UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ENGENHARIA
CÂMPUS DE ILHA SOLTEIRA

ANA BEATRIZ CATARIM TRIZÓLIO

MODÚLO DIDÁTICO IOT PARA AUTOMAÇÃO RESIDENCIAL BASEADO NO ESP8266 ESP-12

Ilha Solteira
2022

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Ilha Solteira

ANA BEATRIZ CATARIM TRIZÓLIO

MODÚLO DIDÁTICO IOT PARA AUTOMAÇÃO RESIDENCIAL BASEADO NO ESP8266 ESP-12

Trabalho de conclusão de curso apresentado à Faculdade de Engenharia de Ilha Solteira – Unesp como parte dos requisitos para obtenção do título de Engenheira Eletrecista.

Profa. Suely Cunha Amaro Mantovani
Orientadora

Ilha Solteira

2022

FICHA CATALOGRÁFICA
Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

T842m

Trizólio, Ana Beatriz Catarim.
Modúlo didático iot para automação residencial baseado no ESP8266 ESP-12 / Ana Beatriz Catarim Trizólio. -- Ilha Solteira: [s.n.], 2022
77 f. : il.

Trabalho de conclusão de curso (Graduação em Engenharia Elétrica) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira, 2022

Orientador: Cunha Amaro Mantovani
Inclui bibliografia

1. Internet das coisas. 2. Automação. 3. Monitoramento. 4. Sensores. 5. Atuadores.

Raiane da Silva Santos
Supervisora Técnica de Seção
Seção Técnica de Referência, Atendimento ao usuário e Documentação
Diretoria Técnica de Biblioteca e Documentação
CRB/8 - 9999

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Ilha Solteira

ATA DE DEFESA DE TRABALHO DE GRADUAÇÃO

Aos dez dias do mês de março do ano de dois mil e vinte e dois, a discente **ANA BEATRIZ CATARIM TRIZÓLIO,** matriculada sob o RA: 162054033, tendo como banca examinadora sua orientadora a Profa. Dra. *Suely Cunha Amaro Mantovani*, a Profa. Dra. *Christiane Marie Schweitzer* e o Engenheiro Eletricista *Henrique Zani Marinho*, apresentou o Trabalho de Graduação intitulado "**MÓDULO DIDÁTICO IOT PARA AUTOMAÇÃO RESIDENCIAL BASEADO NO ESP8266 ESP-12**", obtendo a nota **dez** e o conceito **Aprovado**.

Ana Beatriz Catarim Trizólio
- Discente -

Profa. Dra. Suely Cunha Amaro Mantovani
- Orientadora -

Profa. Dra. Christiane Marie Schweitzer
- Membro da banca -

Eng. Eletricista Henrique Zani Marinho
- Membro da banca -

AGRADECIMENTOS

Agradeço a toda a minha família que me apoiou durante toda a minha trajetória, seja emocianalmente ou financeiramente. Em especial, agradeço meus pais Adriana e João, minha irmã Ana Caroline e minha avó Zenaide, que se fizeram presentes mesmo quando eu estava distante ao ficarem inúmeras horas ao telefone comigo, me consolando e animando em dias tristes, acoselhando sobre os mais diversos problemas, me felicitando em minhas conquistas e acima de tudo me ouvindo sobre as bobagens do dia a dia, fazendo com que eu nunca me sentisse só. Vocês são a minha fortaleza!

Agradeço a todos os meus amigos por terem tornado Ilha Solteira meu lar durante a graduação, por todas as risadas e dificuldades que enfrentamos juntos. Em especial, agradeço minhas amigas Ana Julia Guedes, Giovanna Ferreira Cordeiro e Vitória Linares por serem minhas confidentes e apoiadoras, jamais esquecerei dos momentos que passamos juntas.

A Profª Suely Cunha Amaro Mantovani, por ter aceitado ser minha orientadora neste projeto e em anteriores, pelos ensinamentos e pela amizade que construímos ao longo do tempo.

Aos técnicos Everaldo Leandro de Moraes e Valdemir Chaves pela ajuda inestimável ao compatilharem seus conhecimentos sobre hardwares, circuitos e componentes eletrônicos e pela elaboração do protótipo de uma casa utilizado neste trabalho.

Agradeço a Deus por ter colocado todas as pessoas presentes em minha vida, sem elas eu não seria quem sou hoje. Pelos caminhos e obstáculos,que tornaram meus aprendizados e conquistas mais valiosos.

RESUMO

Trata-se este de um módulo didático em Internet das Coisas (IoT) aplicado na automação residencial. A automação residencial tem crescido muito nos últimos tempos devido, principalmente, às novas tecnologias e aos dispositivos exclusivos para IoT, auxiliando no controle e monitoramento de uma residência. Neste utiliza-se o módulo wifi Esp8266 NodeMCU Esp-12E que é uma plataforma de desenvolvimento com bons recursos de memória, entrada e saída de dados e baixo consumo de energia, entre outros e na integração com o ambiente são usados os sensores de temperatura, umidade, luminosidade e monóxido de carbono. Os dados dos sensores são visualizados em gráficos pelo sistema operacional denominado Home assistant que pode ser acessado por computador e/ou smartphone, quando devidamente configurado, e por meio dele controla o ambiente para acender ou desligar, lâmpadas, microventiladores, e um buzzer. Esses componentes são colocados em um protótipo de uma casa em acrílico que juntamente com o hardware montado, proporciona um módulo didático que pode servir para o uso dos alunos da graduação no aprendizado dos conceitos da Internet das coisas.

Palavras-chave: Internet das coisas; automação; monitoramento; sensores; atuadores.

ABSTRACT

This is a didactic module in Internet of Things (IoT) applied in home automation. Home automation has grown a lot in recent times, mainly due to new technologies and unique devices for IoT, helping to control and monitor a home. In this one, the Esp8266 NodeMCU Esp-12E wifi module is used, which is a development platform with good memory resources, data input and output and low power consumption. In the integration with the environment, sensors of temperature, humidity, luminosity and carbon monoxide are used. Sensor data are visualized in graphics by the operating system called Home assistant, which can be accessed by computer and/or smartphone, when properly configured, and through it control the environment to turn on or off lamps, micro fans, and a buzzer. These components are placed in a prototype of an acrylic house that, together with the assembled hardware, provides a didactic module that can be used by undergraduate students to learn the concepts of the Internet of Thing.

Keywords: Internet of things; automation; monitoring; sensors; actuators.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Internet das coisas, ou Internet of Things (IoT) se refere a interconexão digital de objetos cotidianos com a internet. Este nome, dado por Kevin Ashton (2009), é utilizado atualmente para referir-se a um mundo no qual tudo, a casa, os eletrodomésticos, o carro, o mobiliário urbano, as máquinas das fábricas, estão conectados à internet. Ainda que Ashton tenha comentado que a expressão era de uso corrente em pesquisas desde 1999, o termo se tornou frequente em 2009.

A evolução da Internet das Coisas tem causado um grande impacto no dia a dia das pessoas. É por meio do avanço desta tecnologia que é possível deixar a casa inteligente, com objetos e eletrônicos funcionais e autossuficientes. Ou seja, além dos *smartphones* e computadores pessoais, outros dispositivos poderão identificar padrões, processar informações e executar tarefas com apenas um (ou nenhum) clique, como por exemplo geladeiras inteligentes que emitem avisos de produtos que estão terminando e já realizam cotações de preços informando qual o melhor lugar para realizar as próximas compras (GODOI, 2019).

Antigamente a internet ligava apenas computadores a computadores de uso exclusivo militar e de algumas faculdades, enquanto hoje em dia, o acesso à internet foi ampliado. Essa mudança com a internet e tantos dispositivos conectados traz muitas outras características para o meio, revolucionando o modo em que vivemos facilitando e organizando tarefas do dia a dia (HELPDIGITAL, 2017).

Atualmente se encontram muitos equipamentos e soluções de automação residencial ou conectados e integrados à aplicativos, mas a tecnologia está convergindo para produtos, que qualquer pessoa pode instalar e dentro de alguns minutos começar a usar, necessitando de pouca infraestrutura para usufruir dos benefícios de uma casa inteligente. Basta apenas ter uma casa preparada para a IoT, com uma excelente conexão à internet, melhorando a qualidade e a segurança de vida das pessoas (DESTERRO ELETRICIDADE, 2018).

Considera-se que a quarta revolução industrial, também chamada de Indústria 4.0, é devido à Internet das Coisas. Nas “fábricas inteligentes" vários sistemas passaram a ser distribuídos, com processamento no local. Os sensores enviam, por meio da internet, os dados para um software remoto ou para a nuvem.

Este procedimento tem modernizado as empresas e a sua administração (FREITAS, 2018).

Uma importante aplicação da Internet das Coisas é a residencial, que conecta uma casa a qualquer lugar do mundo por meio de um *smartphone* ou *tablet*. Empresas oferecem os melhores sistemas de automação residencial *wireless*, com equipe de engenheiros que desenvolve projetos residenciais personalizados (DHT AUTOMAÇÃO, 2022)

Além do quesito luxo, a automação residencial também tem se tornado importante na questão da acessibilidade para pessoas com alguma deficiência, em suas atividades rotineiras, como por exemplo, abrir portas e janelas muitas vezes torna-se um obstáculo para cadeirantes, acender as luzes quando o interruptor está em pontos altos na parede; ou alguém com deficiência visual ao ouvir a campanhia ou interfone, precisa identificar quem chegou, encontrar a chave da entrada, dirigir-se até a porta e encaixar a chave na fechadura, entre outros. É de conhecimento que muitas pessoas com deficiência são independentes ou tem menos dependências, mas muitos destes obstáculos dentro de casa podem ser reduzidos com a automação.

Há várias aplicações de segurança na domótica ou automação residencial para controle de acesso a residência, por meio de dispositivos, sensores de abertura de janelas, portas e portões, câmeras de segurança e outros níveis de acesso que informam diretamente ao morador (e a polícia), possíveis problemas de invasões dentro da casa .

O sistema de automação residencial também pode proporcionar ao usuário, uma economia de recursos que pode significar uma diminuição nas despesas, especialmente contas de água e luz em uma residência, quando o controle e o monitoramento são utilizados juntos (LIMA; NOBRE; ALENCAR, 2016) .

Trabalhos encontrados na literatura, como o de Alexandre, Silva e Da Silva (2019) sobre uma aplicação de IoT, e o de Oliveira (2017), que descrevem o uso do Módulo ESP8266 NodeMcu ESP-12E para automação residencial, forneceram entre outros, os subsídios necessários para o desenvolvimento deste projeto de pesquisa .

Devido a todos estes fatores tem-se como proposta elaborar um módulo didático para controlar e supervisionar um ambiente (simulando uma residência) e para isso são utilizados sensores de temperatura, umidade, detector de gás

monóxido de carbono e luminosidade, utilizando a plataforma de desenvolvimento, Módulo ESP8266 NodeMcu ESP-12E e o sistema operacional Home Assistant.

1.2 Objetivos

Têm-se como objetivo neste projeto de pesquisa fazer um módulo IoT didático com aplicação residencial, por meio da utilização de alguns sensores, tais como, os que medem a temperatura e umidade, módulo DHT-11 luminosidade, LDR e para o monóxido de carbono, o MQ-7 e o acionamento de atuadores, como lâmpadas , ventiladores e um buzzer, usando para isso a internet sem fio e aplicativos. Por intermédio dos vários tipos de sensores pode-se proporcionar a uma residência, uma economia de energia, o conforto térmico e uma maior segurança. Utiliza-se na supervisão e controle, a plataforma de desenvolvimento, Módulo ESP8266 NodeMcu ESP-12E e o sistema operacional Home Assistant.

1.3 Organização do texto

O texto está organizado em sete capítulos. Além desta introdução, tem-se no capítulo 2, a revisão de literatura, apresentando alguns trabalhos que embasaram esta pesquisa.

No capítulo 3, faz-se uma breve introdução teórica de algumas tecnologias importantes para IoT que estão disponíveis atualmente no mercado de componentes e software, como o Módulo wifi NodeMCU ESP8266-12E, o protocolo de comunicação MQTT e as plataformas para a análise de dados via internet, em especial o Thing Speak e o sistema operacional Home Assistant..

São apresentadas no capítulo 4 os componentes eletrônicos para a integração com o ambiente, como os sensores e atuadores e os dispositivos condicionadores como o conversor analógico digital e relés, usados no desevolvimento do projeto.

Descreve-se no capítulo 5 o desenvolvimento do módulo didático IoT, com a descrição dos vários experimentos realizados ao longo desta pesquisa .

No capítulo 6 apresentam-se os resultados gráficos e de automação obtidos com a montagem do módulo IoT .

As conclusões encontram-se no capítulo 7, seguida das referências bibliográficas que proporcionaram os subsídios necessários para o desenvolvimento desta pesquisa, e os apêndices.

2 REVISÃO DE LITERATURA

Tem-se o conhecimento que a constante evolução das tecnologias na área de Internet das Coisas, impulsionam projetos e soluções, direcionados para a automação residencial visando não somente o conforto e a praticidade para o cotidiano das pessoas, como também proporcionando a economia de recursos como a água e a energia elétrica. No entanto, as tecnologias existentes no mercado para a criação de novos projetos em IoT oferecem padrões bem definidos, dificultando a integração entre as diversas soluções existentes.

Apresentam-se portanto, entre tantos trabalhos encontrados na literatura que abordam os conceitos e a aplicação em Internet das Coisas, algumas referências que nortearam esta pesquisa.

Em Silva (2019) é descrito o trabalho de hardware e software projetado para gerenciar a iluminação de um ambiente, como por exemplo, um quarto, com o objetivo de integrar ao sistema de controle o acender/apagar as lâmpadas, mas com a permanência do controle manual dos interruptores, compartilhando funções para controlar a iluminação de um ambiente residencial real. Neste foram utilizados o NodeMCU, MQTT e o Home Assistant. Testes funcionais foram desenvolvidos para simular um ambiente de infraestrutura de má qualidade e com muitas falhas visando a validação do protótipo e da proposta. Os resultados relatados sob condições de falhas, se mostraram estáveis e confiáveis para construir automações robustas.

É apresentado em Costa (2018), a proposta de criação de uma rede de diversos tipos de dispositivos que tenham a capacidade de comunicar com o mundo externo, permitindo o acesso por qualquer tipo de utilizador. Com este projeto, o autor espera padronizar os protocolos inerentes a IoT, integrar as diversas soluções de dispositivos, a simplificação da rede e com *hardware* de custo reduzido, de modo a ser incluida em qualquer ambiente doméstico. Em termos de possibilidades open-source, conclui que o Home Assistant é uma boa opção que torna-se amigável depois de configurado adequadamente, ficando mais intuitivo, facilitando a instalacao dos Add-ons oficiais. Com os bons resultados obtidos observa que a evolução dos protocolos aponta para uma solução sem fios, no futuro, e também espera uma evolução das baterias para a comunicação de dados, proporcionando melhores resultados na gestao de energia.

