa aplicação permite operar diferentes dispositivos em um único ambiente, sem a necessidade de estar conectado à mesma rede. Abaixo (figura 1) segue a representação da tela inicial do aplicativo mobile.

Figura 1 - Tela inicial do aplicativo eWeLink



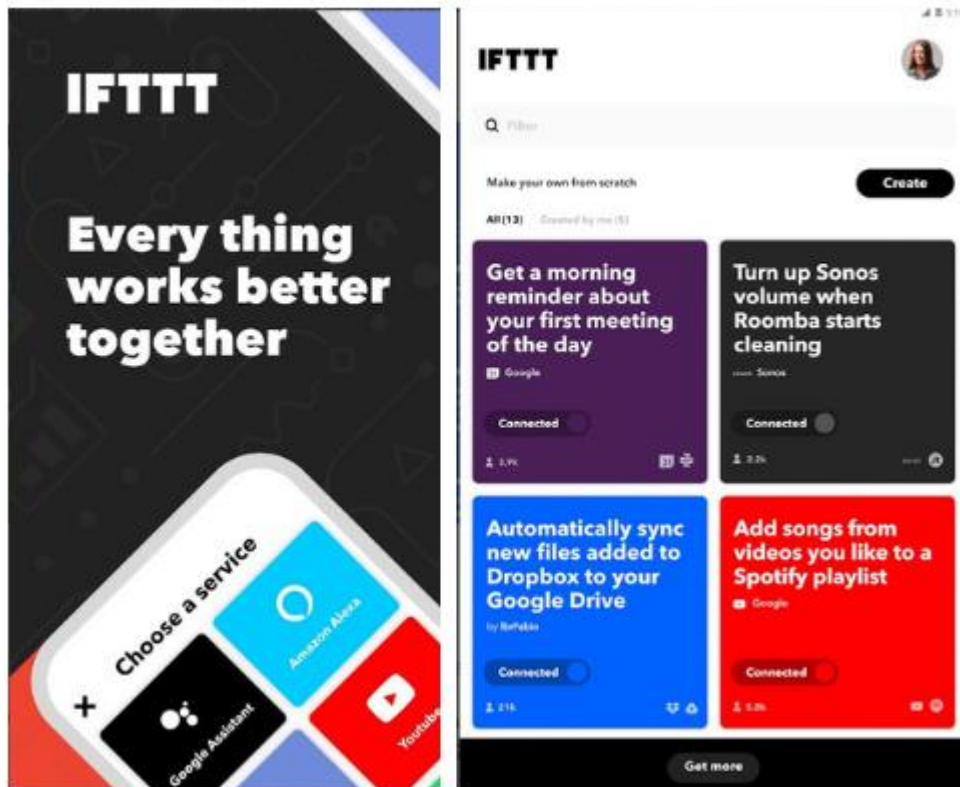
Fonte: Autores (2022).

### 2.1.3 IFTTT

O IFTTT é um serviço disponibilizado para as plataformas Android, IOS e *Web*, que permite criar conexões entre dispositivos trabalhando como eventos, onde determinada ação deve ser executada após uma condição pré-configurada. (TECHTUDO, 2022).

Dentre as vantagens pode-se citar o fato de ser um serviço gratuito, a facilidade de entendimento, e o suporte a diversos sistemas e serviços. Como desvantagem, tem-se o fato de algumas automações levarem mais tempo do que outras para gerar efeito e a falta de feedback em caso de desativação de canais.

Figura 2 - Tela inicial do aplicativo IFTTT



Fonte: Play Store (2022).

## 2.2 DIFERENCIAL DA PROPOSTA APRESENTADA

Analisando as ideias expostas anteriormente, pode-se citar como diferencial deste estudo em relação a ideia apresentada no tópico 3.1.1, a possibilidade de visualizar as salas por meio das câmeras e um maior controle dos administradores da instituição no que diz respeito ao acesso às salas, pois, o protótipo possibilita que apenas os setores responsáveis possam fazer a abertura e fechamento das salas.

Outro ponto diferencial, mas em relação ao projeto eWeLink, refere-se a este não permitir integração de algumas marcas de dispositivos, como por exemplo as câmeras da marca Intelbras. Esta marca é uma das mais presentes no ramo de segurança eletrônica, representando cerca de 44% do *market share*<sup>8</sup> brasileiro. (VIEIRA, 2021).

Já com relação ao projeto IFTTT, pode-se citar o fato de não haver a criação de ambientes para controle de dispositivos, nem a visualização de câmeras, trabalhando apenas com a automação por meio de eventos.

---

<sup>8</sup> “*Market Share* é a participação da empresa no mercado, ou seja, representa a porcentagem do valor de mercado de uma organização perante à concorrência.” (OLIVEIRA, 2022).

### 2.3 ELABORAÇÃO DO PROTÓTIPO

Na elaboração do protótipo foram utilizados alguns materiais, listados na tabela a seguir (tabela 1), com sua representação visual e custo financeiro investido.

Tabela 1 - Materiais utilizados no protótipo

MATERIAIS	DESCRIÇÃO	VALOR	IMAGEM
<i>Sonoff</i> BASICR3	Responsável por ligar/desligar a lâmpada.	R\$ 90,00	
Câmera IP Intelbras IM3 C	Responsável pela geração de imagens do ambiente.	R\$ 240,00	
Lâmpada de LED	Equipamento eletrônico acionado pela <i>Sonoff</i>	R\$ 10,00	
Bocal para lâmpada	Para ligação da lâmpada de LED	R\$ 6,00	
Cabos de energia	Para conectar os dispositivos elétricos	-----	
Plugue para tomada	Para alimentação elétrica	-----	

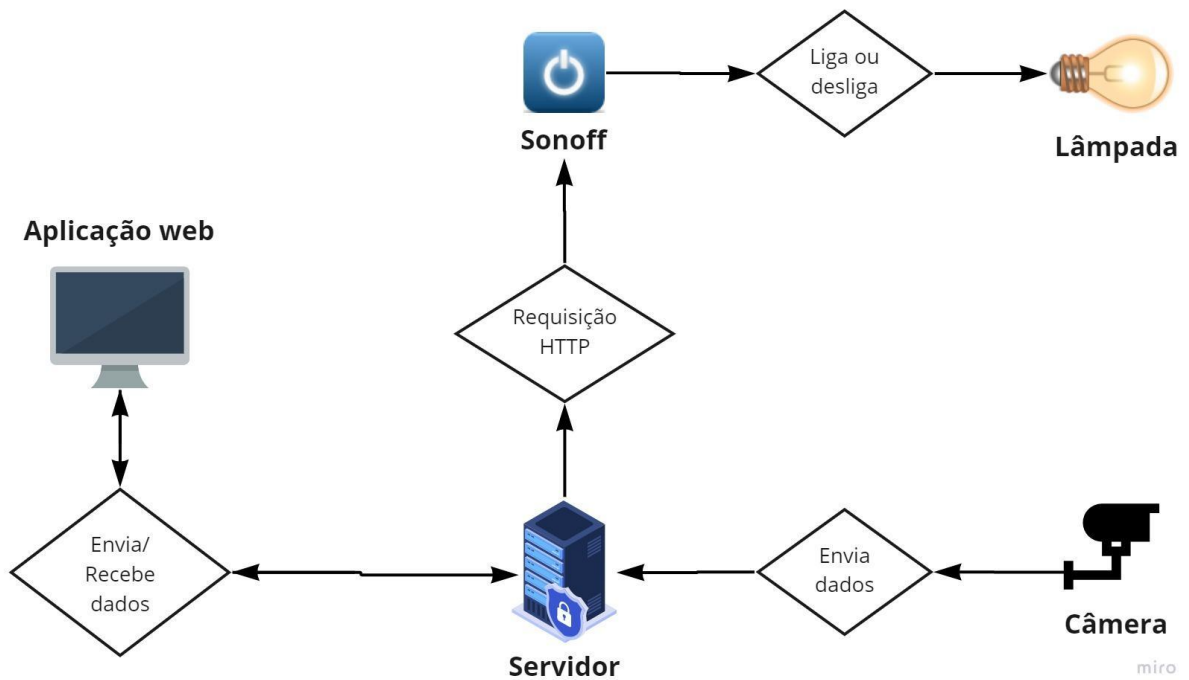
Computador	Responsável por executar a aplicação	-----	-----
Servidor	Ambiente que irá interligar e gerenciar todos os componentes do sistema	-----	-----
<b>TOTAL</b>		<b>R\$ 346,00</b>	

Fonte: Autores (2022).

No processo de construção, inicialmente, foram realizados testes para obtenção de imagem da câmera IP e seu envio para a API para exibição no navegador *web*. Posteriormente, foram realizados testes utilizando a *Sonoff* buscando entender como seria sua utilização e configuração dentro do sistema. Nesta etapa verificou-se os requisitos técnicos referentes à tensão de operação do equipamento, bem como foi realizada a instalação da *Sonoff* responsável pelo controle do equipamento.

A seguir temos uma representação da integração de todos os componentes do protótipo (figura 3). Resumidamente, a parte central do sistema é a API que irá conectar a interface da aplicação com os equipamentos de câmera e permitirá o envio de requisições HTTP do cliente para acionar a lâmpada conectada à *Sonoff*.

Figura 3- Integração do sistema.



Fonte: Autores (2022).

## 2.4 DESENVOLVIMENTO DE CÓDIGO FONTE

O processo de desenvolvimento de código foi executado em duas etapas, *back-end* e *front-end*, utilizando Typescript e aplicando o conceito de *node streams* nos processamentos da API para a interface da aplicação.

A construção do *back-end* deu-se com NodeJs, Docker<sup>9</sup> e o software FFMPEG<sup>10</sup> para a conversão de dados da câmera. Primeiramente foi criado o ambiente com node JS e em seguida configurado o container que executaria a aplicação junto ao software de conversão.

O FFMPEG levou a necessidade da utilização de Docker, pois deve ser instalado diretamente na máquina em uso, assim, com Docker é possível isolar o sistema em um container, simulando um ambiente computacional e permitindo uma melhor administração de recursos.

<sup>9</sup> “O Docker é uma plataforma de código aberto que permite que os desenvolvedores construam, implantem, executem, atualizem e gerenciem *contêineres* — componentes padronizados e executáveis que combinam código-fonte de aplicativo com bibliotecas e dependências do sistema operacional (OS) necessários para executar esse código em qualquer ambiente.” (IBM, 2022).