Em Solha (2018) é descrito a construção de três módulos IoT capazes de coletar dados de sensores e transmiti-los para um sistema de *dashboard* que realiza o controle do projeto. Neste foram utilizados o Raspberry Pi 2 modelo B, MQTT, NodeMCU, os sensores LDR, Reed Switch e DHT11, o PlatformIO (ambiente de desenvolvimento integrado) e o Home Assistant. Testes funcionais foram realizados para coletar as leituras dos sensores visando visualizar os dados na *dashboard* construída no Home Assistant. Os resultados do trabalho desenvolvido, relata o autor, não apresentam uma grande precisão em suas medições, porém, salienta que esse requisito não é necessário para os casos em que os módulos desenvolvidos se aplicam.

No trabalho de Avelar (2019) foi desenvolvido um sistema IoT modular e de baixo custo, visando monitoramento e controle de consumo de água em chuveiros elétricos residenciais por meio da integração de dispositivos e camadas de softwares. Foram utilizados o NodeMCU, Raspberry Pi, MQTT, sensor de fluxo e atuador e o Home Assistant e os testes realizados simulam o cadastro de usuários e o tempo de banho permitido , de forma que o fornecimento de água é interrompido sob certas circunstâncias e os dados são armazenados e visualizados pelo Home Assistant. Como resultado, tem-se que o sistema é eficaz no monitoramento remoto auxiliando o usuário na gestão da utilização de água, tornando-se um aliado na redução do consumo de água da residência.

Com a revisão destes trabalhos obeve-se os subsídios necessários para orientar às escolhas das tecnologias usadas, como a plataforma de desenvolvimento, o NodeMCU, os software de interfaceamento, MQTT e Home Assistant, além dos sensores e atuadores aplicado ao módulo IoT residencial proposto.

3 TECNOLOGIAS PARA A INTERNET DAS COISAS

Para o desenvolvimento desta pesquisa inicia-se por um levantamento bibliográfico visando fornecer um embasamento teórico sobre os dispositivos e ferramentas de software do universo da IoT, tais como plataformas de desenvolvimento e controle, de hardware e software, e protocolos de comunicação. Neste tópico faz-se uma breve descrição destas ferramentas visando aplicá-las no projeto.

3.1 Módulo wifi Esp8266 NodeMCU Esp-12E

O NodeMCU é uma plataforma open source da família ESP8266 criada em 2014, para ser utilizada no desenvolvimento de projetos IoT. É composta basicamente por um chip controlador, ESP8266 ESP-12E, uma porta micro USB para alimentação e programação, um conversor USB serial integrado e um wifi nativo, conforme apresentado na Figura 1. O ESP8266 NodeMcu ESP-12 é um módulo de comunicação sem fio com a internet, possui antena embutida e conector micro-usb para conexão ao computador e pode ser programado usando LUA (linguagem desenvolvida por brasileiros) ou o Ambiente de Desenvolvimento Integrado, do inglês, Integrated Development Environment (IDE) próprio do Arduino, utilizando a linguagem Wiring baseada em C/C++. Além disso, possui 11 pinos de I/O e conversor analógico-digital (BLOG ELETROGATE, 2018).

Figura 1 - Módulo NodeMCU ESP8266 ESP-12E

Fonte: Blog eletrogate, 2018.

Assim como em outras placas da família ESP8266, o NodeMCU é compatível com o ambiente de desenvolvimento do Arduino, sendo possível utilizar

suas bibliotecas. Outro diferencial do NodeMCU é a possibilidade de fazer a programação da placa via Over The Air (OTA), ou seja, por meio do wifi pode-se enviar os códigos para a placa (BLOG ELETROGATE, 2018.).

Apesar da pinagem reduzida, se comparado ao Arduino Mega 2560 R3, por exemplo, há no mercado diversos circuitos integrados que podem ser utilizados para expansão de entradas/saídas digitais e analógicas para usar com o NodeMCU. Porém, é importante destacar que alguns *shields* do Arduino e módulos não são compatíveis com esta plataforma de desenvolvimento.

O processador ESP8266 ESP-12E foi desenvolvido para realizar conexões wifi de forma eficaz, pois trabalha como um Ponto de Acesso (Acess Point) ou como uma Estação (Station), enviando e recebendo dados com baixo consumo de energia. Na Tabela 1 são apresentadas as suas principais características.

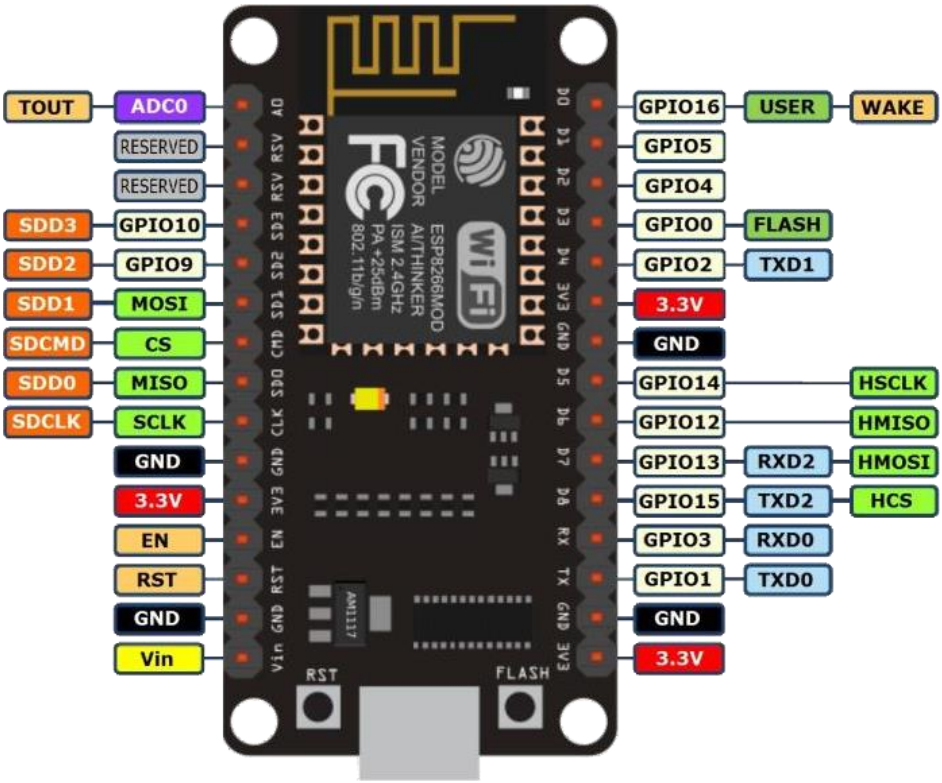
Tabela 1 - Principais características do ESP8266 ESP-12E

Processador	ESP8266 ESP- 12E
Nível lógico:	3.3V
Consumo de Corrente	15 mA output por GPIO pin
Memória Flash	4M Bytes
Velocidade do processador	80MHz / 160MHz
GPIOs	10 pinos de I/O (multiplexados com outras funções)
Conversor analógico digital (ADC)	1 entrada (resolução de 10bits)
Rede Wifi	Frequência de 2.4GHz
Comunicação por Wifi (802.11b/g/n);	Antena integrada em placa
Interface:	micro USB
Integra :	GPIO, PWM, IIC, 1-Wire
Saída TTL: conector 6 pinos:	RXD, TXD, VCC 3V3, VCC 5V, GND
Taxas de transmissão	300 bps a 1,5 Mbps
Dimensões (CxLxE) e Peso:	57x31x5mm (sem os pinos) e 10g

Fonte: Blog eletrogate, 2018.

Outro diferencial do NodeMCU é poder ser programado para interagir com um *broker* MQTT. A pinagem do módulo NodeMCU Esp8266 Esp-12E é apresentada na Figura 2 e na Tabela 2 as funções de cada pino.

Figura 2 –Pinagem do NodeMCU



Fonte: Blog eletrogate, 2018.

Tabela 2 - Funções dos pinos para o NodeMCU

Pino	Função
VIN	Pino de alimentação externa (recomendo 5,0V / 1A). Não use-o se estiver usando a USB.
GND	É o terra da placa.
RST	Reset do módulo ESP-12. Nível LOW(0V)
EN (Enable)	Ativa o módulo ESP-12 quando o nível for HIGH (3,3V).
3.3V	Saída do regulador interno 3,3V – Para alimentar outro dispositivo, máxima corrente 500 mA .
CLK	Interface SPI (clock) – pino SCLK (GPIO_6)
SD0	Interface SPI (master in serial out) – pino MISO (GPIO_7)
CMD	Interface SPI (chip select) – pino CS (GPIO_11)
SD1	Interface SPI (master out serial in) – pino MOSI
SD2	GPIO_9 usado também para comunicação com SD Card (SDD2)
SD3	GIPO_10 – usado também para comunicação com SD Card (SDD3)
RSV	Reservado (não use).
ADC0	Entrada ADC (10 bits). Tensão máxima de 1,1V
D0	GIPO_16 pode ser usado para acordar (WAKE UP) o ESP8266 em modo sono profundo (Deep sleep mode).
D1	GPIO_5 – entrada ou saída.

D2	GIPO_4 – entrada ou saída.
D3	GPIO_0 controla o upload do programa na Flash.
D4	GPIO_2 – UART_TXD1 quando carregando o programa na memória FLASH
D5	GPIO_14 usado em SPI de alta velocidade (HSPI-SCLK)
D6	GPIO_12 usado em SPI de alta velocidade (HSPI-MISO)
D7	GPIO_13 usado em SPI de alta velocidade (HSPI-MOSI) ou UART0_CTS.
D8	GPIO_15 usado em SPI de alta velocidade (HSPI-CS) ou UART0_RTS.
RX	GPIO_3 – U0RXD carregando o programa na memória FLASH.
TX	GIPO_1 – U0TXD carregando o programa na memória FLASH.

Fonte: Blog eletrogate, 2018.

3.2 MQTT

O MQTT (Message Queue Telemetry Transport) foi criado pela IBM em 1999 para comunicação M2M (Machine to Machine), trata-se de um protocolo de mensagens leve, ideal para aplicações IoT, que comumente tem recursos e capacidades limitados. Por exigir muito pouco em termos de processamento e banda (consumo de Internet), este é um dos protocolos ideais para dispositivos embarcados, também por esta razão, o MQTT é bastante usado para aplicações em IoT.

No final de 2014 o MQTT tornou-se oficialmente um padrão aberto da Organization for the Advancement of Structured Information Standards (OASIS), com suporte nas mais diversas linguagens de programação, usando implementações de software livre (YUAN, 2017).

Uma comunicação MQTT é composta de três partes: *publishers* (quem irá disponibilizar as informações), *subscribers* (quem irá receber as informações) e o *Broker* (servidor MQTT, na nuvem, acessível de qualquer lugar do mundo que contenha conexão com a Internet, responsável por receber e redirecionar todas as mensagens dos clientes-conecta aos clientes). Cliente (ou *client*) é qualquer dispositivo ou programa que é conectado pela rede e troca mensagens por meio do MQTT. Um cliente pode ser tanto um *publisher* e/ou um subscriber.

Teoricamente, não há limite especificado de *subscribers* e *publishers* em uma mesma comunicação MQTT, pois o limite nesse aspecto é do servidor em lidar com as conexões (KODALI, 2017).

Uma mensagem MQTT publicada / enviada possui duas partes importantes:

- **Tópico** – “chave” / identificação da informação publicada, usado para direcionar a informação publicada / enviada a quem assina (quem “dá subscribe”) no tópico;
- **Payload** – informação que se deseja enviar, propriamente dita.

O tópico é a forma como é gravado o interesse pelas mensagens recebidas ou como é especificado onde pretende publicar a mensagem. Os tópicos são representados com variavéis strings separadas por uma barra e cada barra indica um nível de hierarquia do tópico, no registro dos tópicos deve-se considerar que são *case sensitive*, isto é , importa se são escritas em letras maíscuslas ou minúsculas.

A comunicação MQTT acontece da seguinte forma:

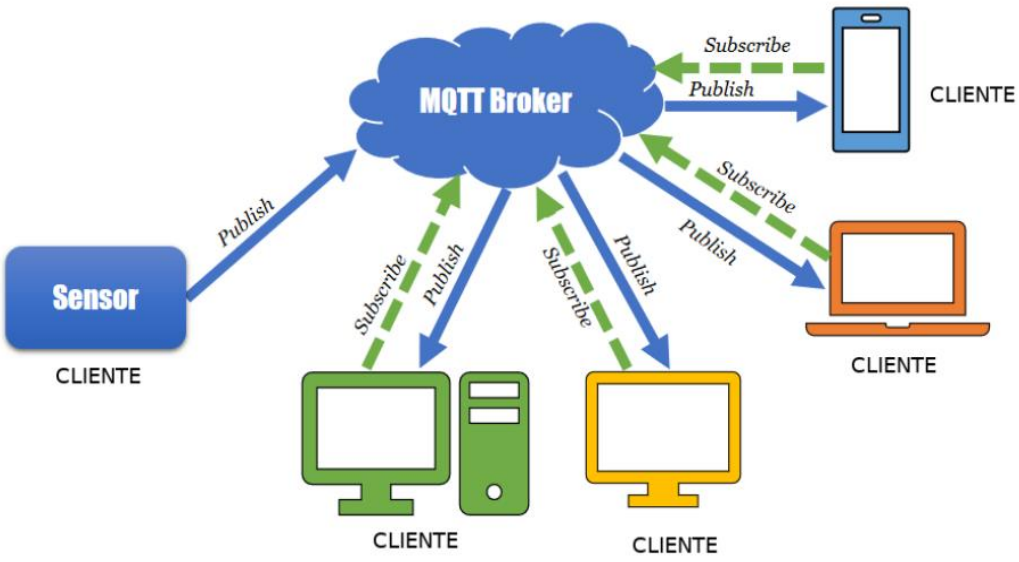
1. O cliente conecta-se ao *broker* por meio de uma conexão TCP/IP, assinando um "tópico" de mensagem;

2. O cliente publica as mensagens em um tópico no *broker*;

3. O *broker* encaminha a mensagem a todos os clientes que assinam esse tópico.

Um exemplo de arquitetura MQTT comum é apresentado na Figura 3. Na Figura o *broker* MQTT coleta e redirecionar os dados, as mensagens publicadas por clientes MQTT são transmitidas para outros clientes MQTT que se inscreverem ao tópico. Os publishers e subscribers são isolados, o que significa que eles não precisam conhecer a existência do outro. Outra informação importante é que um mesmo *client* MQTT pode ser subscriber e publisher de diversos tópicos. No caso de sensores existe apenas o publicador (KODALI, 2017).

Figura 3 - Exemplo de arquitetura MQTT



Fonte: Laboratory, 2018.

Uma solução em IoT que usa MQTT possui somente um servidor (*broker*), sendo todo o restante composto de *clients* MQTT.

3.2.1 NodeMCU e MQTT

O NodeMCU para interagir com um broker MQTT, necessita ser programado para ser um *client* MQTT. Antes de ser programado é necessário baixar e instalar a biblioteca de código aberto *PubSubClient*. Com essa biblioteca é possível trocar mensagens MQTT com um *broker*, de forma simplificada, facilitando a utilização do protocolo MQTT pelo NodeMCU. Depois disso, deve-se escolher um *broker* MQTT para utilizar entre os inúmeros disponíveis. Há brokers para uso, tanto públicos, sem custo, quanto privados (PUBSUBCLIENT, 2019).