<sup>10</sup> FFMPEG é um *software* utilizado para codificação, decodificação, conversão, reprodução e transmissão de áudio e vídeo em diversas plataformas como linux, windows, mac. (FFMPEG, 2022).

No envio dos dados da câmera para a interface, por meio da API, utilizou-se as *node streams*, que são basicamente uma forma de processar dados por demanda no Node.JS. Nesse caso, a API aguarda o processamento de parte dos dados e então os envia para o cliente ou para o próximo processo, sem que seja necessário processar todos os dados em memória.

Para a criação do *front-end* da aplicação foi utilizado o React JS, pois é uma biblioteca que fornece uma série de praticidades na construção de interfaces, possibilitando ainda integrar vários recursos úteis nas aplicações.

A interface deste protótipo então se dá por uma tela de login, tela de cadastro de dispositivos (câmeras e *sonoffs*), tela para criação de salas e a *dashboard*, onde será possível organizar as salas por ambientes e acionar os dispositivos eletrônicos.

## 2.5 REALIZAÇÃO DE TESTES

Durante os testes do protótipo, teve-se como foco as principais funcionalidades, ou seja, a geração de imagens das câmeras e o controle de equipamentos eletrônicos por meio dos botões de ação presentes na interface.

Assim, foram cadastrados previamente a câmera e a *sonoff* utilizadas com seus respectivos endereços de IP, e ainda cadastradas outros dispositivos fakes apenas para uma melhor representação visual do sistema.

Nos testes de câmera, o objetivo foi observar e tentar atingir o menor tempo de resposta na geração de imagens na interface do protótipo. Para isso, foi utilizado um cronômetro para monitoramento do tempo e o processo consistia em realizar um movimento em frente a câmera e marcar em quanto tempo seria reproduzido esse movimento na imagem exibida na interface do protótipo.

Em relação ao teste de utilização do sistema, foi inicializada a API, utilizando o ambiente local de desenvolvimento, e executado a interface da aplicação em um navegador *web*, o usuário ao realizar o login deve observar a tela de *dashboard* com todas as câmeras listadas, exibindo as imagens que estão sendo enviadas por meio da API.

O próximo teste foi relacionado ao controle dos dispositivos por meio da *Sonoff*. Assim, cada dispositivo estava listado com seu nome de identificação e um botão representando o estado atual do mesmo. Basicamente, nesta etapa buscou-se verificar se os dispositivos listados apresentaram seu estado atual real, e se foi possível controlar os dispositivos por meio dos botões do sistema, ligando ou desligando uma *Sonoff*.

Para a última etapa de testes, foi verificada a tela de dispositivos cadastrados, mais especificamente a tabela de *Sonoffs*. Esta lista contém botões de ação, onde também deve ser possível acionar estes dispositivos cadastrados. Assim, foram testados os botões, que acionaram

com sucesso os equipamentos. Dessa forma, chegou-se a cobertura de todos os testes referentes às principais funcionalidades presentes neste protótipo.

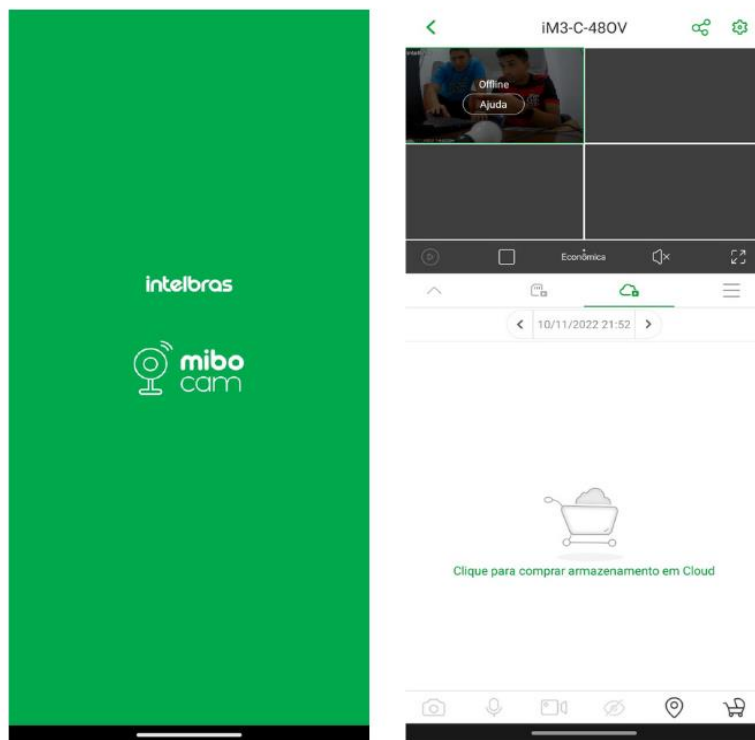
## ANÁLISE E DISCUSSÃO DOS RESULTADOS

Com base no objetivo deste estudo que é desenvolver um protótipo de uma aplicação *web* onde será possível visualizar imagens de câmeras de salas e controlar dispositivos eletrônicos de maneira remota, temos a seguir a análise e resultados obtidos em cada etapa de elaboração, passando pela construção da API e configuração de dispositivos até a criação da interface do protótipo. O repositório do sistema completo pode ser acessado através do link do Github: [https://github.com/Raineriojr/tcc\\_ofc](https://github.com/Raineriojr/tcc_ofc).

### 2.6 DESENVOLVIMENTO DA API PARA CONTROLE E GERENCIAMENTO DE FLUXO DE DADOS

Inicialmente, foi necessário realizar a configuração da câmera em seu aplicativo nativo, para isso, foi necessário a utilização de um smartphone com o aplicativo Mibo Cam, mostrado a seguir (figura 4). Este aplicativo disponível para Android e IOS, foi desenvolvido pela Intelbras com intuito de centralizar informações e dados, que auxiliam no manuseio de seus dispositivos de câmera.

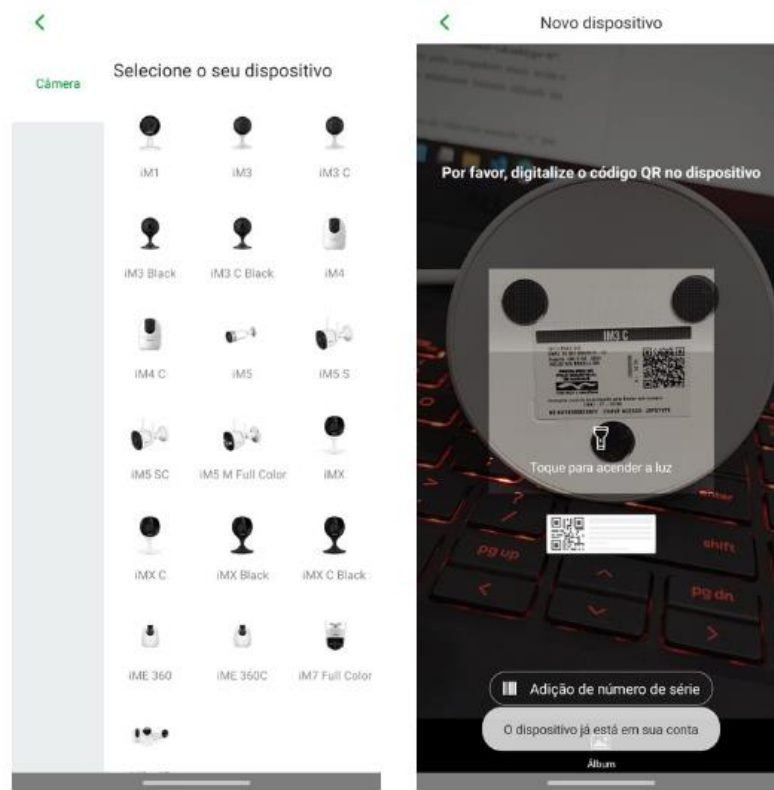
Figura 4 - Tela inicial e *dashboard* do aplicativo Mibo Cam



Fonte: Autores (2022).

No aplicativo Mibo Cam, foi criada primeiramente uma conta e, em seguida, foi selecionado o modelo da câmera e escaneado o *QR code* localizado na base da câmera para conectá-la ao app, como mostrado na figura 5. Esse processo precisa da utilização de uma conexão *wi-fi* com frequência de 2,4 GHz e os dispositivos devem ser conectados na mesma rede para que ambos tenham a mesma faixa de IP.

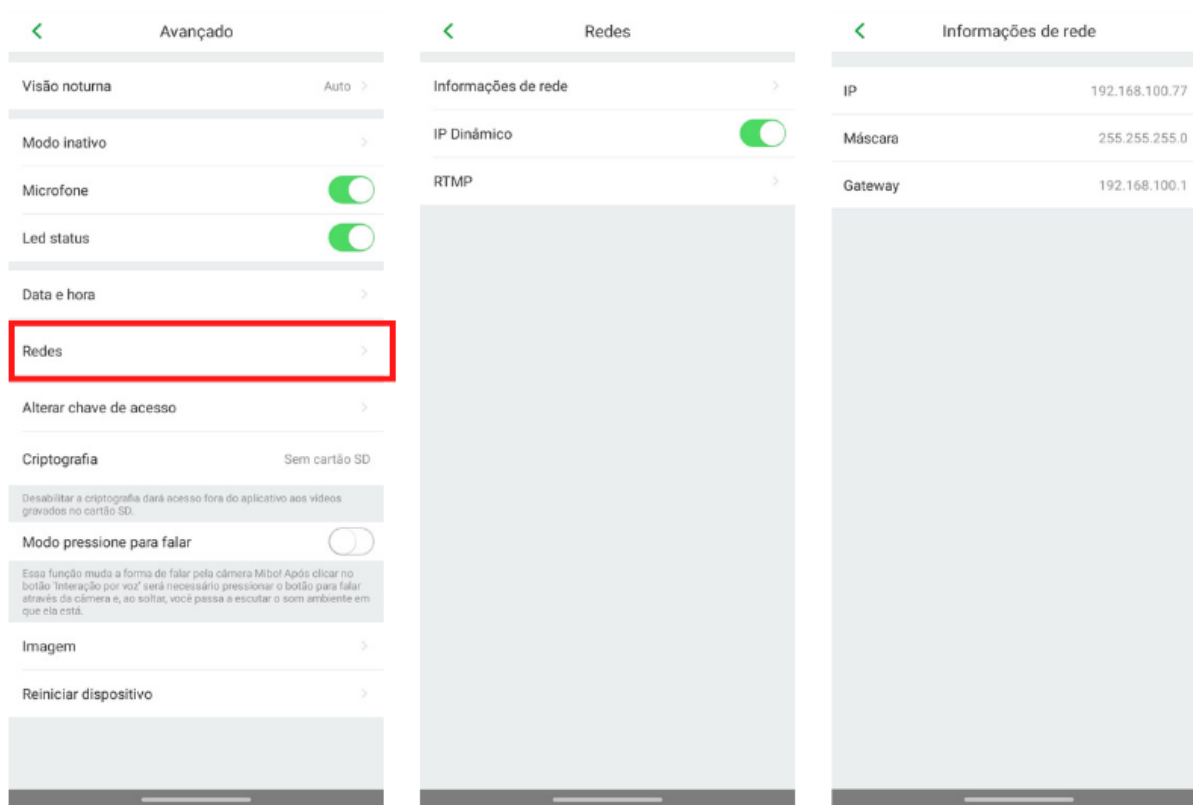
Figura 5 - Seleção de câmera e leitura de *QR code*



Fonte: Autores (2022).