3.3 Plataformas para a análise de dados via internet

A análise de dados por meio da internet é essencial para o monitoramento e controle de sistemas remotos. Por isso, dentre as várias plataformas para análise e visualização de dados disponíveis atualmente tem-se a Google Cloud IoT, dweet.io, The IoT Guru, e outras, dentre as quais destacam-se aThingSpeak e a Home Assistant. Devido à sua compatibilidade com o NodeMCU, fácil manuseio e a disponibilidade da versão gratuita, serão abordadas com mais detalhes neste tópico,

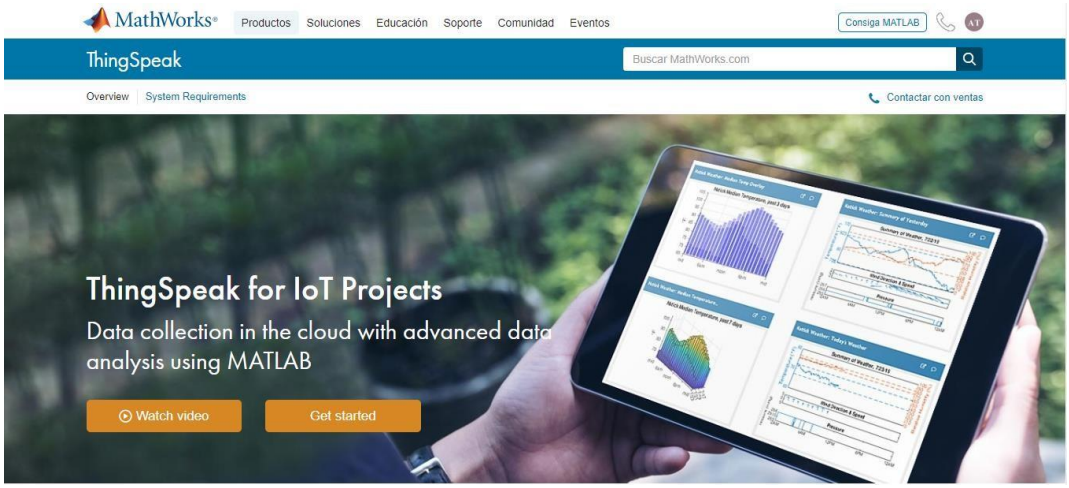
a ThingSpeak e a Home Assistant.

3.3.1 ThingSpeak

O ThingSpeak é uma plataforma IoT que permite o *upload* de dados numéricos que são armazenados na nuvem (em um computador no mundo), possuindo tanto a versão paga quanto a gratuita de uso. Esta plataforma pode ser usada por meio do website[1]ou do aplicativo disponível para Android e IOS, com isso é possível monitorar remotamente dados de qualquer lugar do mundo, desde que se tenha acesso a um computador ou smartphone com conexão a internet.

A plataforma Thingspeak tem como grande vantagem permitir a visualização dos dados enviados pelos dispositivos, em tempo real, mas também a possibilidade de análise, recorrendo ao software Matlab, visto que esta plataforma pertence ao MathWorks, conforme divulgação no site do ThingSpeak (2019). Tem-se na Figura 4 a tela do aplicativo ThingSpeaker.

Figura 4 - ThingSpeak (com Matlab)



Fonte: ThingSpeak , 2019

Para enviar dados ao ThingSpeak faz-se uma requisição HTTP (HyperText Transfer Protocol) ao servidor do ThingSpeak. Uma requisição HTTP constitui-se em uma string que contém as informações que é enviada via socket TCP client (Transmission Control Protocol) a um cliente (quem deseja fazer a requisição), a um socket TCP server (servidor que receberá a requisição, no caso o servidor do

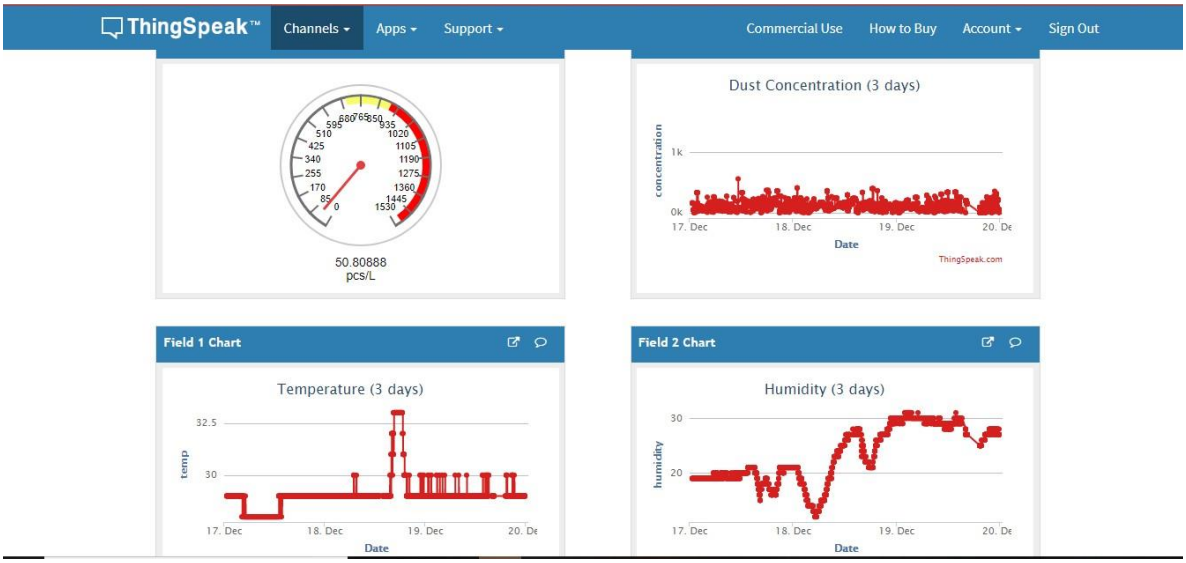
[1]https://thingspeak.com

ThingSpeak) usando a porta 80.

Outra informação importante é que na requisição HTTP constará uma chave de escrita para o canal ThingSpeak. Trata-se de uma string (única para cada canal), necessária para o ThingSpeak “saber” para qual canal direcionar os dados enviados.

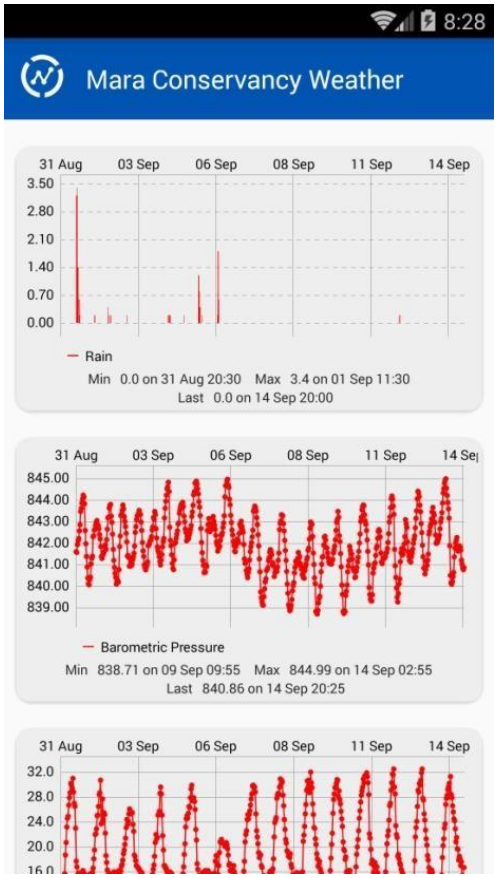
O ThingSpeak possui algumas limitações na versão gratuita, como quantidade de canais e *widgets* à sua disposição, e o tempo entre *upload* de dados que deve ser de, no mínimo, 15 segundos (refreshrate), pois os dados enviados fora deste intervalo de tempo, serão ignorados ou não registrados. Vale ressaltar que o ThingSpeak está em constante evolução, portanto novos recursos podem ser adicionados a qualquer momento. Na Figura 5 tem-se uma ilustração do *website* ThingSpeak utilizado no navegador e na Figura 6 o uso do aplicativo para smartphone.

Figura 5 - Website ThingSpeak



Fonte: Elaborado pelo autor.

Figura 6 - Aplicativo ThingSpeak para smartphone



Fonte: Elaborado pelo autor.

3.3.2 Home Assistant

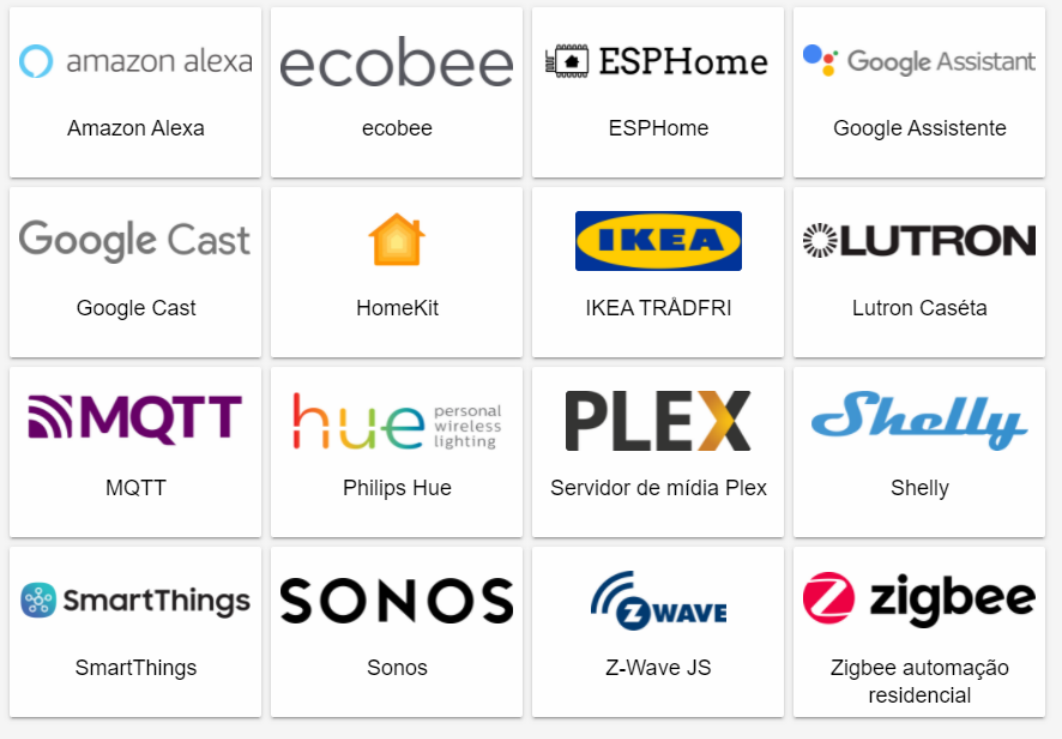
O Home Assistant também é uma alternativa gratuita e open source utilizada para desenvolver sistemas de automação residencial descentralizados que interpretam dados coletados por sensores, e além disso, controlam dispositivos atuadores, implementam regras de automação e gerenciam a comunicação entre dispositivos. O Home Assistant pode ser executado em computadores Linux, diretamente em SBCs (Single Board Computers) com o sistema operacional dedicado Hass.io ou em qualquer outro ambiente que tenha Python instalado (SILVA, 2019).

O Hass.io é um sistema operacional customizado e configurado para executar o Home Assistant, e foi desenvolvido para facilitar a instalação e utilização da plataforma. O Hass.io pode ser instalado e executado em uma máquina virtual e/ou um Raspberry Pi 3.

O Home Assistant não é apenas um aplicativo para controle e automação

residencial, é um sistema embarcado que oferece uma ampla integração com sensores e atuadores do mercado, como por exemplo, a assistente Alexa da Amazon, capaz de interagir com outros componentes instalados no Home Assistant por meio de comandos de voz. A loja virtual de *add-ons*[2] do Home Assistant é muito vasta e permite adicionar componentes de terceiros pelo Github além dos *add-ons* oficiais que adicionam várias funcionalidade ao sistema, por exemplo: MQTT Home Broker e servidor DNS (Domain Names System). Na Figura 7, apresentam-se alguns elementos de integração suportados pelo Home assistant.

Figura 7 - Integrações Home Assistant



Fonte: Assistant, 2019.

As integrações com o sistema principal da plataforma são chamadas de componentes, os quais são facilmente configurados usando um arquivo principal de configuração chamado *configuration.yaml* que possui o formato YAML. Os fundamentos da sintaxe YAML são coleções de blocos e mapeamentos contendo pares de valores-chave. O recuo é uma parte importante da especificação de relacionamentos e hierarquias dentro do YAML, além disso o Home Assistant é *case*

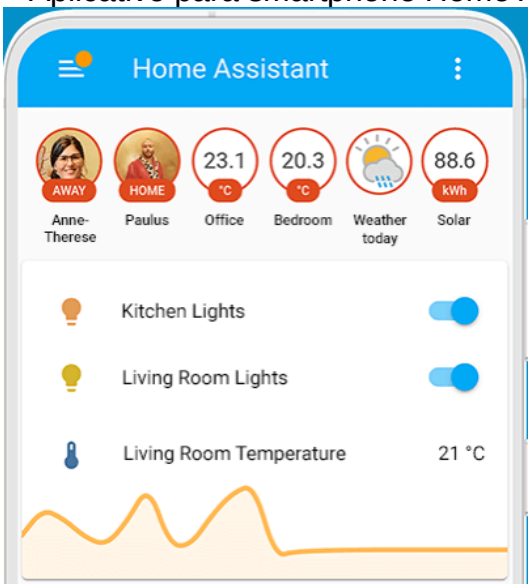
[2] Componentes de software que adicionam novas funcionalidades ou características ao Home Assistant.

sensitive, isto é, ele diferencia maiúsculas de minúsculas.

Assim como no ThingSpeak, é possivel monitorar a leitura de sensores em tempo real, tanto em servidor web como em um smartphone, como ilustrado na Figura 8. O grande diferencial do Home Assistant encontra-se no acionamento de atuadores via interface web manualmente, por botões implementados em um *dashboard* ou por acionamentos automáticos programados por meio de um editor de automações. No editor de automações é necessário definir quatro parâmetros para configurar uma automação, são eles: o nome, os gatilhos, as condições e as ações. Também é possível programar intervalos de acionamento, isto é, a automação pode ocorrer dentro de horários e dias da semana pré programados, por exemplo, horário comercial. Visto a necessidade de acionamentos remotos prevista no projeto, escolheu-se trabalhar com o Home Assistant .

Devido a utilização do protocolo MQTT durante o desenvolvimento do projeto, um componente que recebe o mesmo nome do protocolo é utilizado para que o sistema possa assinar tópicos e se torne capaz de receber os dados transmitidos através dos módulos IoT, assim eles podem ser exibidos na interface do usuário.

Figura 8 - Aplicativo para smartphone Home Assistant



Fonte: Assistant, 2019.

4 COMPONENTES ELETRÔNICOS

Neste tópico são apresentados os conceitos básicos sobre os componentes eletrônicos e as características de alguns que são usados no projeto visando fornecer os subsídios necessários para o entendimento do trabalho e que adicionados ao NodeMCU compõem o módulo IoT. Dentre eses, encontram-se os sensores, conversor analógico digital, regulador de tensão, relés e motor CC.

4.1 Sensores

Sensores são dispositivos sensíveis à alguma fonte de energia do ambiente que pode ser luminosa, térmica, cinética, relacionando informações sobre uma grandeza física que precisa ser mensurada (medida), como temperatura, pressão, velocidade, corrente, aceleração, posição, etc. Basicamente o sensor tem a função de detectar e responder com eficiência a algum estímulo do meio em que se encontra.

Existem vários tipos de sensores que respondem à estímulos diferentes como por exemplo, calor, pressão, movimento, luz e outros. Depois que o sensor recebe o estímulo, a sua função é emitir um sinal, geralmente elétrico, que seja capaz de ser convertido e interpretado por outros dispositivos.

Dentre os sensores de interesse deste projeto estão o LDR , sensor de gás MQ-7, sensor de temperatura e umidade relativa, DHT-11.

4.1.1 Sensor de Luminosidade - LDR

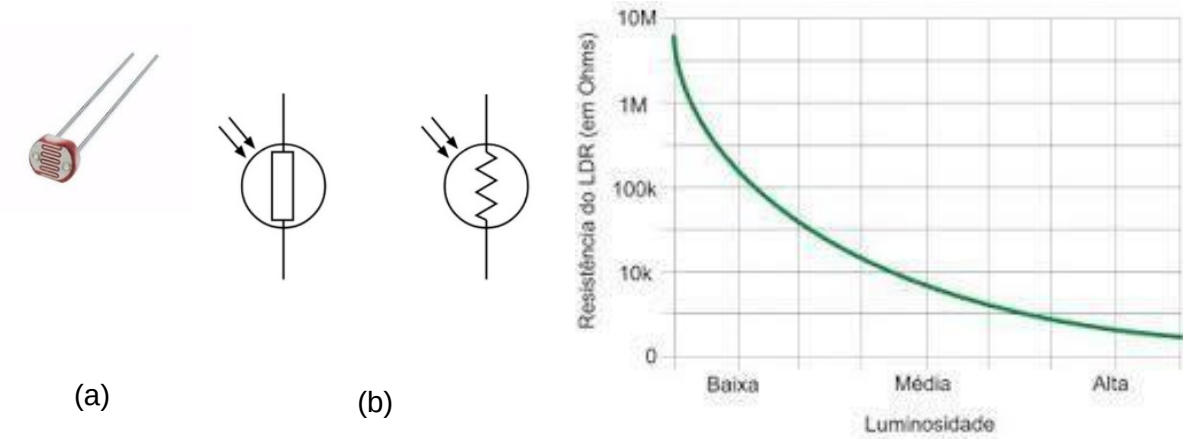
O LDR (Light Dependent Resistor) é um resistor dependente de luz ou seja sua resistência varia com a quantidade de luz , quanto maior a luz incidente nesse componente, menor será sua resistência.

São fabricados com materiais de alta resistência, como, por exemplo, o Sulfeto de Cádmio (CdS) ou o Sulfeto de Chumbo (PbS). Esses materiais possuem poucos elétrons livres quando colocados em ambiente escuro, e liberam elétrons quando há incidência de luz sobre eles, aumentando sua condutividade, efeito chamado de Fotocondutividade.

Dessa forma, o LDR pode assumir resistências na ordem de Mega Ohm no escuro e resistência na ordem de poucas centenas quando exposto a luz. Tem-se na

Figura 9 (a) um exemplar do LDR, em (b) o símbolos elétrico do sensor LDR e em (c) a curva de resposta tensão x resistência. Neste projeto utiliza-se o sensor LDR para detectar a quantidade de luz que incide no ambiente, e poder acender/apagar uma lâmpada remotamente(GTA UFRJ,2021)

Figura 9 - a) Sensor LDR. b) Símbolos elétricos. c) Curva de resposta tensão x resistência.

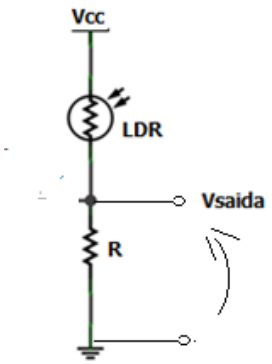


(a) (b) (c)

Um LDR em série com um resistor, *R*, conforme apresentado na Figura 10, compõem o circuito de medida alimentado por uma tensão VCC, sendo a saída, V_{saida} , dada por um divisor de tensão , Equação (1),

$$V_{saida} = \frac{R}{R_{LDR} + R} * V_{cc} \quad (1)$$

Figura 10 - Circuito de medida para o LDR



Fonte: Elaborado pelo autor

Para determinar o valor do resistor, R, em série com o LDR devem ser feitos alguns testes em laboratório, com diferentes valores de resistores comerciais, variando a intensidade luminosa. Com base nesses testes pode ser traçado o gráfico da Figura 9 (c), onde se observa a intensidade luminosa versus resistência do LDR em Ω(ohms). A característica analógica do sinal de saída do LDR , caso não possa ser tratada pelo NodeMCU pela falta de entradas analógicas, usa-se um conversor análogico digital (Analog Digital Conversor -.ADC) .

4.1.2 Sensor de Gás MQ-7

O sensor de Gás MQ-7 faz parte da família de sensores de gás da série MQ que tem sinal de saída do tipo analógica, mas de fácil integração com microcontroladores. A série MQ utiliza um pequeno aquecedor interno em conjunto a um sensor eletroquímico. Estes sensores são sensíveis a uma variedade de gases, conforme apresentado na Tabela 5, no Apêndice A e a escolha do mais adequado dependerá da aplicação do projeto. Apresentam funcionamento similar, embora a sensibilidade para os diversos sensores da série MQ sejam distintas.

O sensor MQ-7, mostrado na Figura 11, possui alta sensibilidade para gás monóxido de carbono com precisão de 10-10000ppm, opera entre -10 e 50ºC e consome menos de 150mA para alimentação de 5VCC. Apresenta baixo custo, adequado para diferentes aplicações tais como, projetos de segurança e automação residencial, visto que este gás é altamente tóxico para o ser humano.

Quando o sensor de gás MQ-7 detecta a presença ou mesmo a concentração do gás Monóxido de Carbono (CO) ele envia os sinais para um processador que pode ser programado para alertar emitindo sinais sonoros, ou luminosos, ou realizar tarefas, como liberar trancas de portas, etc. Este sensor de gás não foi desenvolvido para aplicações que envolvam segurança humana ou patrimonial profissional, é destinado apenas a propósitos experimentais (CANDIDO, 2017).

Figura 11 - Módulo com o sensor MQ-7 (a) Vista de cima. (b) Verso

(a) (b)

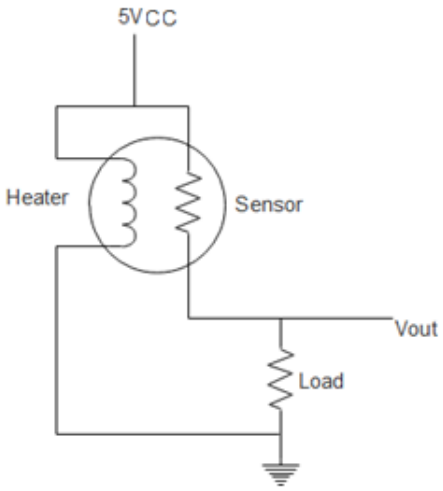
Fonte: Elaborado pelo autor

O trimpot na Figura 11(b) serve para calibrar o ponto de acionamento do sensor. Quando a concentração de Monóxido de Carbono fica acima do nível ajustado pelo potenciômetro, a saída digital *D0UT* fica em estado alto, caso fique abaixo do nível, fica em estado baixo. Para ter uma resolução melhor e medir a variação da concentração do gás no ar, usa-se a saída analógica *AOUT* conectada a um conversor ADC, como a do Arduino ou de outra placa.

O módulo MQ-7 possui um pino de saída digital que pode ser utilizado para o acionamento direto de um relé ou sirene, por exemplo, além de servir como um pino de indicação para que um microcontrolador interprete e produza uma determinada ação; um pino de saída analógica que faz a leitura, em tempo real, de monóxido de carbono presente no ambiente; um pino de alimentação e um pino de GND. Na Figura 12 tem-se o esquema elétrico para um sensor MQ genérico alimentado em 5 VCC.

Em condições normais, a resistência do sensor é alta e portanto a queda de tensão na carga é baixa e constante. Se o sensor detectar algum gás que tenha sensibilidade, a resistência do sensor irá cair, ou seja, mais corrente irá fluir no resistor de carga, fazendo com que a tensão aumente. Esta tensão de saída aumenta com o aumento da concentração de gás no ar. Este sensor necessita de aquecimento, logo, é normal um aquecimento e emissão de um cheiro inicial.

Figura 12 - Esquema elétrico para o sensor MQ



Fonte: Candido, 2017.

Neste projeto utiliza-se o sensor MQ-7 para monitorar a presença de monóxido de carbono no ambiente, caso seja detectado um valor elevado de gás, aciona um buzzer, indicando um vazamento de gás. Na Tabela 3 têm-se o resumo das especificações e pinagem do sensor MQ-7 (CANDIDO, 2017).

Tabela 3 - Especificações e pinagem do MQ-7

Especificações	
Detecção do gás:	Monóxido de Carbono
Concentração de detecção:	10-10.000ppm
Tensão de operação:	3-5V
Resistência de aquecimento:	$31\Omega \pm 3\Omega$
Tensão de aquecimento:	$5V \pm 0{,}2V$
Potência de aquecimento:	$\leq 350mW$
Sensibilidade:	Ajustável via potenciômetro
Saída:	Digital e Analógica
Comparador:	LM393
Dimensões:	32 x 20 x 15mm
Pinagem	
VCC:	5V
GND:	GND
DOUT:	Saída Digital
AOUT:	Saída Analógica

Fonte: Filipeflop. Sensor de Gás MQ-7, mónoxido de carbono, 2019

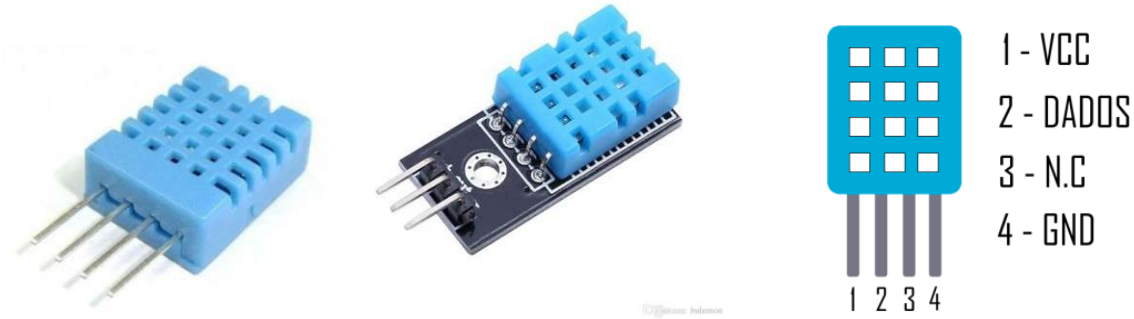
4.1.3 Sensor DHT-11

O dispositivo DHT-11 é um sensor de umidade e temperatura de baixo custo, que utiliza um termistor do tipo NTC para medir a temperatura entre 0 a 50º Celsius com uma precisão de 2 graus e o sensor de umidade é do tipo capacitivo que mede a umidade do ambiente entre 20 a 90 %, com uma precisão de 5%.

O módulo DHT-11 possuí um controlador de 8 bits que converte o sinal de temperatura e umidade dos sensores em um sinal serial pelo pino *Data*, isto é, o DHT11 usa uma comunicação simplificada de barramento único. A troca de dados ocorre por meio de um dreno aberto ou porta tri-state conectada à linha de dados. A utilização do sensor é facilitada por uma bilioteca que deve ser carregada na IDE do Arduino, nomeada “DHT.h” (Filipeflop. Monitorando Temperatura e Umidade com o sensor DHT11, 2013).

Apresenta-se na Figura 13(a) o sensor, e em 13 (b), o módulo, finalmente na Figura 13(c), a pinagem. Na Tabela 4, encontra-se o resumo das características técnicas deste sensor.

Figura 13 - (a) Sensor DHT-11. (b) Módulo DHT-11. (c) Pinagem do DHT-11



(a) (b) (c)

Fonte: Elaborado pelo autor

Tabela 4 - Características técnicas do DHT-11

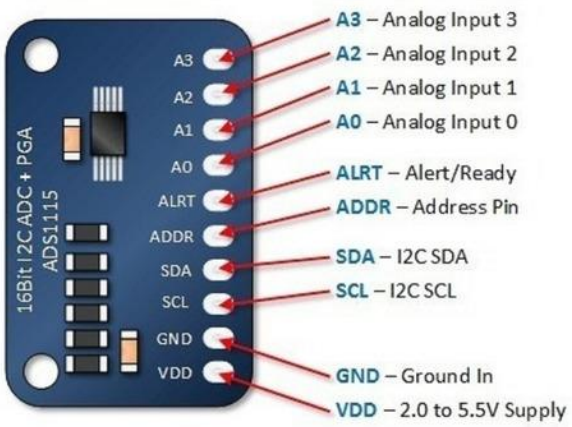
Tensão de operação	3,5 a 5,5VCC
Corrente de operação	0,3mA
Corrente de operação (em stand by)	60µA
Resolução	16 bits
Faixa de medição (umidade)	20 a 90%
Faixa de medição (temperatura)	0º a 50ºC
Precisão (umidade)	±5%
Precisão (temperatura)	±2º C
Tempo de resposta	2s

Fonte:Filipeflop. Monitorando Temperatura e Umidade com o sensor DHT11, 2013.

4.2 Conversor Analógico Digital ADS1115

O Conversor Analógico Digital ou Analog Digital Conversor (ADC) ADS1115 mostrado na Figura 14, tem resolução de 16 bits que melhora a precisão da medição e a resolução de um microcontrolador. Possui quatro canais de entrada que podem ser configurados nos modos *Medições de Terminação Simples*, *Diferencial* ou *Comparadora*. Ideal para projeto que seja necessário expandir as portas analógicas, como é o caso do NodeMCU que disponibiliza apenas um pino para entrada analógica com resolução de 10 bits .

Figura 14 - Pinagem para o ADS1115



Fonte: Hackster.io, 2017

Este conversor funciona com tensões entre 2 e 5.5VCC, e a tensão máxima nos pinos analógicos é igual à tensão de alimentação. Os pinos analógicos podem ser

programados como 4 pinos independentes, ou dois canais diferenciais. A interface de comunicação utilizada pela placa é a I^2C, de fácil conexão com placas como Arduino, Raspberry Pi, Beaglebone, etc.

Embora o ADS1115 tenha resolução de 16 bits, um dos bits na palavra de 16 bits é usado para indicar o bit do sinal, Isso significa que existem 32.768 valores de saída possíveis, onde o zero é o primeiro e 32.767 é o último (HACKSTER.IO, 2017).

A tensão máxima mensurável é estabelecida pela tensão de alimentação do chip, especificamente, a tensão máxima mensurável é 0,3 volts a mais que a tensão de alimentação. De forma, que se for aplicado VCC de 5V para o ADS1115 e definida a configuração PGA (Ganho programável via software) padrão (+/- 6.144V), a tensão máxima que pode ser aplicada na entrada analógica não deve ser maior que 5.3V (5 + 0.3). Essa tensão não pode exceder esse valor na entrada analógica, sob risco de danificar o chip.