Após essa configuração de rede, foi acessado o menu de configurações do app para gerenciar o IP que foi obtido pela câmera. É possível utilizar um IP de modo manual/fixo ou automático, exposto a seguir (figura 6). Essa configuração é necessária para a obtenção do número de IP que irá possibilitar a comunicação da câmera dentro do sistema.

Figura 6 - Configuração e informações de rede da câmera



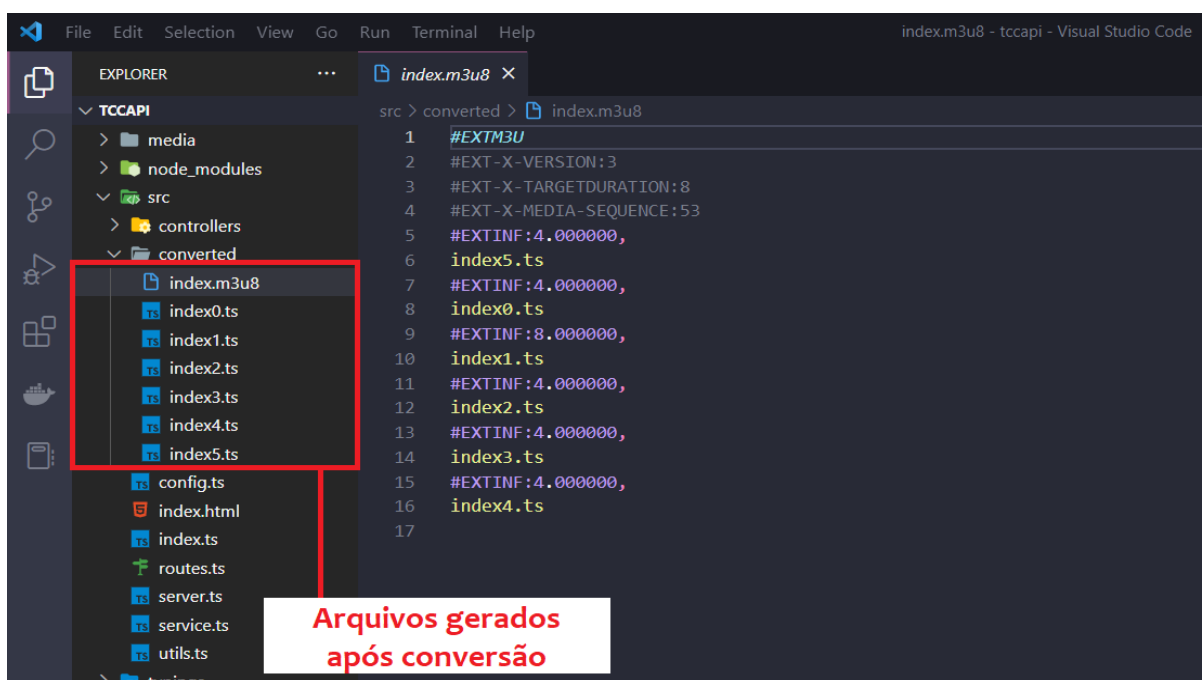
Fonte: Autores (2022).

Os dados recebidos da câmera IP utilizam protocolo RTSP<sup>11</sup> com a url **“rtsp://[usuário]:[senha]@[endereçoIP]:554/cam/realmonitor?channel=1&subtype=0”**. No entanto, este protocolo não é reconhecido diretamente pelos navegadores atuais, assim o FFMPEG o converte para o protocolo HLS, protocolo atualmente bastante utilizado em *streams* de vídeo como citado por Kauffmann (2020).

O processo de conversão gera fragmentos de arquivos com extensão “.ts”, que são cortes de dois segundos de duração de vídeo recebido da câmera, e que são gerenciados por outro arquivo de extensão “.m3u8”, um arquivo de texto que contém a organização dos demais arquivos para que seja feita a leitura sequencial dos mesmos, como exibido a seguir (figura 7).

<sup>11</sup> *Real Time Streaming Protocol (RTSP)* é um protocolo de transmissão de áudio/vídeo entre dois pontos finais. Este protocolo já dominou o cenário de *stream*, porém já existem protocolos mais atuais e mais utilizados. Mesmo assim, o RTSP ainda é muito utilizado em circuitos de câmera e circuito fechado de televisão. (WOWZA, 2022).

Figura 7 - Arquivos gerados após conversão de vídeo do protocolo RTSP para HLS



Fonte: Autores (2022).

Com essas configurações realizadas, foram iniciados os primeiros testes para geração de imagens buscando otimizar esse processo. Assim, foram realizados testes para verificação de tempo de resposta da câmera dentro da interface do protótipo. Os dados coletados nesse teste são apresentados na tabela 2, a seguir.

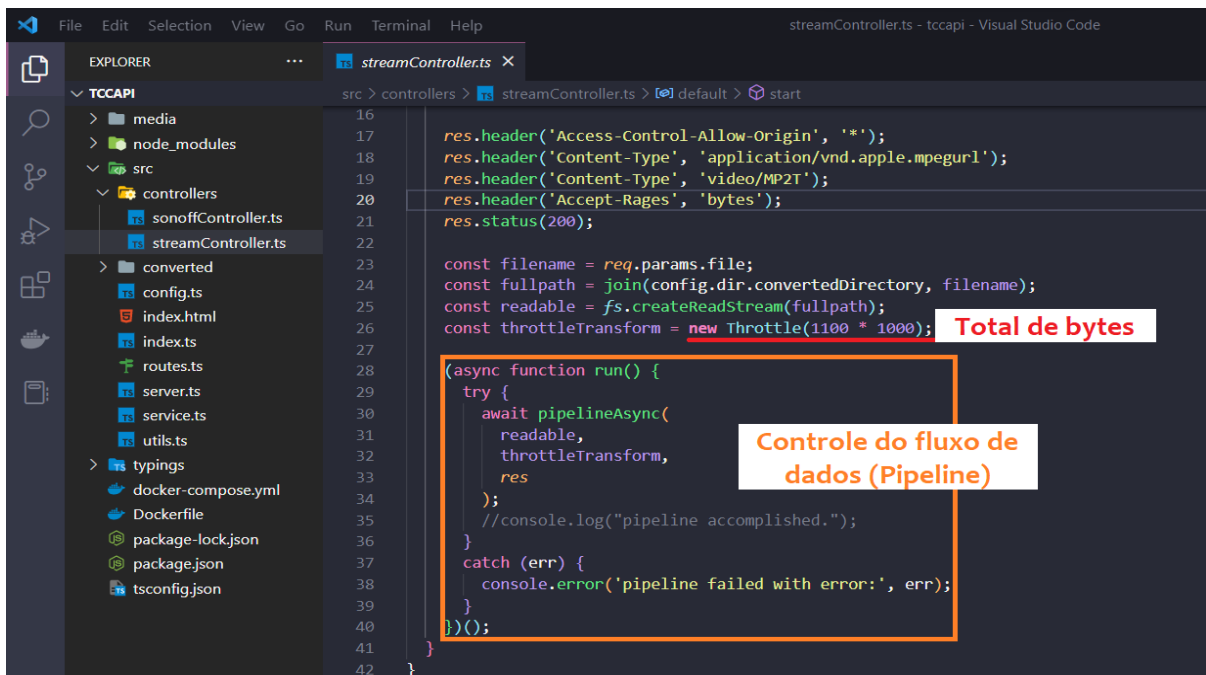
Tabela 2 - Teste de tempo de resposta da câmera para a interface

TESTE	TEMPO DE RESPOSTA EM SEGUNDOS
1	18 segundos
2	15 segundos
3	19 segundos
4	16 segundos
5	18 segundos
<b>MÉDIA</b>	<b>17,2 SEGUNDOS</b>

Fonte: Autores (2022).

Os arquivos gerados ao serem solicitados pelo cliente, os arquivos são enviados como *stream*, aguardando que uma quantidade de *bytes* seja processada, para então enviar os dados ao cliente que passa a receber esses fragmentos reproduzindo-os como um único arquivo de vídeo. A figura a seguir (figura 8), representa o trecho de código responsável por esse controle do fluxo de dados.

Figura 8 - Trecho de código responsável pelo controle de fluxo de dados



```
16
17
18 res.header('Access-Control-Allow-Origin', '*');
19 res.header('Content-Type', 'application/vnd.apple.mpegurl');
20 res.header('Content-Type', 'video/MP2T');
21 res.header('Accept-Ranges', 'bytes');
22 res.status(200);
23
24 const filename = req.params.file;
25 const fullpath = join(config.dir.convertedDirectory, filename);
26 const readable = fs.createReadStream(fullpath);
27 const throttleTransform = new Throttle(1100 * 1000);
28
29 (async function run() {
30   try {
31     await pipelineAsync(
32       readable,
33       throttleTransform,
34       res
35     );
36     //console.log("pipeline accomplished.");
37   } catch (err) {
38     console.error('pipeline failed with error:', err);
39   }
40 })();
41
42 }
```

Fonte: Autores (2022).

Com isso, pôde-se realizar o processo de obtenção de imagem das câmeras, configurando seu endereço de IP por meio do aplicativo Mibo Cam e realizando a conversão do fluxo recebido, através do software FFMPEG, para então repassar esses dados para o cliente final. Dessa forma, foi possível exibir a imagem das câmeras na interface *web* do protótipo.

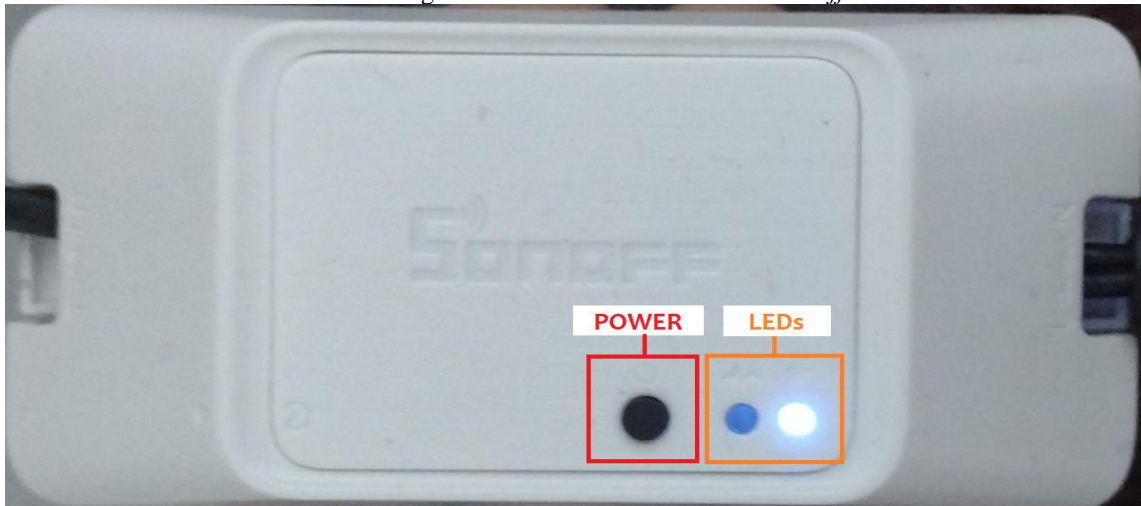
## 2.7 CONFIGURAÇÃO DE *SONOFF* PARA OPERAR EM MODO DIY

A seguir é descrito o processo de configuração da *Sonoff* para o modo DIY, este modo permite a utilização da *Sonoff* por meio de requisições HTTP em uma rede local, assim possibilita sua integração ao protótipo. A *Sonoff* é o controlador *on/off* responsável pelo acionamento dos dispositivos eletrônicos conectados, como citado por Santos e Zamberlan (2021) no corpo deste estudo.

Para isso, deve-se pressionar o botão *Power* por 5 segundos para entrar no modo de pareamento, o qual é usado no aplicativo padrão do dispositivo. Novamente, deve-se pressionar

por mais 5 segundos para entrar no modo DIY, onde o LED do dispositivo irá piscar frequentemente. O botão *Power* e os LEDs da *Sonoff* podem ser vistos a seguir (figura 9).

Figura 9 - Botão *Power* e LEDs da *Sonoff*



Fonte: Autores (2022).

Nesse momento o dispositivo gerou um ponto de acesso com o nome semelhante a “**ITEAD-XXXXXXX**”, onde por meio de um computador foi realizada a conexão como mostrado a seguir (figura 10).

Figura 10 - Rede local criada pela *Sonoff*



Fonte: Autores (2022).

Após a conexão, em um navegador *web*, foi acessado o seguinte endereço “<http://10.10.7.1>”, onde foi configurada a rede *wi-fi* que de fato seria utilizada no sistema (figura 11). Lembrando que o dispositivo aceita apenas redes com frequência de 2.4Ghz.

Figura 11 - Página de configuração de rede da *Sonoff*

The image shows a web browser window with a red header. The address bar displays '10.10.7.1' and the page is labeled 'Não seguro'. The main content area has a white background. The first section is titled 'Network name(SSID):' in bold black text, followed by a large, empty text input box with the placeholder text 'Network name'. The second section is titled 'Network password:' in bold black text, followed by another large, empty text input box with the placeholder text 'Network password'.

Fonte: Autores (2022).

Após a conexão de rede, a *Sonoff* já estava configurada em modo *DIY*, sendo necessário utilizar algum software para obter o endereço de IP do dispositivo ou acessar diretamente o painel do roteador.

Dessa forma, para sua utilização deve-se fazer uma requisição HTTP do tipo *POST* para o endereço “[http://\[endereçoIP\]:\[porta\]/zeroconf/switch](http://[endereçoIP]:[porta]/zeroconf/switch)”, informando o IP, porta e um objeto *JSON*<sup>12</sup> contendo a chave *switch* de valor *ON* ou *OFF* para ligar ou desligar o equipamento. A representação do *JSON* completo e resposta retornada pode ser vista na figura 12.

---

<sup>12</sup> Segundo o site DevMedia (2011), *JSON* (JavaScript Object Notation) é um formato para troca de informações que tem como características ser leve, rápido e de simples leitura, utilizado por diversos sistemas.

Figura 12 - Requisição e resposta para ligar/desligar *sonoff*

The screenshot displays a REST client interface for a POST request to `http://192.168.100.65:8081/zeroconf/switch`. The request body is a JSON object: `{ "deviceid": "", "data": { "switch": "on" } }`. The response is a `200 OK` status with a `seq` of 3 and `error` of 0. Red labels "REQUISIÇÃO" and "RESPOSTA" are overlaid on the request and response panes respectively.

```
POST http://192.168.100.65:8081/zeroconf/switch 200 OK 109 ms 19 B 3 Days Ago
```

```
JSON Auth Query Headers 1 Docs Preview Headers 4 Cookies Timeline
```

```
1 * {
2   "deviceid": "",
3   "data": {
4     "switch": "on"
5   }
6 }
```

```
1 * {
2   "seq": 3,
3   "error": 0
4 }
```

**REQUISIÇÃO** **RESPOSTA**

Fonte: Autores (2022).

Para a obtenção do estado atual de uma *Sonoff*, no momento em que se abre a interface da aplicação, deve realizar uma requisição HTTP do tipo *POST* para o endereço “`http://[endereçoIP]:[porta]/zeroconf/info`”, com um objeto *JSON* e a chave *data* vazia, como mostrado a seguir (figura 13).

Figura 13 - Requisição e resposta para obter informações de *Sonoff*

The screenshot displays a REST client interface for a POST request to `http://192.168.100.65:8081/zeroconf/info`. The request body is a JSON object: `{ "deviceid": "", "data": {} }`. The response is a `200 OK` status with a `seq` of 26 and `error` of 0. The response body contains detailed device information including `switch`, `startup`, `pulse`, `pulseWidth`, `ssid`, `otaUnlock`, `fwVersion`, `deviceid`, `bssid`, and `signalStrength`. Red labels "REQUISIÇÃO" and "RESPOSTA" are overlaid on the request and response panes respectively.

```
POST http://192.168.100.65:8081/zeroconf/info 200 OK 59.8 ms 237 B 4 Days Ago
```

```
JSON Auth Query Headers 1 Docs Preview Headers 4 Cookies Timeline
```

```
1 * {
2   "deviceid": "",
3   "data": {
4   }
5 }
```

```
1 * {
2   "seq": 26,
3   "error": 0,
4   "data": {
5     "switch": "on",
6     "startup": "off",
7     "pulse": "off",
8     "pulseWidth": 5000,
9     "ssid": "RAINERIO_JR_OI_FIBRA_2.4G",
10    "otaUnlock": false,
11    "fwVersion": "3.6.0",
12    "deviceid": "100086f50c",
13    "bssid": "20:ab:48:ab:ed:f0",
14    "signalStrength": -49
15  }
16 }
```

**REQUISIÇÃO** **RESPOSTA**

Fonte: Autores (2022).

Esta requisição retorna todas as informações de uma *Sonoff* como dados da rede, potência do sinal de *wi-fi*, número de versão e ID do dispositivo e principalmente o estado atual, indicando se o dispositivo se encontra ligado ou desligado.

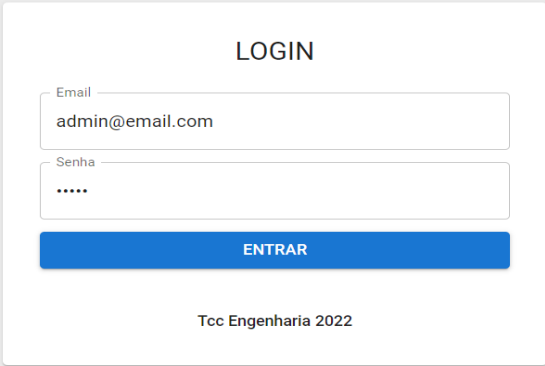
Estas requisições integram a API da *Sonoff* modelo BASIC R3 para sua utilização em modo *DIY*, que também permite realizar outras ações como indicar um intervalo de tempo em que um dispositivo deve ser desligado ou atualizar o *firmware* do dispositivo. Com essas requisições *POST*, pode-se perceber a aplicação real do conceito de API, bem como os métodos e parâmetros que compõem essa arquitetura, como exposto por AWS (2022).

## 2.8 DESENVOLVIMENTO DA INTERFACE DA APLICAÇÃO E INTEGRAÇÃO COM A API

A interface do protótipo conta com algumas telas que são essenciais à maioria dos sistemas, porém com foco na *dashboard* que contém a câmera e a *Sonoff* utilizada para testes de funcionamento do sistema.

Ao abrir a aplicação o usuário deverá realizar o *login* utilizando um e-mail e senha previamente cadastrado, como mostrado na figura 14.

Figura 14 - Tela de *login*



LOGIN

Email  
admin@email.com

Senha  
.....

ENTRAR

Tcc Engenharia 2022

Fonte: Autores (2022).

Após o *login*, devem ser cadastrados no sistema os dispositivos (câmeras e *sonoffs*) que serão utilizadas. Assim, o usuário deve ter acesso aos endereços de IP que foram configurados nos tópicos anteriores.

Para o cadastro de câmeras o usuário deve acessar o menu **Dispositivos** e clicar no botão **Novo dispositivo**, informando no cadastro o endereço de IP, o nome do dispositivo para sua identificação, além de dados de autenticação da própria câmera que são usuário e senha como pode ser visto na figura 15.

Figura 15 - Tela de cadastro de câmeras

Dispositivos

+ NOVO DISPOSITIVO

Cadastro de dispositivo

CÂMERAS SONOFF

Dados de exibição

Nome do dispositivo

Endereço de IP

Dados de autenticação de câmera

Usuário

Senha

CANCELAR CADASTRAR

ID	Endereço
1	192.168.2.100
2	192.168.2.101
3	192.168.2.102

Fonte: Autores (2022).