A tensão de alimentação do NodeMCU é de no máximo 3,3V , e os sensores podem trabalhar no mesmo valor de tensão. Isso significa que obtem-se um fator de escala de (4,096/32767) , ou seja, 1 bit = 0,125 mV. No caso do projeto, porém, o PGA está sendo configurado para +/- 4,096V, por causa da tensão nos sensores.

4.3 Módulo regulador de tensão LM2596

O módulo regulador de tensão LM2596, ilustrado na Figura 15, trabalha como conversor DC-DC no modo Step Down, sendo capaz de reduzir uma carga de até 3A com ótima eficiência. A tensão de saída pode ser ajustada entre 1,5 a 35V, tendo como entrada 3,2 a 40V .

Figura 15 - Módulo regulador de tensão LM2596

Fonte: Filipeflop. Regulador de Tensão LM2596, 2022.

No módulo tem-se um potenciômetro para ajustar a tensão de saída desejada, dois terminais de entrada e dois de saída.

Este módulo regulador de tensão é ideal para alimentação de motores, relés, displays ou outros componentes elétricos que operem dentro das faixas de tensões e correntes especificadas (FILIPFLOP. REGULADOR DE TENSÃO LM2596 CONVERSOR DC-DC STEP DOWN, 2022).

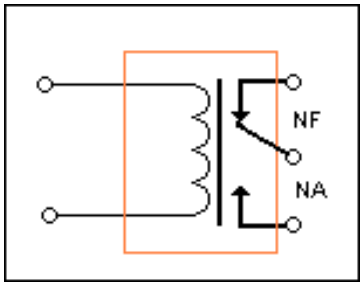
Neste projeto utilizou-se o módulo regulador de tensão para reduzir a tensão de entrada na fonte de alimentação de 12V para 5V, de forma que fosse possivel alimentar o módulo relé 5V de 2 canais de modo seguro. Este por sua vez, foi responsável pelo acionamento da lâmpada e do microventilador .

4.4 Relés

O relé é um dispositivo eletromecânico com inúmeras aplicações em comutação de contatos elétricos, usado para ligar ou desligar dispositivos. No relé eletromecânico a comutação é realizada alimentando-se a sua bobina. Quando uma corrente originada no primeiro circuito passa pela bobina, um campo eletromagnético é gerado, possibilitando o funcionamento do segundo circuito. Sendo assim, uma das aplicações do relé é usar baixas tensões e correntes para o comando no primeiro circuito, protegendo o operador das possíveis altas tensões e correntes que irão circular no segundo circuito (contatos).

Na Figura 16, mostra-se um circuito elétrico interno de um relé. Quando a bobina é energizada, ela cria um campo eletromagnético, que funciona como um imã e portanto, atrai e desloca o contato. Então o relé passa a desconectar do NF (Normalmente Fechado) do contato central que passa a estar conectado no NA (Normalmente Aberto). Em estado de repouso, o contato está na posição NF (normalmente fechado). Dessa forma, a bobina é totalmente isolada dos contatos que serão chaveados (CIRCUITOS-ELECTRICOS, 2021).

Figura 16 - Circuito interno de um relé



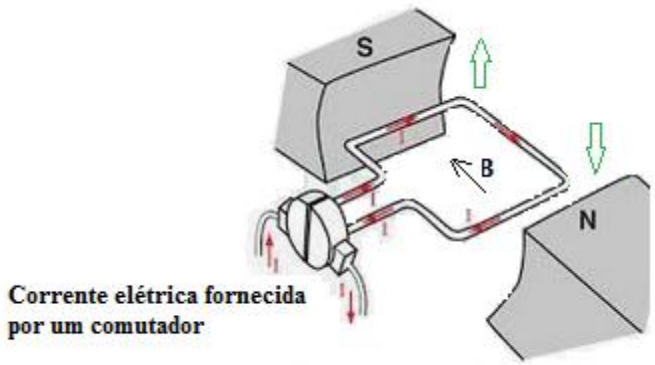
Fonte: Circuitos-Electricos, 2021

4.4 Motor CC

Um motor de corrente contínua (CC) é um dispositivo que faz a conversão de energia elétrica para energia mecânica, classificado como um atuador eletromagnético. Geralmente trabalha junto a um equipamento de controle e posicionadores. Atuadores eletromagnéticos são os atuadores mais utilizados em robôs, principalmente, os motores CC que são mais fáceis de programar seus movimentos.

Um motor de corrente contínua é composto por três partes básicas, o estator, o rotor e o comutador. O princípio básico de funcionamento ocorre sempre que um condutor conduzindo uma corrente elétrica (em vermelho) é colocado em um campo magnético, B (direcionado do Norte para Sul) este condutor experimenta uma força mecânica (em verde), gerando o torque e o giro do eixo do motor, conforme ilustrado na Figura 17. O comutador inverte a corrente a cada meia volta da bobina para manter o torque agindo na mesma direção (FITZGERALD, 1975).

Figura 17 – Circuito básico para um motor CC



Fonte: Elaborado pelo autor.

Neste trabalho tem-se um motor CC no microventilador que terá seu acionamento controlado pelo NodeMCU sempre que houver necessidade de ventilar o ambiente, detectado pelo sensor de temperatura.

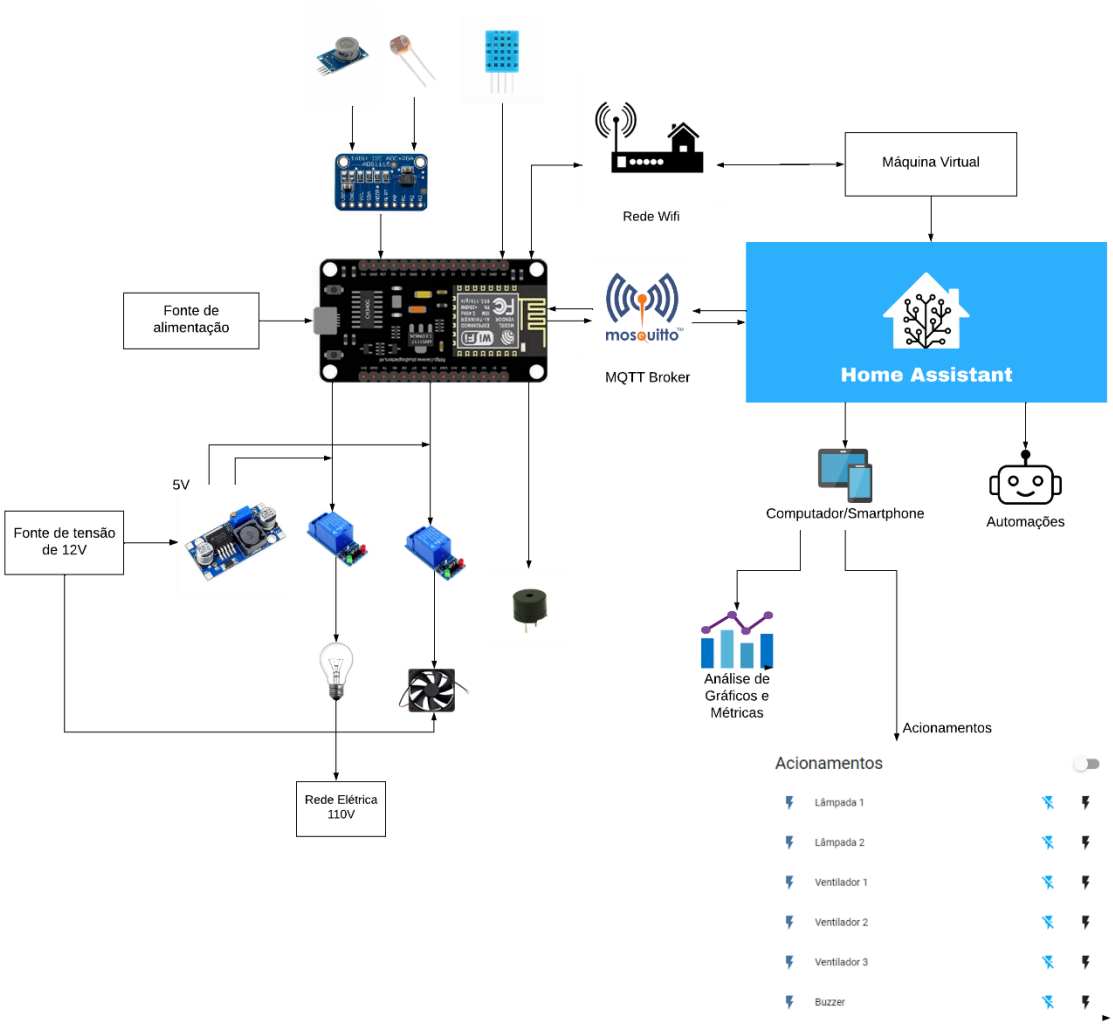
5 DESENVOLVIMENTO DO MÓDULO DIDÁTICO IoT

Depois de obter os subsídios necessários sobre plataformas de controle, software de desenvolvimento , estudo dos vários sensores, e o interfaceamento com um aplicativo de monitoramento e atuação , descrevem-se neste capítulo, as etapas realizadas para a execução do projeto, incluindo os testes iniciais de funcionamento dos sensores até a conexão com o *Home Assistant*, e a simulação com os sensores em um ambiente (casa) monitorada.

5.1 Arquitetura implementada para o módulo didático IoT

O Diagrama de blocos ilustrativo geral para o módulo didático IoT implementado é apresentado na Figura 18.

Figura 18 - Diagrama de blocos para o módulo didático IoT.



Fonte: Elaborado pelo autor

Neste têm-se a ilustração dos sensores de luminosidade, LDR e monóxido de carbono, MQ-7 com saídas analógicas, conectados ao conversor analógico, ADS1115 e o sensor digital DHT11 conectado diretamente ao NodeMCU fornecendo as medidas de temperatura e umidade que são armazenados em strings.

O NodeMCU conecta-se ao *Home Assistant* por meio de *wifi*, cujo aplicativo recebe via assinatura pelos tópicos MQTT, a leitura dos sensores e estados dos atuadores. O NodeMCU é programado pela porta USB de um computador (PC) e o *Home Assistant* é executado em uma máquina virtual instalada neste mesmo computador.

Por meio de um ponto de acesso, o NodeMcu conecta-se à internet e se comunica com o sistema operacional Hassio, que pode ser acessado de qualquer lugar por computadores e/ou smartphones. Além disso, o NodeMCU e o Home Assistant se conectam ao broker Mosquitto para que seja possível a utilização do protocolo MQTT e a inscrição nos tópicos de interesse. Este sistema operacional, assina os tópicos MQTT do broker Mosquitto e faz o monitoramento dos sensores e acionamento dos atuadores, lâmpada e microventilador, armazenando esses resultados na nuvem e disponiblizando os valores das leituras e a indicação do acionamento na forma gráfica, em um *dashboard*.

Neste diagrama é mostrado também, que o acionamento pode ser realizado manualmente por botões implementados no *dashboard* do *Home Assistant* ou usando as automações programadas que interpretam os dados recebidos pelos sensores e fazem o acionamento dos relés e um buzzer. Os relés acionam (on/off) uma lâmpada e um mocroventilador de 12V. Dessa forma, o *Home Assistant* centraliza o monitoramento e controla todo o módulo, fazendo o papel de servidor de aplicação.

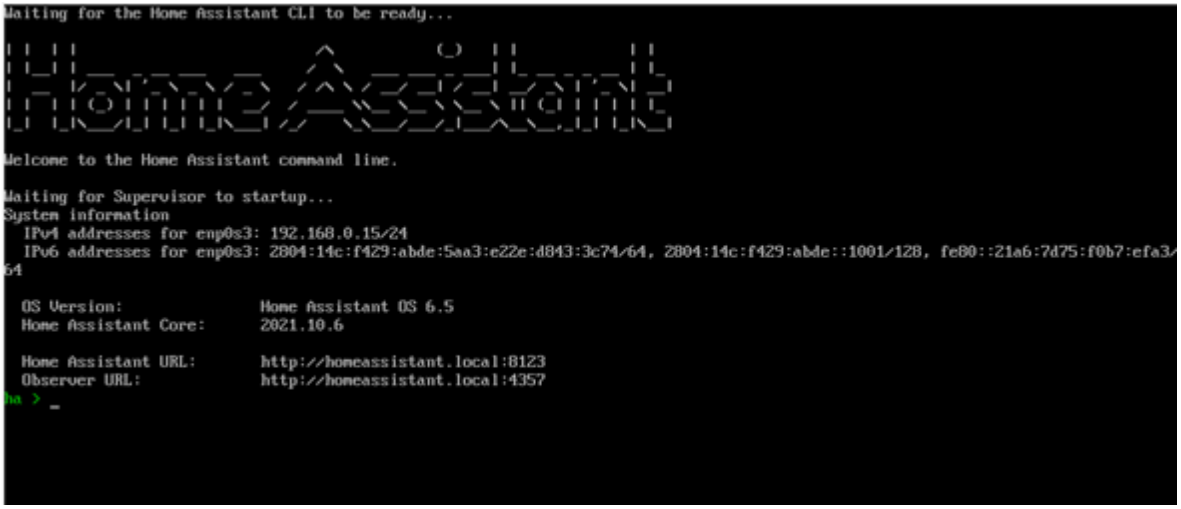
5.2 Conectando o NodeMCU ao *Home Assistant*

Primeiramente configurou-se a IDE do Arduino para o uso do módulo ESP8266 NodeMcu, com o auxílio do Guia IoT (FLOP, 2018).

Para que fosse possível realizar o envio de dados à plataforma *Home Assistant* foi necessário instalar o sistema operacional Hass.io em uma máquina virtual no PC. Ao instalar esta máquina virtual é possível criar um nome e senha de

acesso, para maior segurança do projeto. Para executar o Home Assistant, basta inserir o Ipv4 (Internet protocol v.4) ou a URL (Uniform Resource Locator) do endereço local, no navegador web, tais dados são disponibilizados pela execução na máquina virtual, conforme ilustrado na Figura 19.

Figura 19 - Home Assistant em máquina virtual



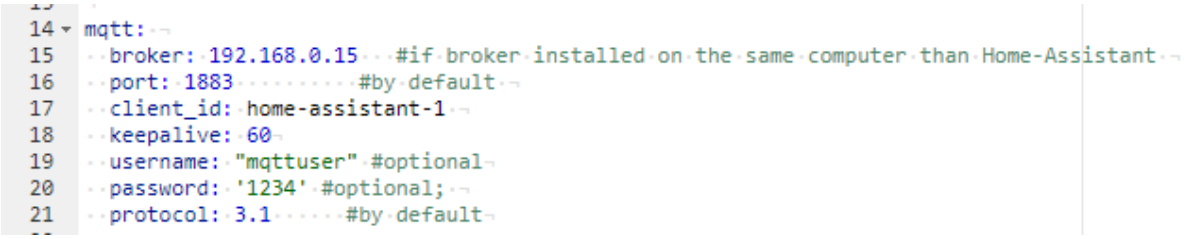
Fonte: Elaborado pelo autor.

Na loja de *add-ons* oficiais do sistema operacional, instalou-se o *Mosquitto broker*, criou-se um usuário MQTT, configurando-o , como ilustrado nas Figuras 20 e 21.

Figura 20 - Configurações broker MQTT dentro do add-ons

Fonte: Elaborado pelo autor.