Já para o cadastro das *Sonoffs* o usuário deve acessar o mesmo menu para cadastro de câmera, porém acessando a guia *Sonoffs* na tela de cadastro mostrado na figura 16. Para inserir uma nova *Sonoff* é necessário informar apenas um nome para identificação e o endereço de IP referente a ela.

Figura 16 -Tela de cadastro de *Sonoffs*

Dispositivos

+ NOVO DISPOSITIVO

Cadastro de dispositivos

CÂMERAS SONOFF

Dados de exibição

Nome

Lâmpada protótipo

Endereço de IP

0.0.0.0

CANCELAR CADASTRAR

ID	Endereço IP	Ações
1	192.168.2.100	ON
2	192.168.2.101	ON
3	192.168.2.102	OFF

Fonte: Autores (2022).

Ao cadastrar os dispositivos eles serão listados separadamente em duas guias, para melhor visualização destes. A figura 17 a seguir, mostra a listagem de *Sonoffs* cadastradas contendo informações da mesma, além de uma coluna *Status* que contém o estado atual e controle de cada *Sonoff*.

Figura 17 - Listagem de *Sonoffs* cadastradas

ID	Endereço IP	Nome	Status	Ações
1	192.168.2.100	Lâmpada lab. prog.	ON	
2	192.168.2.101	Central lab. circ.	ON	
3	192.168.2.102	Tomadas lab. circ.	OFF	

Fonte:

Autores (2022).

O protótipo traz ainda a possibilidade de criar salas para melhor gerenciamento dos componentes cadastrados. Dessa forma, é possível organizar os dispositivos e relacioná-los criando uma sala contendo um conjunto de câmeras e *sonoffs*.

Neste caso, o usuário deverá acessar o menu **Salas** e criar uma nova sala informando o nome e selecionando as câmeras e *sonoffs* que compõem a sala como mostrado na figura 18.

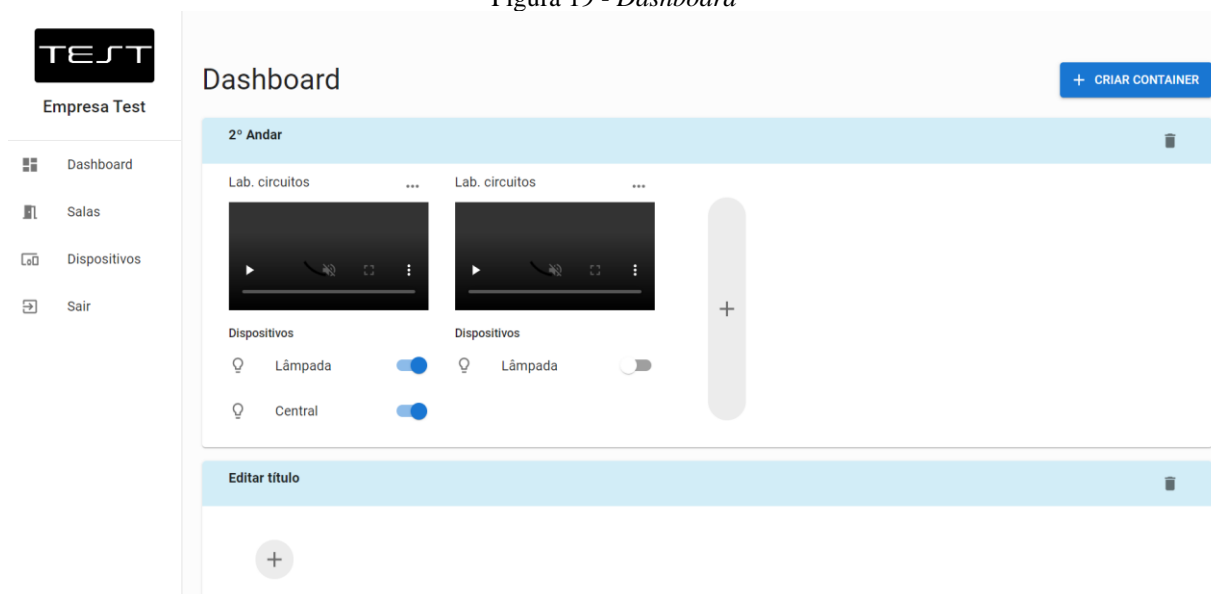
Figura 18 - Tela de cadastro de salas

ID	Nome	Sonoffs	Ações
1	Lab. Programação	2	
2	Lab. Circuitos	1	
3	Biblioteca	0	

Fonte: Autores (2022).

Com todas essas informações cadastradas, o usuário poderá personalizar sua *dashboard* criando containers e adicionando salas a eles, dessa forma, o usuário tem uma visão ampla e organizada de seus dispositivos e ambientes, possibilitando a checagem das salas, por meio das câmeras, e controle dos eletrônicos através dos botões de ação presentes abaixo de cada câmera, figura 19.

Figura 19 - Dashboard



Fonte: Autores (2022).

Com isso, chegou-se à etapa final do protótipo, onde pode-se demonstrar o conceito de *IOT* integrando sistemas virtuais com dispositivos físicos, corroborando o conceito apresentado por Santos e Sales (*apud* ATZORI; IERA; MORABITO, 2010).

## 2.9 PROTÓTIPO DE APLICAÇÃO WEB COM VISUALIZAÇÃO DE IMAGENS DE CÂMERA DE SEGURANÇA E CONTROLE DE DISPOSITIVO ELETRÔNICOS

Seguindo o objetivo geral do estudo que se trata da construção de um protótipo de uma aplicação *web*, onde será possível visualizar imagens de câmeras de salas e controlar dispositivos eletrônicos de maneira remota utilizando conceitos de *IOT*, é apresentado a seguir, o processo de montagem e funcionamento da versão final do protótipo.

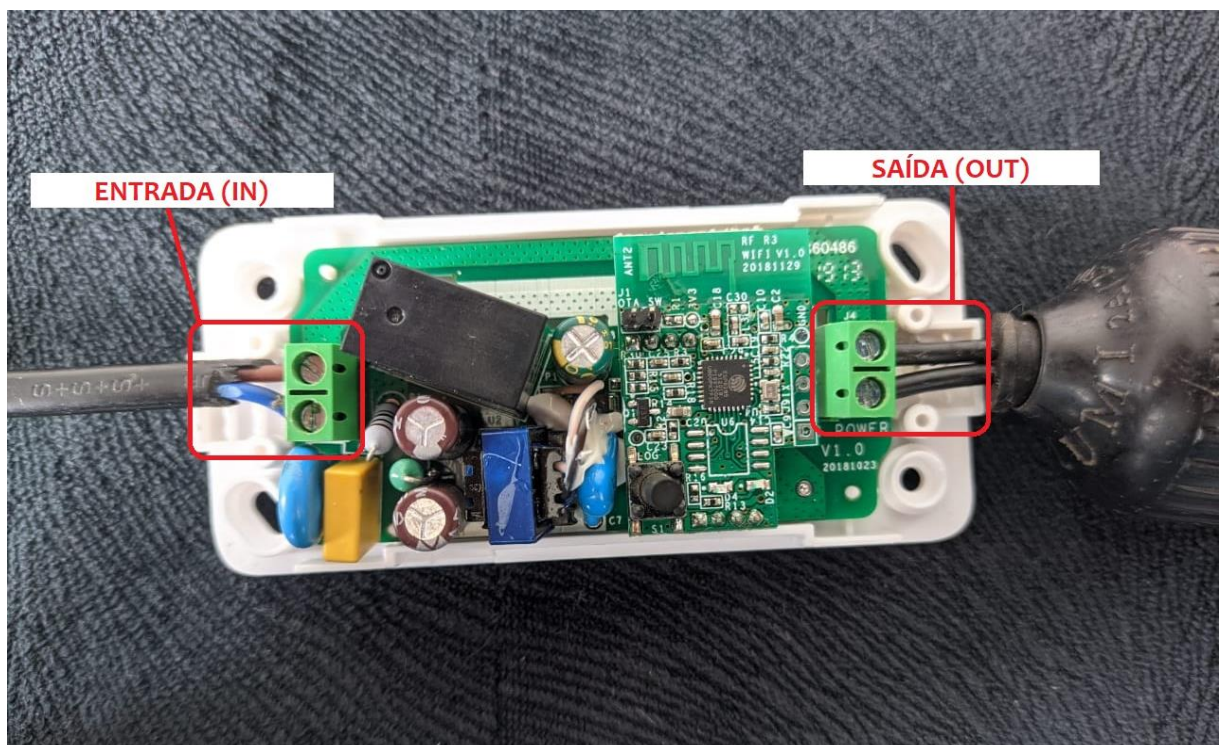
Como este protótipo utiliza apenas uma câmera e uma *Sonoff* do modelo *BASIC R3*, para fins de demonstração e foco na funcionalidade principal, estes já se encontram pré cadastrados no sistema para utilização.

No capítulo de materiais e métodos deste estudo foi apresentado a lista de materiais utilizados na construção deste protótipo. A seguir, será descrito o processo de montagem do

mesmo e demonstrado, por meio de imagens, seu funcionamento junto a todos os recursos utilizados. Para análise das etapas aqui descritas, deve-se levar em conta que haviam sido realizadas as configurações da câmera e *Sonoff*, conforme descritas nos tópicos anteriores.

Com isso, foi realizada a ligação de todos os materiais utilizados na parte elétrica do protótipo. Abaixo, temos a imagem (figura 20) da *Sonoff* aberta onde temos duas portas, uma para entrada (*in*) e outra para saída (*out*).

Figura 20 - Portas de ligação da *Sonoff*



Fonte: Autores (2022).

Nessas portas de entrada e saída, foram conectados os cabos utilizados na ligação elétrica. Na entrada foi conectado o cabo com o plugue de tomada que foi posteriormente utilizado na alimentação elétrica do dispositivo.

Na saída foram conectados os cabos que levam até o bocal da lâmpada, e realizada a conexão da lâmpada no mesmo. Dessa forma, encontra-se montado a parte elétrica do protótipo pronto para utilização dentro da aplicação, como mostrado na figura 21.

Figura 21 - Conexão dos dispositivos à Sonoff



Fonte: Autores (2022).

Para a câmera, levando em consideração a execução das configurações de IP e seu cadastro dentro da interface do protótipo, basta que ela seja conectada à energia elétrica para que passe a gerar imagens para o sistema. Na demonstração ela foi instalada em uma bancada, junto aos demais dispositivos para melhor visualização de seu funcionamento junto ao protótipo, apresentado na figura 22.