Figura 21 - Configurações MQTT YAML dentro do arquivo de configurações do Home Assistant



```
mqtt:
  broker: 192.168.0.15   #if broker installed on the same computer than Home-Assistant
  port: 1883             #by default
  client_id: home-assistant-1
  keepalive: 60
  username: "mqttuser" #optional
  password: '1234' #optional;
  protocol: 3.1          #by default
```

Fonte: Elaborado pelo autor.

Todas as configurações do *Home Assistant* são feitas por arquivos de configuração escritos na linguagem YAML, que facilita o entendimento da serialização dos dados pelos desenvolvedores. Neste trabalho utilizam-se três arquivos de configurações YAML, sendo :

- Automation.yaml, onde são descritas todas as automações do sistema;
- Configuration.yaml onde descrevem-se todas as configurações dos sensores e atuadores, além de configurações gerais do sistema ;
- Customize.yaml , que descrevem as customizações de entidades, por exemplo, um ícone customizado.

A conexão do NodeMcu com a internet , por meio de um ponto de acesso a um computador com Windows 10 e com o broker MQTT Mosquitto, é realizada adicionando o nome e senha da rede wifi, usuário e senha do broker MQTT e o IP do servidor. Estes dados e trecho do código implementado na IDE do Arduino podem ser vistos na Figura 22.

Figura 22 - Código da conexão NodeMCU com a internet e o broker MQTT

```
#define wifi_ssid "Bia"
#define wifi_password "bia123456"

#define mqtt_server "186.217.93.131"
#define mqtt_user "mqttuser"
#define mqtt_password "1234"
```

Fonte: Elaborado pelo autor.

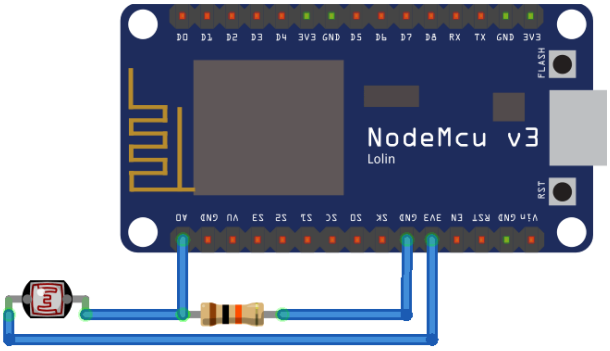
5.3 Testes individuais realizados para os sensores

Neste item apresentam-se os testes realizados com os sensores utilizados no projeto visando verificar seu funcionamento, calibração e o desempenho. São apresentados o LDR, MQ7 e por fim o de umidade e temperatura integrados.

5.3.1 Testes para o LDR

Visando observar a variação da resistência do sensor LDR, conforme a incidência de luz do ambiente, foi montado o circuito esquemático mostrado na Figura 23. Neste circuito foi usado um resitor de 10kΩ em série com o LDR conectados ao NodeMCU, conforme circuito esquemático da Figura 10.

Figura 23 - Circuito como o NodeMCU para os testes com o LDR



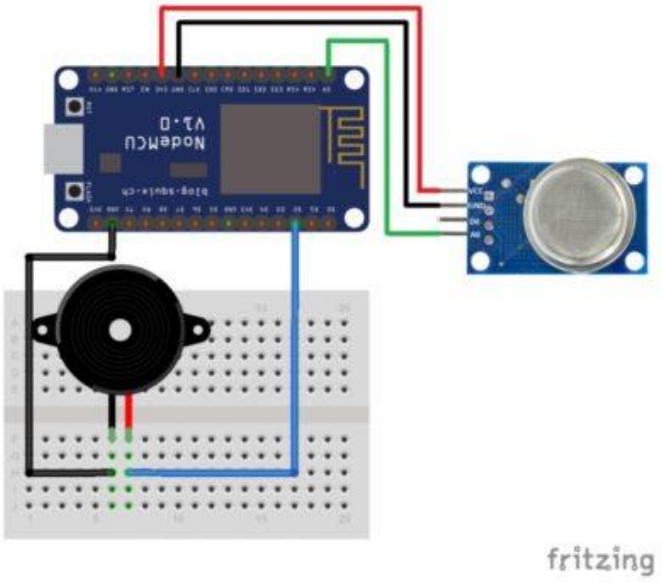
Fonte: Geekering, 2021

A calibração do sensor consistiu em anotar a variação dos valores de tensão obtidos no ambiente com máxima luminosidade, com pouca luz, e por fim nenhuma luz, após isso, definiu-se no programa principal, um valor para o qual as lâmpadas fossem acesas automáticamente (para um mínimo de luz no ambiente). Com este sensor mostra-se didaticamente que é possível fazer o monitoramento visando à economia de energia em uma casa.

5.3.2 Testes para o MQ-7

Testou-se o sensor de gás MQ-7 simulando o vazamento de gás e o surgimento de fogo no ambiente, usando um isqueiro, cujo alerta é dado por um buzzer com alimentação em 5VCC. A montagem, utilizando o NodeMCU é ilustrada na Figura 24.

Figura 24 - Circuito detector de vazamento de gás com MQ-7 e buzzer



Fonte: Mindstorm engineering , 2021.

Nos testes para este sensor atribuí-se um valor máximo na codificação para representar o nível normal de gás no ambiente. Caso este valor seja excedido, o buzzer é ativado, emitindo um sinal sonoro que permanecerá ativado até que o nível de gás volte ao normal.

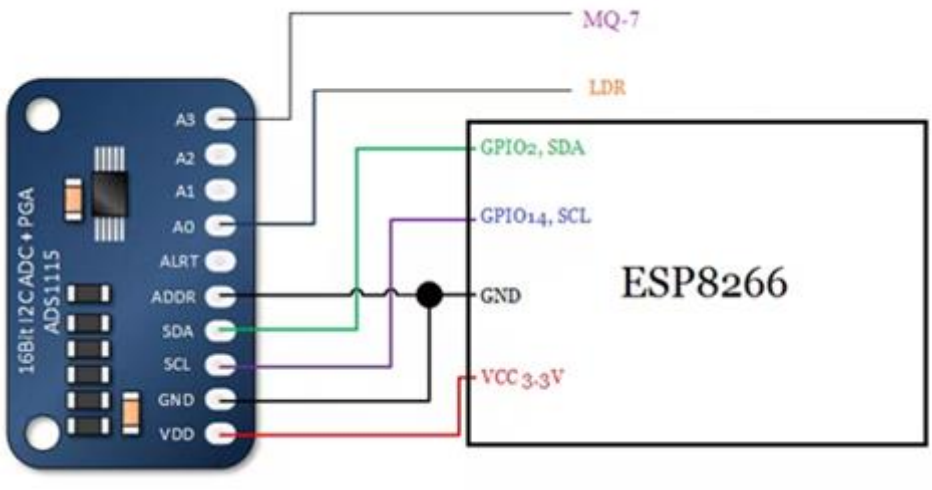
Com o isqueiro aceso próximo ao sensor, eleva-se pelo aquecimento o nível de monóxido de carbono. Na verdade, deve-se esperar cerca de 3 minutos para que a resistência do sensor diminua e o sensor se estabilize, para poder considerar o resultado da medida. Com a programação adequada usando o monitor serial da IDE, pode-se visualizar a indicação do valor analógico dado pelo sensor e o sinal digital emitido pelo módulo MQ-7, em resposta à detecção do gás. O sinal analógico, AOUT de saída , fornece uma resolução melhor, medindo a variação da concentração do gás no ar.

A escolha do sensor de gás para esta pesquisa se deve a preocupação com a segurança de uma casa (ou ambiente) e por meio dele acionar um alarme, de forma a iniciar um socorro e salvar vidas, visto ser este gás altamente tóxico para o ser humano. Este sensor é frequentemente usado em projetos de segurança e automação residencial do tipo experimental.

Por fim, for falta de entradas analógicas no NodeMCU conectou-se os dois

sensores LDR e o módulo MQ-7 com saídas analógicas, a um canal do ADC ADS1115, e em seguida ao NodeMCU, conforme apresentado na Figura 25, visando os testes finais com estes sensores.

Figura 25 - Sensores LDR e MQ-7 ligados no ADS1115 e ao NodeMCU



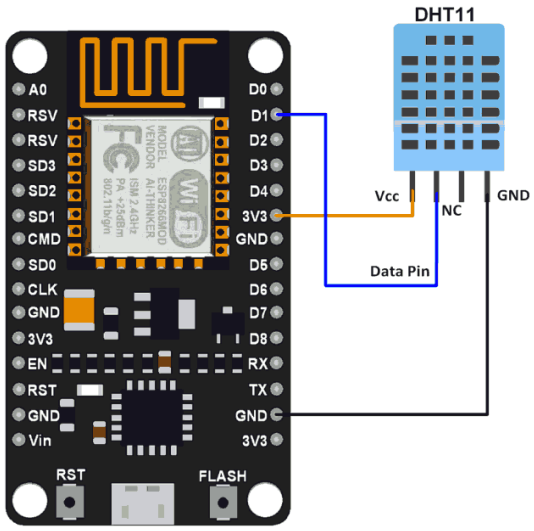
Fonte: Elaborado pelo autor.

Após configurar o PGA para +/- 4,096V, devido a tensão máxima dos sensores ser de 3,3V, tensão de alimentação dada pelo NodeMCU, repetiu-se o procedimento de calibragem dos respectivos sensores e anotou-se os valores de referência para o acionamento automático do atuador relacionado ao LDR, a lâmpada e no caso do MQ-7, o buzzer.

5.3.3 Testes para o sensor DHT-11

Como se trata de um módulo, basta fazer a ligação direta do DHT-11 no NodeMCU, como mostrado na Figura 26 e criar o programa. Na codificação do programa foi utilizada a biblioteca "DHT.h", disponível na lista de bibliotecas da IDE do Arduino.

Figura 26 - Esquema de ligação DHT-11 e o NodeMCU



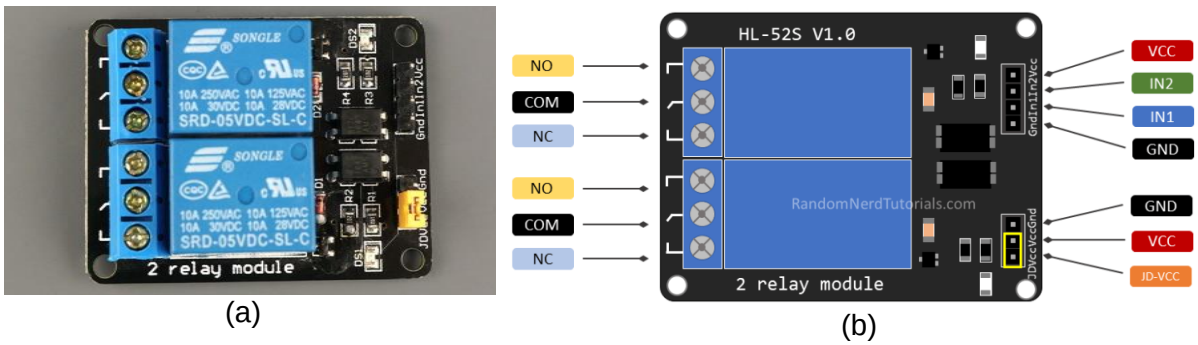
Fonte: Eletrowings, 2018.

Para simular as condições dos testes para o módulo DHT-11 utilizou-se o ar-condicionado do ambiente (laboratório) para obter a variação da temperatura e umidade relativa. Com este sensor no projeto de supervisão e o monitoramento de um ambiente tem-se como objetivo o conforto térmico.

5.4 Módulo Relé e os testes de acionamento

Neste projeto utiliza-se um módulo relé de 5VCC com 2 canais, mostrado na Figura 27 (a) e em (b) a pinagem, equipado com transistores, conectores, dois fotoacopladores, leds, diodos e relés de alta qualidade.

Figura 27 - (a) Módulo relé 5VCC, 2 canais. (b) Pinagem



(a) (b)

Fonte: Random nerd tutorials, 2019

O módulo possui um jumper conectando aos pinos VCC e JD-VCC, ilustrado na Figura 25 (a) em amarelo. Esse jumper significa que o eletroímã do relé

é alimentado diretamente pelo pino de alimentação do NodeMCU ESP8266, de modo que o módulo relé e os circuitos do NodeMCU não são fisicamente isolados. Sem o jumper, é necessário fornecer uma fonte de energia independente para alimentar o eletroímã do relé pelo pino JD-VCC. Essa configuração isola fisicamente os relés do NodeMCU por meio de um optoacoplador embutido no módulo, o que evita danos ao NodeMCU em caso de picos elétricos. Cada canal possui um LED para indicar o estado da saída do relé, cujas especificações são fornecidas na Tabela 5 (Randomnerdtutorials, 2019).

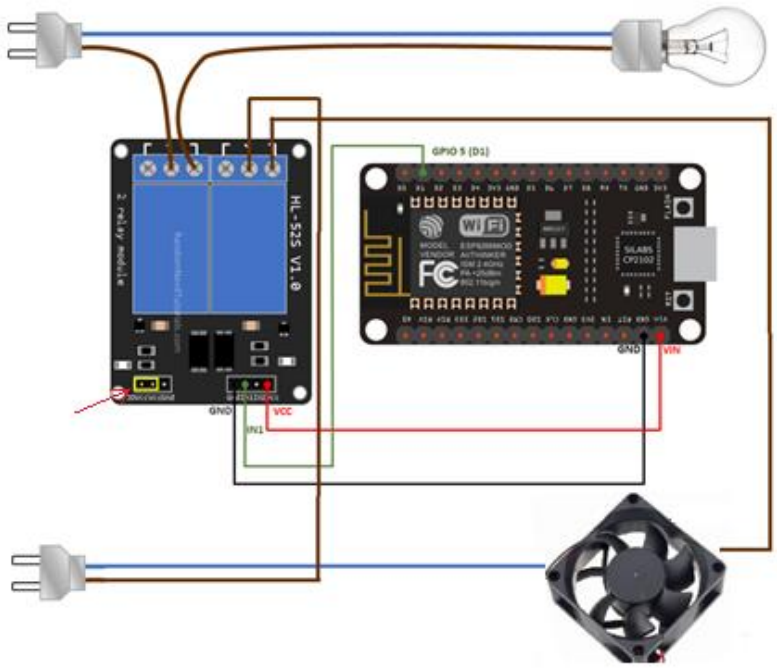
Tabela 5 - Especificações do módulo relé, modelo SRD-05VDC-SL-C

Tensão de operação	5VCC
Permite controlar cargas de	220V AC
Corrente típica de operação	15~20mA
LED indicador de status	2
Pinagem	Normal Aberto, Normal Fechado e Comum
Tensão de saída	(30 VCC a 10A) ou (250VAC a 10A)
Tempo de resposta	5~10ms
Dimensões e peso	51 x 38 x 20mm, 30g

Fonte: Arducore, 2021

Neste projeto os relés foram utilizados para acionamento de uma lâmpada de 110V e de um microventilador de computador, de 12VCC, usando o esquema de ligação mostrado na Figura 27. Devido a presença dos fotoacopladores no módulo e a diferença de tensão de entrada entre o relé e o NodeMCU, 5V e 3,3V, respecitvamente, retirou-se o jumper das entradas JD e VCC do módulo relé, seta em vermelho na Figura 28, para que funcionasse corretamente protegendo o NodeMCU das tensões e correntes elevadas que passam pelo relé. Para a alimentação do JD-VCC, utilizou-se um regulador de tensão de 5V. Testou-se o acionamento do módulo tanto por meio de botões, quanto por meio de automação, programadas a partir de valores dos sensores LDR e DHT-11, obtidos anteriormente, codificando o valor de acionamento da lâmpada e do microventilador.

Figura 28 - Esquema de ligação dos relés



Fonte: Elaborado pelo autor.

5.5 Configurando tópicos MQTT e automações no Home Assistant

Para registrar os sensores e seus respectivos tópicos de leitura MQTT no Home Assistant, é necessário adicionar o código apresentado na Figura 29, dentro do arquivo configurations.yaml.

Figura 29 - Sensores configuration.YAML

```
sensor:
- platform: mqtt
  state_topic: "sensor/temperature"
  name: "Temperature"
  qos: 0
  unit_of_measurement: "°C"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/humidity"
  name: "Humidity"
  qos: 0
  unit_of_measurement: "percent"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/light"
  name: "Light Level"
  qos: 0
  unit_of_measurement: "lux"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/gas"
  name: "CO Level"
  qos: 0
  unit_of_measurement: "ppm"
  #value_template: '{{ payload }}'
```

Fonte: Elaborado pelo autor.

Como dito anteriormente, o recuo em cada linha do programa é uma parte importante da especificação de relacionamentos e hierarquias dentro do YAML, assim todos os sensores devem ser especificados dentro do bloco “sensor”, como mostrado na Figura 29. O parâmetro “plataform: mqtt” indica para o sistema que é utilizado o protocolo em nossos sensores; para assinar um tópico de leitura deve-se especificar em “state_topic” exatamente com o mesmo nome dos tópicos descritos no código gerado na IDE do arduino; “name” e “unit_of_measurement”, neste caso, referem-se ao nome amigável do sensor e a sua unidade de medida, respectivamente, que é desejado que apareça no *dashboard* com os botões. Utilizou-se QoS (Quality of Service) como “0” (pelo menos uma vez), pois este entrega o dado aos assinantes ocupando menor quantidade de memória *Flash* do NodeMCU.

Para os dispositivos atuadores, realizou-se um processo similar, onde os relés e o buzzer foram definidos como “*switch1*”, “*switch2*” e “*switch3*”, mostrado na Figura 30.

Figura 30 - Atuadores configuration.YAML

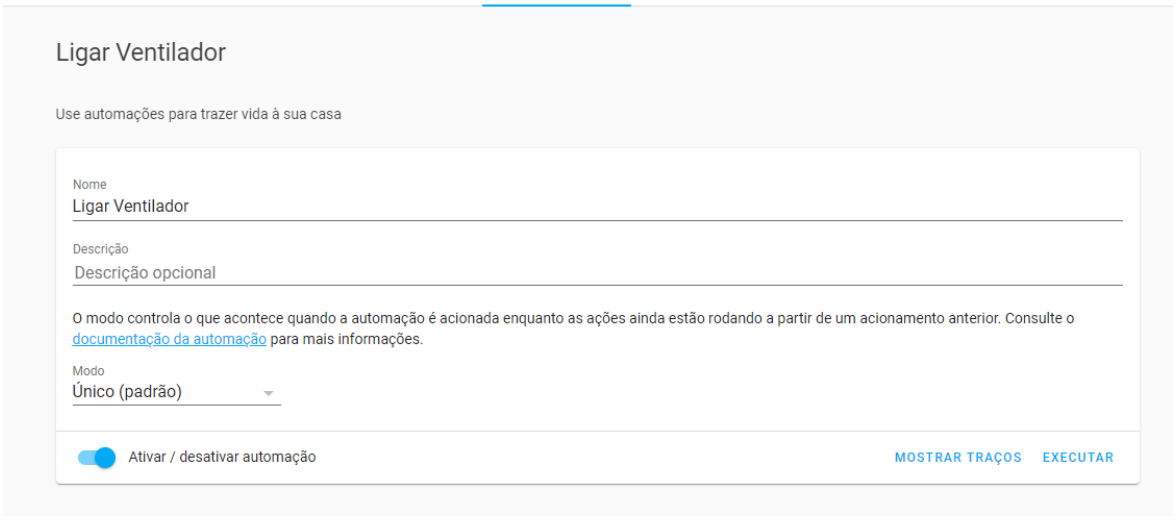
```
switch 1:
  platform: mqtt
  name: "Lâmpada 1"
  command_topic: "homeassistant/switch1"
  payload_on: "ON"
  payload_off: "OFF"
  optimistic: true   # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'

switch 2:
  platform: mqtt
  name: "Ventilador"
  command_topic: "homeassistant/switch2"
  payload_on: "ON"
  payload_off: "OFF"
  optimistic: true   # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'

switch 3:
  platform: mqtt
  name: "Buzzer"
  command_topic: "homeassistant/switch3"
  payload_on: "ON"
  payload_off: "OFF"
  optimistic: true   # # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'
```

Fonte: Elaborado pelo autor.

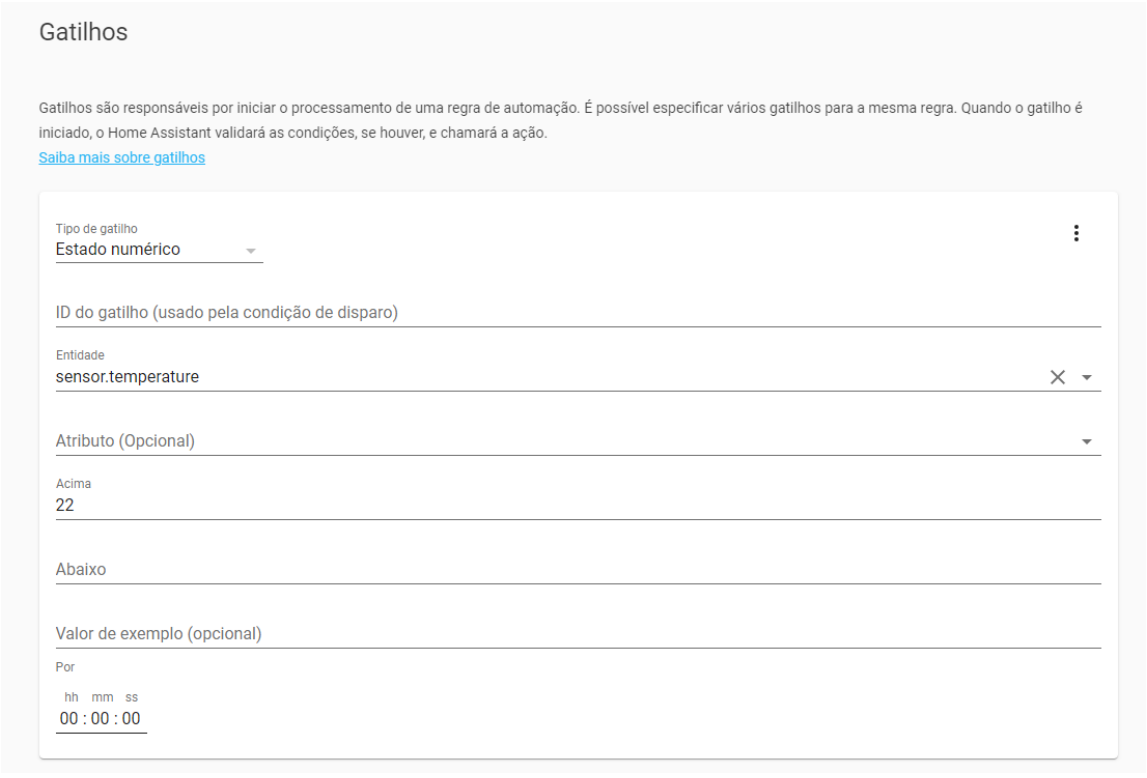
Os parâmetros de *payload* geram serviços no Home Assistant que são chamados pelas automações para realizar ações no sistema, onde *ON* e *OFF* representam os estados ligado e desligado, respectivamente. “Retain” e “optimistic” foram definidos como true para que a informação do último estado lido seja mantida no broker.

Com os sensores físicos e lógicos devidamente conectados, é possível gerar automações pelo Home Assistant baseadas nesses sensores, basta inserir os parâmetros da automação, como é possível observar nas Figuras 31(a), (b) e (c). O Home Assistant fornece um gerador automático de automações no arquivo automations.yaml.

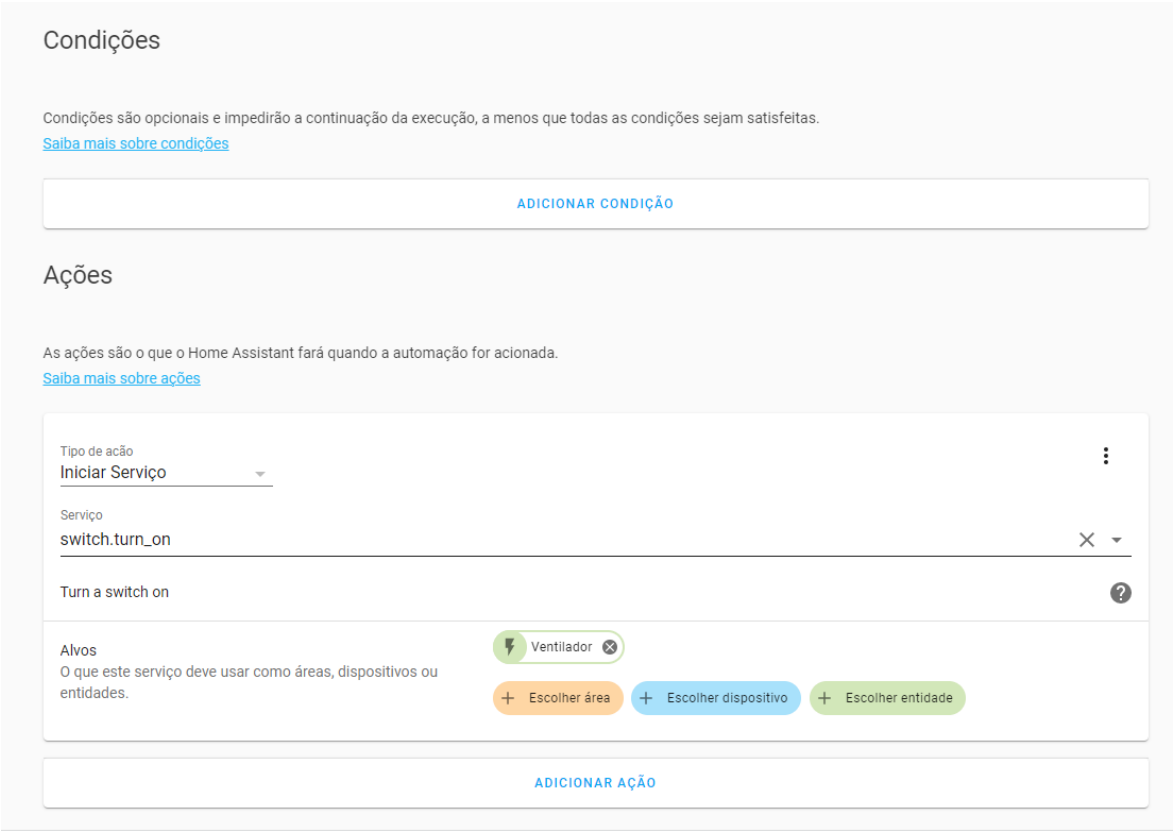
Figura 31 - Automação “Ligar ventilador” pelo Home Assistant. a) Parâmetros “Nome e Modo”. b) Gatilhos. c) Condições e ações.



(a)



(b)



(c)
Fonte: Elaborado pelo autor.

Este procedimento foi realizado para todas as seis automações configuradas, sendo: “Ligar Lâmpada”, “Desligar Lâmpada”, “Ligar Ventilador”, “Desligar Ventilador”, “Ligar Alarme” e “Desligar Alarme”, conforme lista de automações do Home Assistant na Figura 32. As automações criadas podem ser ativadas ou desativadas a qualquer momento pelo utilizador do Home Assistant.

Figura 32 - Lista de automações programadas

Fonte: Elaborado pelo autor.

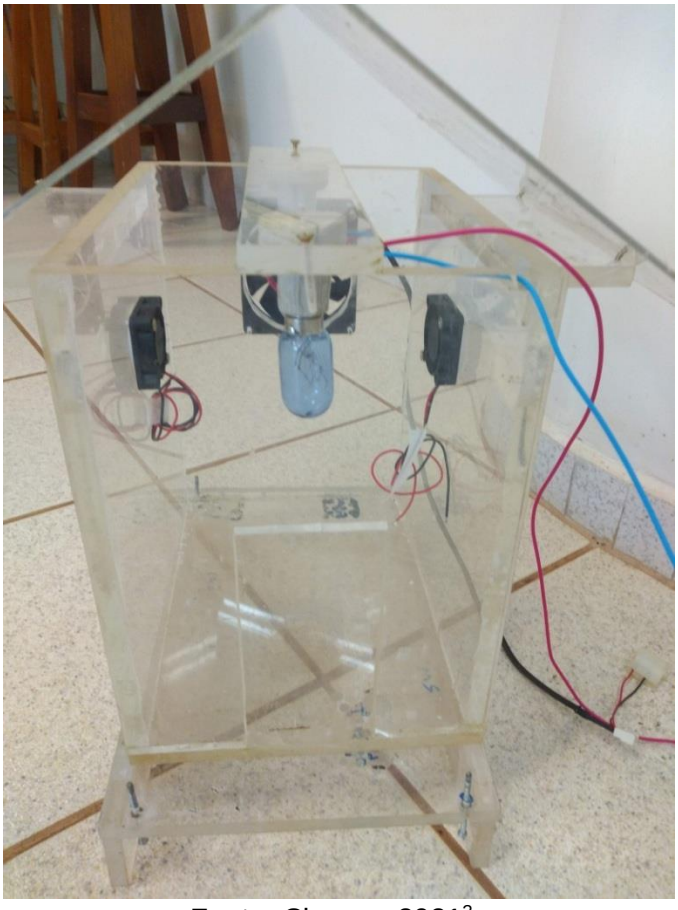
6 RESULTADOS DOS TESTES FINAIS E DISCUSSÔES

Depois da apresentação do desenvolvimento do projeto descrevem – se a seguir, os resultados dos testes finais com o circuito, protótipo e software implementado no NodeMCU, utilizando a plataforma Home Assistant.

6.1 O protótipo e o circuito

Os testes finais foram realizados em um protótipo de uma casa, em acrílico, no tamanho, 19cmX24cmx35,5cm (comprimentoxlarguraxaltura), mostrado na Figura 33 e nesta foram colocadas lâmpadas de 110VAC e microventiladores de 12VCC. Uma observação se faz mecessária quanto aos testes realizados, embora no protótipo da casa tenham 3 ventiladores e duas lâmpadas, nos testes apresentados somente foram acionadas uma lâmpada e um microventilador.