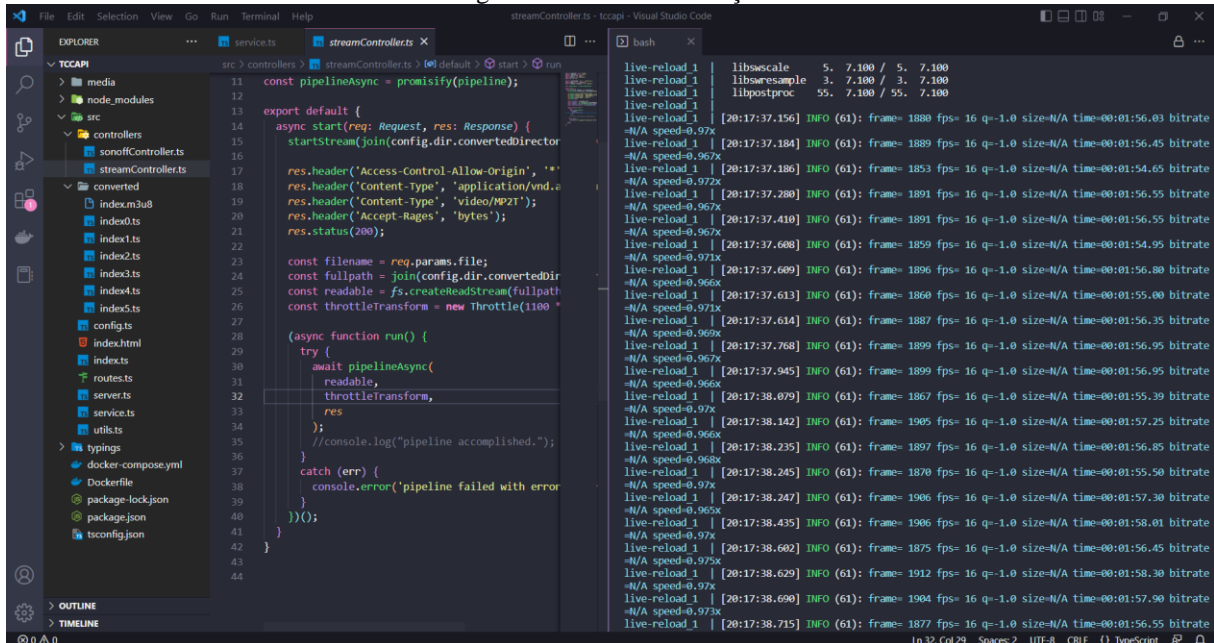
Figura 22 - Câmera instalada na bancada



Fonte: Autores (2022).

Com a câmera e a *Sonoff* devidamente instaladas e ligadas, foi inicializado, em um computador, a API do sistema para que fossem recebidas as imagens da câmera conforme a figura 23.

Figura 23 - API em execução



The image shows a Visual Studio Code window with a TypeScript file named `streamController.ts` and a terminal window. The code defines an API endpoint for video streaming. The terminal displays the output of the application, showing a series of log messages for each video frame received, including frame number, FPS, size, and bitrate.

```
const pipelineAsync = promisify(pipeline);

export default {
  async start(req: Request, res: Response) {
    startStream(join(config.dir.convertedDirector

    res.header('Access-Control-Allow-Origin', '*');
    res.header('Content-Type', 'application/vnd.a
    res.header('Content-Type', 'video/MP2T');
    res.header('Accept-Ranges', 'bytes');
    res.status(200);

    const filename = req.params.file;
    const fullpath = join(config.dir.convertedDir
    const readable = fs.createReadStream(fullpath
    const throttleTransform = new Throttle(1100 *

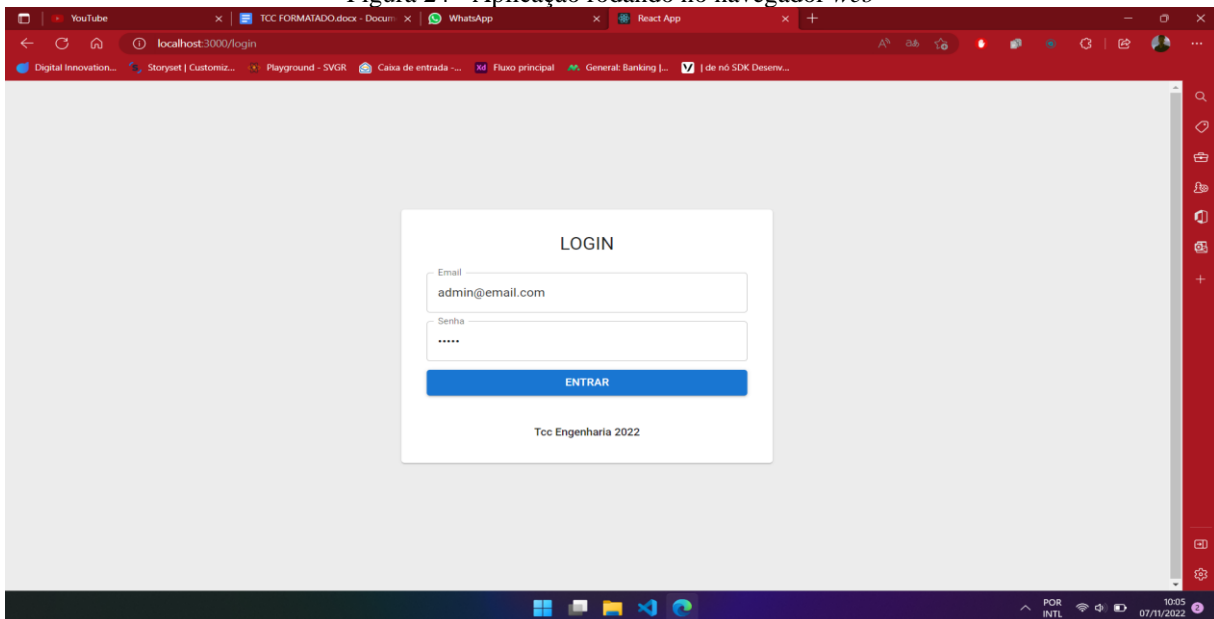
    (async function run() {
      try {
        await pipelineAsync(
          readable,
          throttleTransform,
          res
        );
        //console.log("pipeline accomplished.");
      } catch (err) {
        console.error("pipeline failed with error
      )());
    }
  }
}
```

```
live-reload_1 | libswscale 5. 7.100 / 5. 7.100
live-reload_1 | libswresample 3. 7.100 / 3. 7.100
live-reload_1 | libpostproc 55. 7.100 / 55. 7.100
live-reload_1 | [20:17:37.156] INFO (61): frame= 1880 fps= 16 q=-1.0 size=N/A time=00:01:56.03 bitrate
-N/A speed=0.97x
live-reload_1 | [20:17:37.184] INFO (61): frame= 1889 fps= 16 q=-1.0 size=N/A time=00:01:56.45 bitrate
-N/A speed=0.967x
live-reload_1 | [20:17:37.186] INFO (61): frame= 1853 fps= 16 q=-1.0 size=N/A time=00:01:54.65 bitrate
-N/A speed=0.972x
live-reload_1 | [20:17:37.280] INFO (61): frame= 1891 fps= 16 q=-1.0 size=N/A time=00:01:56.55 bitrate
-N/A speed=0.967x
live-reload_1 | [20:17:37.410] INFO (61): frame= 1891 fps= 16 q=-1.0 size=N/A time=00:01:56.55 bitrate
-N/A speed=0.967x
live-reload_1 | [20:17:37.608] INFO (61): frame= 1859 fps= 16 q=-1.0 size=N/A time=00:01:54.95 bitrate
-N/A speed=0.971x
live-reload_1 | [20:17:37.609] INFO (61): frame= 1896 fps= 16 q=-1.0 size=N/A time=00:01:56.80 bitrate
-N/A speed=0.966x
live-reload_1 | [20:17:37.613] INFO (61): frame= 1860 fps= 16 q=-1.0 size=N/A time=00:01:55.00 bitrate
-N/A speed=0.971x
live-reload_1 | [20:17:37.614] INFO (61): frame= 1887 fps= 16 q=-1.0 size=N/A time=00:01:56.35 bitrate
-N/A speed=0.969x
live-reload_1 | [20:17:37.768] INFO (61): frame= 1899 fps= 16 q=-1.0 size=N/A time=00:01:56.95 bitrate
-N/A speed=0.967x
live-reload_1 | [20:17:37.945] INFO (61): frame= 1899 fps= 16 q=-1.0 size=N/A time=00:01:56.95 bitrate
-N/A speed=0.966x
live-reload_1 | [20:17:38.079] INFO (61): frame= 1867 fps= 16 q=-1.0 size=N/A time=00:01:55.39 bitrate
-N/A speed=0.972x
live-reload_1 | [20:17:38.142] INFO (61): frame= 1905 fps= 16 q=-1.0 size=N/A time=00:01:57.25 bitrate
-N/A speed=0.966x
live-reload_1 | [20:17:38.235] INFO (61): frame= 1897 fps= 16 q=-1.0 size=N/A time=00:01:56.85 bitrate
-N/A speed=0.968x
live-reload_1 | [20:17:38.245] INFO (61): frame= 1870 fps= 16 q=-1.0 size=N/A time=00:01:55.50 bitrate
-N/A speed=0.97x
live-reload_1 | [20:17:38.247] INFO (61): frame= 1906 fps= 16 q=-1.0 size=N/A time=00:01:57.30 bitrate
-N/A speed=0.965x
live-reload_1 | [20:17:38.435] INFO (61): frame= 1906 fps= 16 q=-1.0 size=N/A time=00:01:58.01 bitrate
-N/A speed=0.97x
live-reload_1 | [20:17:38.602] INFO (61): frame= 1875 fps= 16 q=-1.0 size=N/A time=00:01:56.45 bitrate
-N/A speed=0.975x
live-reload_1 | [20:17:38.629] INFO (61): frame= 1912 fps= 16 q=-1.0 size=N/A time=00:01:58.30 bitrate
live-reload_1 | [20:17:38.690] INFO (61): frame= 1904 fps= 16 q=-1.0 size=N/A time=00:01:57.90 bitrate
live-reload_1 | [20:17:38.715] INFO (61): frame= 1877 fps= 16 q=-1.0 size=N/A time=00:01:56.55 bitrate
live-reload_1 | [20:17:38.715] INFO (61): frame= 1877 fps= 16 q=-1.0 size=N/A time=00:01:56.55 bitrate
```

Fonte: Autores (2022).

Em seguida, em um navegador *web*, foi executada a interface da aplicação no ambiente local no endereço “<http://localhost:3000/login>”, figura 24.

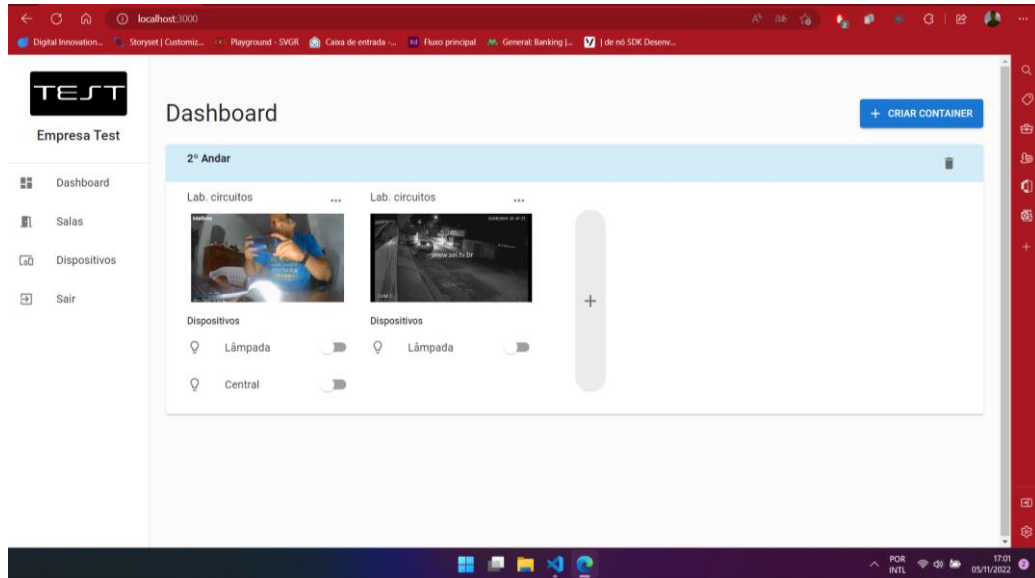
Figura 24 - Aplicação rodando no navegador *web*



Fonte: Autores (2022).

Após a realização do login, abre-se a *dashboard* contendo os dispositivos e ambientes cadastrados na aplicação. Nesta tela, é exibido à esquerda, a imagem da câmera utilizada no protótipo, e a outra imagem simula uma segunda câmera apenas para demonstração do ambiente, conforme a figura 25.

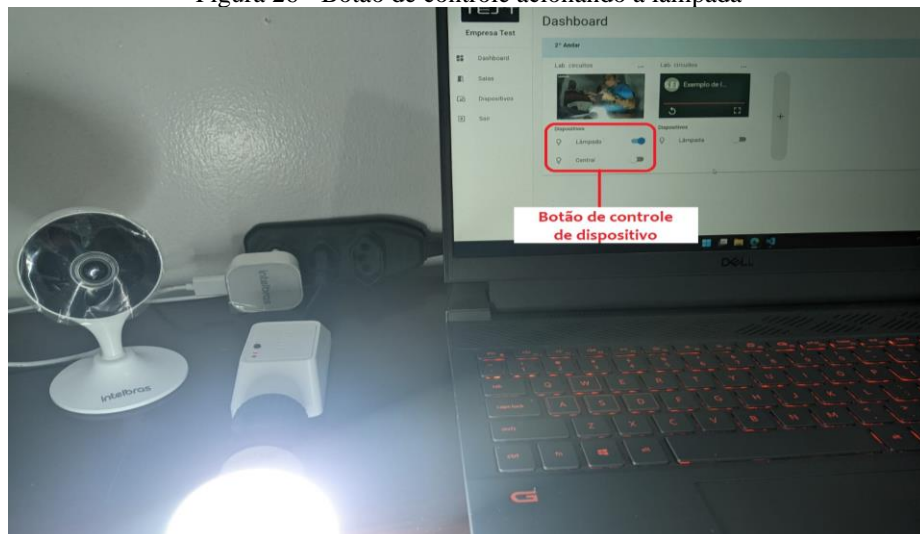
Figura 25 - *Dashboard* e exibição de câmeras



Fonte: Autores (2022).

É possível ver ainda os dispositivos presentes nesta sala, neste caso a lâmpada. Todos eles estão na cor cinza, indicando que estão desligados. Assim, ao clicar no botão ele ficará na cor azul, mostrando assim que o dispositivo se encontra ligado. A imagem abaixo (figura 26) retrata o momento em que o botão está ligado e conseqüentemente a lâmpada também está ligada.

Figura 26 - Botão de controle acionando a lâmpada



Fonte: Autores (2022).

Assim, temos a demonstração da montagem, configuração e funcionamento do protótipo, com ênfase em seus principais pontos que são geração de imagens e controle de dispositivos de maneira remota, permitindo a integração de diversos dispositivos e câmeras através de seus endereços de IP.

## CONSIDERAÇÕES FINAIS

Com base na problemática apresentada neste estudo, que se trata de problemas relacionados ao monitoramento de salas, e conseqüentemente, o desperdício de energia na Faculdade de Tecnologia do Amapá - META, teve-se como objetivo geral o desenvolvimento de um protótipo de aplicação *web* onde fosse possível visualizar imagens de câmeras das salas e controlar dispositivos eletrônicos de maneira remota utilizando conceitos de *IOT*. Para chegar a esse objetivo, buscou-se aprofundar os conhecimentos sobre conceitos de automação e aplicação de *IOT*, bem como, maneiras de exibir imagens de câmeras IP em navegadores *web*.

Com isto, pôde-se seguir na execução dos objetivos específicos, no qual o primeiro refere-se ao desenvolvimento de uma API para controle e gerenciamento de fluxo de dados de câmeras e regras de negócio da aplicação. Nesta etapa, foi elaborada a base da aplicação, responsável por integrar todos os demais recursos utilizados no protótipo. Teve-se como foco as funcionalidades de conversão de dados da câmera para geração de imagens na interface do protótipo e a comunicação da aplicação com a API da *Sonoff*.

O segundo objetivo específico teve como propósito a configuração da *Sonoff* para operar em modo *DIY*, para que fosse possível controlá-la pela aplicação. Assim, buscou-se entender o processo de configuração, montagem e operação do dispositivo nesse modo. Esta etapa incluiu testes de comunicação entre aplicação e *Sonoff*, testes de operação em diferentes redes e testes de controle de aparelhos eletrônicos.

Por último, foi realizado o desenvolvimento da interface da aplicação e integração com a API para utilização dos usuários. Para isso, foram explorados recursos para a construção de interfaces *web* utilizando a biblioteca ReactJS, bem como testes de comunicação entre cliente e servidor. Toda a comunicação entre interface e API, e os demais dispositivos, se dá por meio de requisições HTTP.

Conforme os objetivos apresentados, obteve-se os seguintes resultados. Foi possível realizar a geração de imagens no protótipo, simulando o ambiente das salas da instituição, com um tempo de resposta médio da câmera de 17,2 segundos no retorno das imagens para o cliente. Ressaltando que pode haver variações nesse valor a depender de variações de velocidade de internet.

Em relação a *Sonoff*, chegou-se ao objetivo de realizar sua integração ao sistema do protótipo, e controlar dispositivos eletrônicos remotamente. O tempo de resposta obtido ao ligar ou desligar um dispositivo acionando o botão presente na interface do protótipo, foi bastante satisfatório, tendo um retorno quase que instantâneo nos testes realizados. Outro ponto a se destacar é o fato de a *Sonoff* manter sua configuração de rede e de operação em caso de interrupções de energia elétrica. Assim, a *Sonoff* se mostrou um dispositivo prático para uso e com bom custo benefício para aplicações de *IOT*.

Todas as funcionalidades desenvolvidas, citadas anteriormente, foram centralizadas na interface *web* do protótipo, possibilitando ao usuário um monitoramento amplo das salas de sua instituição. Pois, a interface permite a criação de salas contendo todas as câmeras e dispositivos vinculados a ela, além de ambientes personalizados pelo usuário na *Dashboard*, para melhor organização e visualização dos mesmos.

Dessa forma, foi construído um modelo de protótipo capaz de acionar dispositivos eletrônicos de maneira remota e também permitindo a visualização de ambientes por meio de câmeras, ressaltando a sua integração com câmeras da marca Intelbras, a qual representa uma parcela grande do mercado atualmente no Brasil, diferenciando assim o protótipo de outras plataformas que não permitem a integração com essa marca de câmera.

Por se tratar de um estudo que teve como objetivo o desenvolvimento de um protótipo, este ainda requer melhorias para sua validação e aplicação em um ambiente real. Com isto, pode-se citar como possíveis melhorias a redução no tempo de retorno das imagens da câmera para o cliente, buscando reduzir o tempo de conversão de vídeo. Verificar questões de desempenho da aplicação ao utilizar múltiplas câmeras, uma vez que o protótipo utilizou apenas uma câmera. Examinar a possibilidade de acionar a *Sonoff* em modo *DIY*, em redes distintas, uma vez que este modo permite o controle apenas entre dispositivos conectados à rede local.

Dentre as dificuldades enfrentadas no processo de construção do protótipo, pode-se citar a dificuldade de obtenção de conteúdos referentes à realização de *streaming* de vídeo e formas de tratar e converter imagens para obter um melhor desempenho no processo. Este ponto chave da aplicação levou bastante tempo para chegar ao resultado apresentado e ainda requer melhorias para a operação.

Diante dos dados apresentados, este estudo mostrou-se promissor no âmbito da Internet das coisas, que é atualmente uma área muito explorada e com grandes desafios e descobertas todos os dias. Com isso, buscou-se contribuir com o desenvolvimento de uma solução a fim minimizar um problema real e que ocorre em diferentes ambientes.

O protótipo desenvolvido neste estudo, é o começo de um projeto que tende a crescer atingindo um patamar de implementação e possibilidade de chegar ao mercado, sendo uma aplicação que agrega tanto na logística de gerenciamento de ambientes, como na redução do desperdício de energia para as empresas.

## REFERÊNCIAS

ABESCO. **Desperdício de energia atinge R\$ 61,7 bi em três anos**. 2017. Disponível em: <<http://www.abesco.com.br/novidade/desperdicio-de-energia-atinge-r-617-bi-em-tres-anos/>>. Acesso em: 28 de abril de 2022.

ACCARDI, Adonis; DODONOV, Eugeni. **Automação Residencial: Elementos Básicos, Arquiteturas, Setores, Aplicações e Protocolos**. Revista TIS, v. 1, n. 2, p. 157, nov. 2012. Disponível em: <<http://revistatis.dc.ufscar.br/index.php/revista/article/view/27/30>>. Acesso em: 03 de maio de 2022.

AGUIAR, Matheus Fontinele de. **Desenvolvimento de um sistema de controle de periféricos via Web para ambiente residencial, utilizando Internet das Coisas (IoT) e o protocolo Message Queuing Telemetry Transport (MQTT)**. 2018. Universidade Estadual do Amazonas, Manaus, 2018.

ANDRADE. Maria Margarida de. **Introdução à Metodologia do Trabalho Científico**. 10. ed. São Paulo. 2010.

AWS. **O que é uma API? – Guia de APIs para iniciantes**. Amazon, 2022. Disponível em: <<https://aws.amazon.com/pt/what-is/api/#:~:text=API%20significa%20Application%20Programming%20Interface,de%20servi%C3%A7o%20entre%20duas%20aplica%C3%A7%C3%B5es.>>. Acesso em: 11 de Novembro de 2022.

BAYER, Fernando Mariano; ECKHARDT, Moacir; MACHADO, Renato. **Automação de Sistemas**. 4ª Ed. Santa Maria. 2011.

BRASIL. Ministério de Minas e Energia - MME. **Balanco Energético Nacional - BEN (Relatório Síntese 2021)**. 2021, EPE. Disponível em: <[https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-601/topico-588/BEN\\_S%C3%ADntese\\_2021\\_PT.pdf](https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-601/topico-588/BEN_S%C3%ADntese_2021_PT.pdf)>. Acesso em: 28 de abril de 2022.

CHASE, Otavio. **Sistemas Embarcados**. SBAJovem. 2007. Disponível em: <<http://www.lyfreitas.com.br/ant/pdf/Embarcados.pdf>>. Acesso em: 10 de maio de 2022.

DEVMEDIA. **O que é JSON**. DevMedia, 2011. Disponível em: <<https://www.devmedia.com.br/o-que-e-json/23166>>. Acesso em: 10 de outubro de 2022.

ELIAS, Glêdson; LOBATO, Luiz Carlos. **Arquitetura e protocolo de rede TCP-IP**. – 2. ed. Rio de Janeiro: RNP/ESR, 2013.

EWELINK. **Home - eWeLink**. eWeLink, 2022. Disponível em: <<https://ewelink.cc/>>. Acesso em: 02 de Novembro de 2022.

FFMPEG. **Sobre ffmpeg**. ffmpeg, 2022. Disponível em: <<https://ffmpeg.org/about.html>>. Acesso em: 03 de outubro de 2022.

IBM. **O que é Docker?**. Ibm, 2022. Disponível em: <<https://www.ibm.com/cloud/learn/docker>>. Acesso em: 30 de setembro de 2022.

KAUFFMANN, Boris. **Protocolos de Transmissão de Vídeo Sobre IP com Compressão**. AD Digital, 2020. Disponível em: <<https://setexperience2020.set.org.br/wp-content/uploads/2020/11/Protocolos-de-Transmissao-de-Video-sobre-IP.pdf>>. Acesso em: 15 de maio de 2022.

LIMA, Charles Borges; VILLAÇA, Marco V. M. **AVR e Arduino: Técnicas de Projeto**. 2ª ed. Florianópolis: Ed. dos autores, 2012.

LIMA, Mariana Ruivo Rabello de. **Sistema Embarcado para Detecção e Alerta de Atitudes de Desatenção ao Dirigir**. Universidade Presbiteriana Mackenzie, 2021. Disponível em: <<https://dspace.mackenzie.br/handle/10899/29107>>. Acesso em: 10 de maio de 2022.

LIMA, Thiago Fernandes Oliveira de. **Desenvolvimento de sistema de baixo custo para monitoramento e controle de processos de usinagem usando Raspberry Pi**. 2021. Disponível em: <<http://hdl.handle.net/11449/217107>>. Acesso em: 18 de maio de 2022.

MAGRANI, Eduardo. **A Internet das Coisas**. 1ª Edição. Rio de Janeiro: FVG Editora, 2018.  
MDN. **Guia JavaScript**. MDN Web Docs, 2020. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Introduction>>. Acesso em: 25 de maio de 2022.

OLIVEIRA, Aline. **Market Share: confira a definição e dicas para calcular**. mindminers, 2022. Disponível em: <<https://mindminers.com/blog/market-share-definicao-como-calcular/>>. Acesso em: 02 de Novembro de 2022.

OLIVEIRA, André de; ANDRADE, Fernando de. **Sistemas Embarcados - Hardware e Firmware na Prática**. 2. edo. Érica, 2010.

PEREIRA, Jorge. et al. **Uma Solução de IOT para Uso Eficiente de Energia Elétrica em Prédios Inteligentes**. Voltimum, 2017. Disponível em: <[https://www.voltimum.com.br/sites/www.voltimum.com.br/files/pdflibrary/01\\_st1-11.pdf](https://www.voltimum.com.br/sites/www.voltimum.com.br/files/pdflibrary/01_st1-11.pdf)>. Acesso em: 21 de abril de 2022.

PETERS, Shanan E.; MCCLENNEN, Michael. **The Paleobiology Database application programming interface**. Paleobiology, p. 1–7, 2015. Disponível em: <<https://www.cambridge.org/core/services/aop-cambridge-core/content/view/4D20F5CAFA1B0AC7033975418668D82B/S0094837315000391a.pdf/the-paleobiology-database-application-programming-interface.pdf>>. Acesso em 25 de maio de 2022.

REACT. **React – Uma biblioteca JavaScript para construir interfaces de usuário.** React JS. 2022. Disponível em: <<https://reactjs.org/>>. Acesso em 29 de setembro de 2022.

RÊGO, Leonardo Nunes Cornélio. **Desenvolvimento de um Protocolo Eficiente IOT com LORA Para Automação Predial.** Brasília. 2021. Disponível em: <[https://bdm.unb.br/bitstream/10483/30562/1/2020\\_LeonardoNunesCornelioRego\\_tcc.pdf](https://bdm.unb.br/bitstream/10483/30562/1/2020_LeonardoNunesCornelioRego_tcc.pdf)>. Acesso em: 11 de maio de 2022.

RIOS, Renan Osório. **Protocolos e serviços de redes: curso técnico em informática.** Colatina: CEAD / Ifes, 2011.

SANTOS, Aloisio Kneipp dos; Zamberlan Alexandre. **Sistemas pervasivos integrados por agentes inteligentes em JASON e Raspberry Pi.** 2021. Disponível em: <[https://tfgonline.lapinf.ufn.edu.br/media/midias/TFG\\_Aloisio\\_jpaOLeO.pdf](https://tfgonline.lapinf.ufn.edu.br/media/midias/TFG_Aloisio_jpaOLeO.pdf)>. Acesso em: 10 de maio de 2022.

SANTOS, Bruno P. et al. **Internet das Coisas: da Teoria à Prática,** 2016. Disponível em: <<https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf>>. Acesso em: 21 de abril de 2022.

SANTOS, Carlos Cesar; SALES, Jefferson David de Araújo. **O Desafio da Privacidade na Internet das Coisas.** Revista Gestão.Org, v.13, Edição Especial, 2015. p. 283. Disponível em: <<http://www.revista.ufpe.br/gestaoorg>>. Acesso em: 29 de abril de 2022.

SILVA, Matheus dos Santos Cunha; CARDOSO, Luciana Rocha. **Sistema de Gestão para Oficinas Mecânicas.** 2021. Disponível em: <<http://www.pensaracademico.unifacig.edu.br/index.php/repositorioctcc/article/view/3435/2481>>. Acesso em: 24 de maio de 2022.

SOARES, Henrique Ribeiro dos Santos. **CONDE: Um Sistema de Controle e Decisão para Edifícios Inteligentes usando Redes de Sensores e Atuadores Sem Fio.** Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2012. Disponível em: <<http://objdig.ufrj.br/15/teses/794399.pdf>>. Acesso em: 03 de maio de 2022.

SONEGO, Arildo Antônio. et al. **A Internet das Coisas aplicada ao conceito de eficiência energética: uma análise quantitativo-qualitativa do estado da arte da literatura.** Revista ATOZ, v.5, n. 2. 2016. Disponível em: <<https://revistas.ufpr.br/atoz/article/view/47860/29517>>. Acesso em: 28 de abril de 2022.

SONOFF. **Documentação do desenvolvedor SONOFF DIY.** SONOFF Official. 2022. Disponível em: <<https://sonoff.tech/diy-developer/>>. Acesso em 03 de outubro de 2022

SOUSA, Roberto Conhago Tavares. **Desenvolvimento de um sistema de controle de acesso e acionamento da iluminação e ar-condicionado das salas de aula, utilizando internet das coisas (IOT).** 2019. Disponível em: <<http://repositorioinstitucional.uea.edu.br/bitstream/riuea/2499/1/Desenvolvimento%20de%20um%20sistema%20de%20controle%20de%20acesso%20e%20acionamento%20da%20ilumina%C3%A7%C3%A3o%20e%20ar%20condicionado%20das%20salas%20de%20aula%20c%20utilizando%20internet%20das%20coisas%20IOT.pdf>>. Acesso em: 26 de abril de 2022

TECHTUDO. **IFTTT | Software | TechTudo**. TechTudo, 2022. Disponível em: <<https://www.techtudo.com.br/tudo-sobre/ifttt/>>. Acesso em: 02 de Novembro de 2022.

TYPESCRIPT. **TypeScript: JavaScript With Syntax For Types**. (typescriptlang.org). 2022. Disponível em: <<https://www.typescriptlang.org/>>. Acesso em 29 de setembro de 2022.

VIEIRA, Sérgio. **Sorria, a Intelbras está te filmando**. ISTO É DINHEIRO, 2021. Disponível em: <<https://www.istoedinheiro.com.br/sorria-a-intel-bras-esta-te-filmando/>>. Acesso em: 02 de Novembro de 2022.

VISUAL STUDIO CODE. **IntelliSense em Visual Studio Code**. Visual Studio Code. Disponível em <<https://code.visualstudio.com/docs/editor/intellisense>>. Acesso em: 29 de setembro de 2022.

WOWZA. **RTSP: O protocolo de streaming em tempo real explicado**. Wowza. Disponível em: <<https://www.wowza.com/blog/rtsp-the-real-time-streaming-protocol-explained>>. Acesso em: 03 de outubro de 2022.