Figura 33 - Protótipo da casa em acrílico

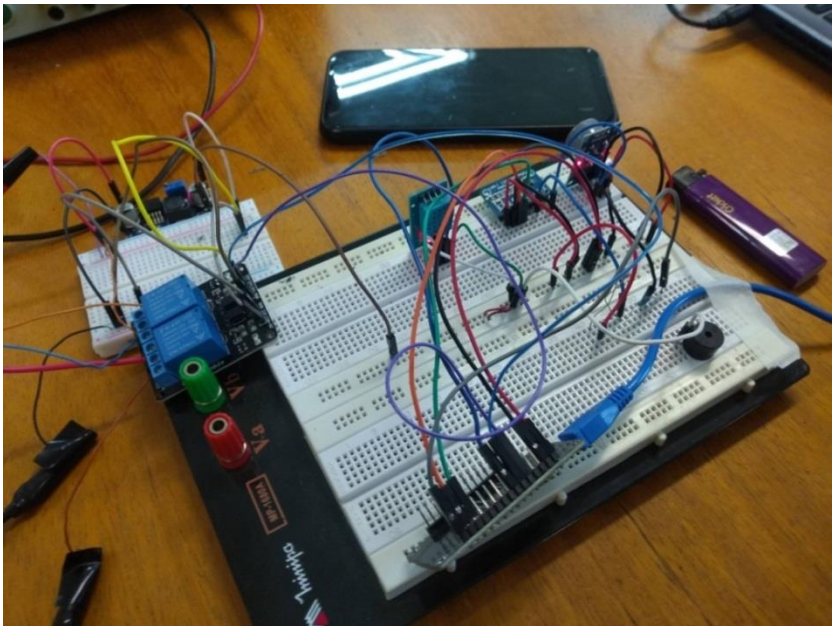


Fonte: Chaves, 2021[3] .

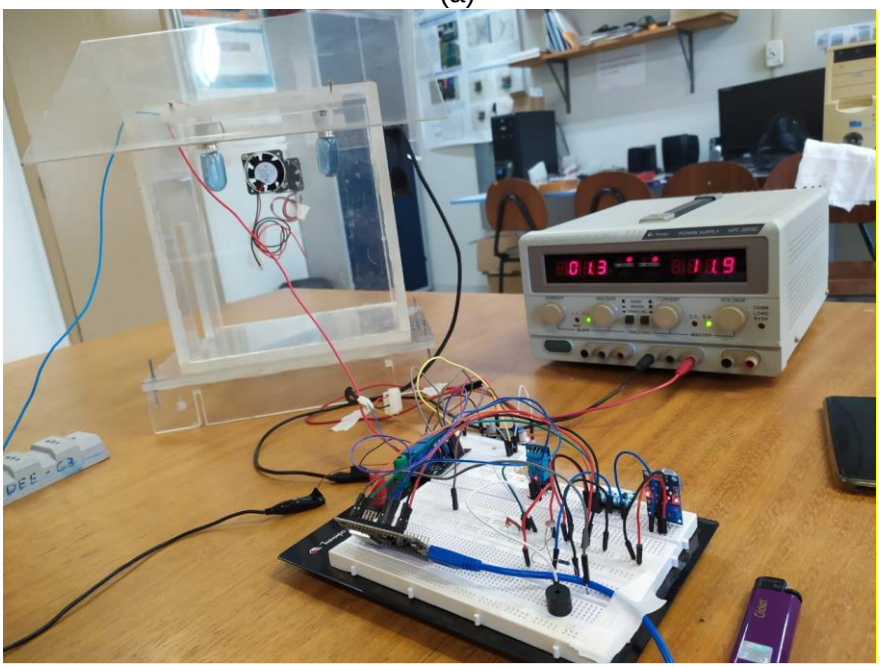
[3] Protótipo elaborado por Valdemir Chaves ,técnico de laboratório –DEE-FEIS-UNESP

O circuito com sensores e atuadores foi montado em uma *protoboard*, devido principalmente à situação de pandemia no mundo e a dificuldade de acesso aos laboratórios. A montagem completa com computador, fonte e o protótipo da casa são mostrados na Figura 34.

Figura 34 - a) Circuito com sensores e atuadores. b) Montagem completa.



(a)



(b)

Fonte: Elaborado pelo autor

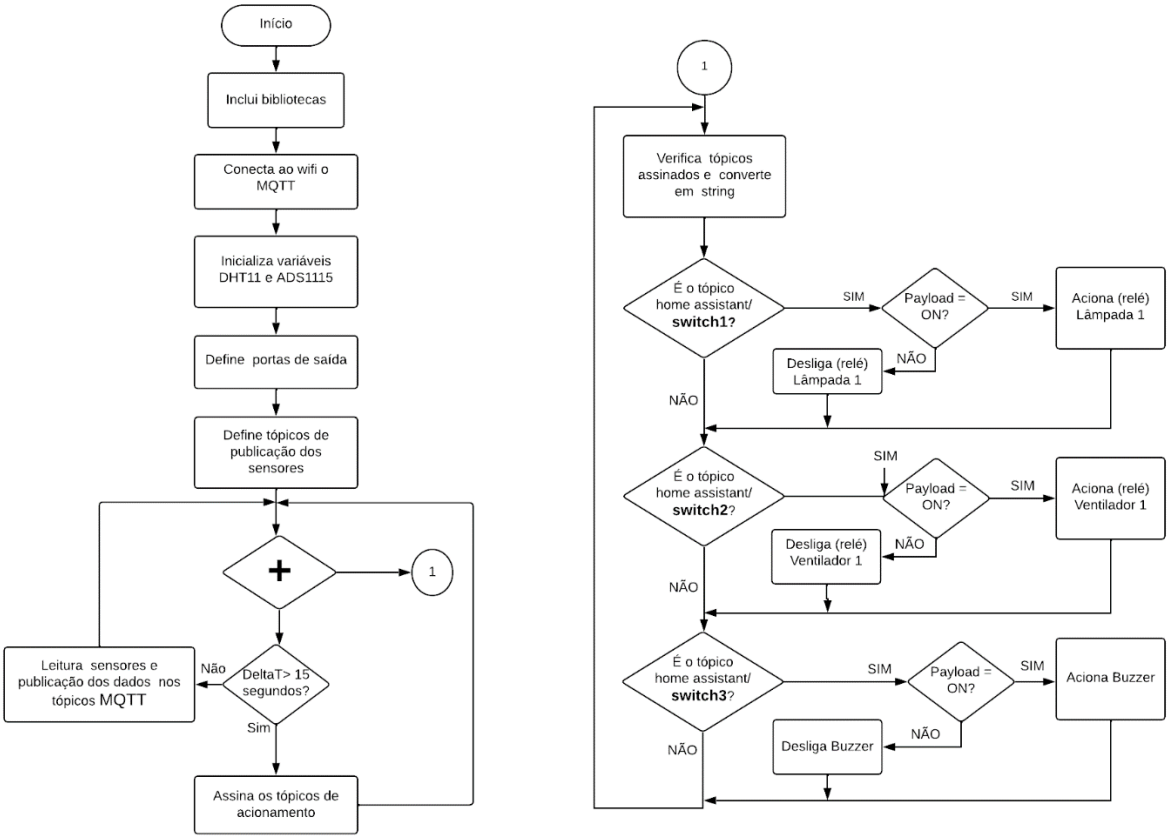
Quanto às alimentações foram utilizadas uma fonte de tensão regulada em 11.9V, para alimentar o microventilador e o regulador de tensão, que por sua vez foi

ajustado em 5V para alimentar os relés; a lâmpada foi alimentada pela rede elétrica local de 110V e o NodeMCU pela entrada USB do computador.

6.2 Fluxograma do programa do NodeMCU

Com a finalidade de documentar melhor o programa criado para a aplicação do módulo IoT, é apresentado na Figura 35, o fluxograma do programa desenvolvido para o NodeMCU ESP8266, cujo código encontra-se no Apêndice B.

Figura 35 - Fluxograma do programa embarcado no NodeMCU



Fonte: Elaborado pelo autor

Ao ser inicializado, o programa realiza a leitura das bibliotecas incluídas e das crendeciais de conexão com a rede wifi e com o broker MQTT Mosquitto. Em seguida, define-se os tópicos MQTT que os sensores publicarão seus dados e serão assinados pelo NodeMCU.

Inicializa-se as bibliotecas do sensor DHT11 e o conversor analógico ADS1115, assim como as variáveis que serão utilizadas. Define-se a pinagem das portas de saídas digitais e o ganho do conversor analógico.

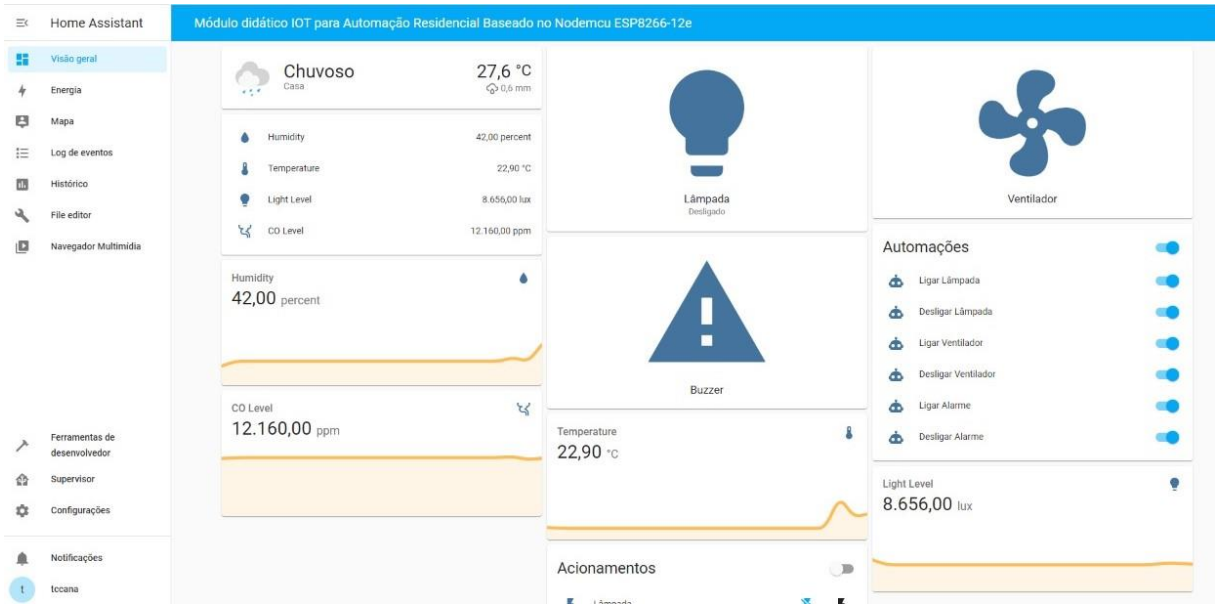
Dentro do loop, verifica-se todos os tópicos MQTT assinados, tanto os de leitura quanto os de publicação, e converte-se os dados floats lidos pelos sensores em strings, bem como a leitura das saídas dos relés. As strings são enviadas para os tópicos assinados pelo Home Assistant e o mesmo exibe os valores lidos pelos sensores em sua *dashboard,* este processo se repete a cada 15 segundos.

Caso um dos relés ou o buzzer tenha seu estado alterado, a função “call-back” é chamada e verifica-se o parâmetro “payload”, se este for igual à “ON” então envia-se o sinal “LOW” para um dos relés ou “HIGH” para o buzzer, ativando-os. Senão, envia-se o sinal “HIGH” para um dos relés ou “LOW” para o buzzer, desativando-os.

6.3 *Dashboard* do projeto no Home Assistant

Com a montagem realizada, conectou-se o NodeMCU ao wifi e estabeleceu-se a conexão com o Home Assistant. Inicialmente, realizou-se a leitura dos sensores e fez-se os acionamentos de forma manual por meio do *dashboard* do Home Assistant para verificação do funcionamento de todas as variáveis, cujo registro da imagem da tela é mostrado na Figura 36.

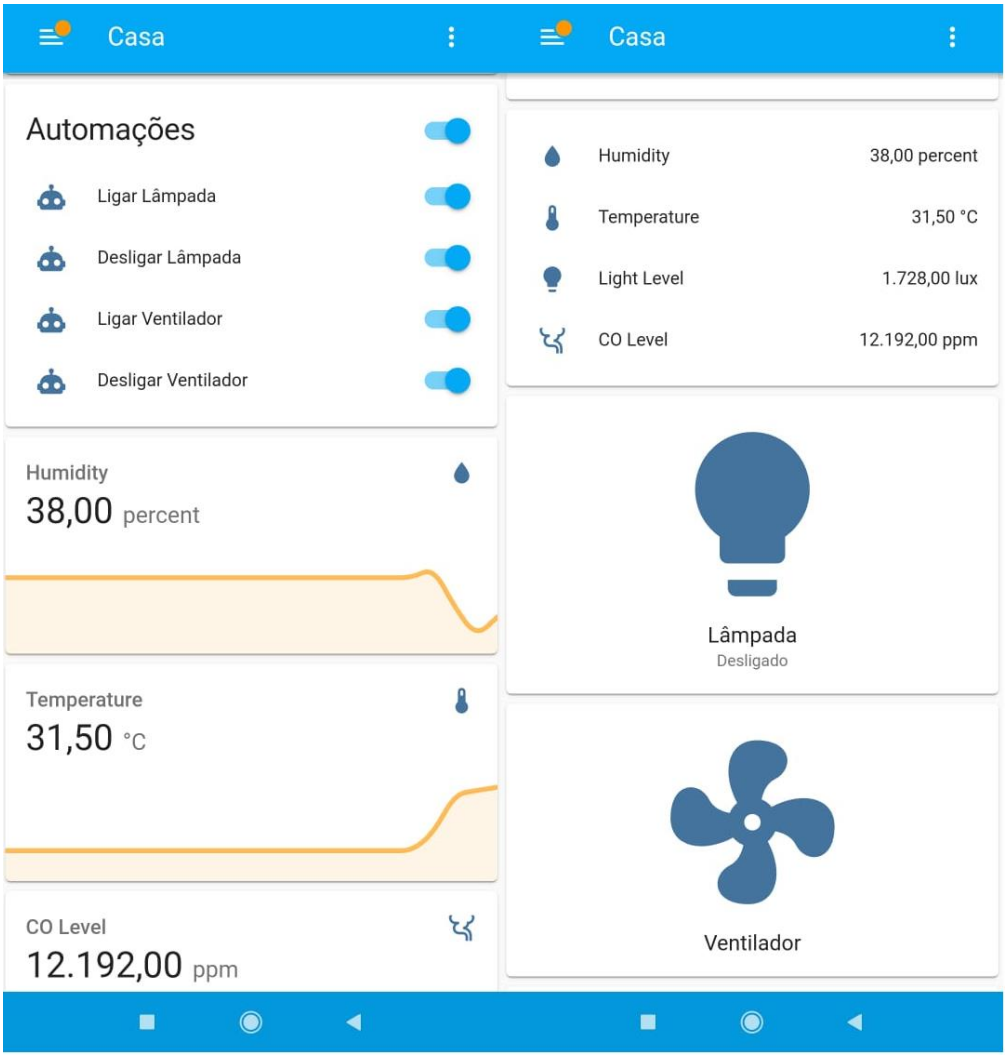
Figura 36 - *Dashboard* Home Assistant na versão web



Fonte: Elaborado pelo autor.

Em seguida foram realizados os mesmos testes na versão do Home Assistant para *smartphone*, e o resultado do acesso é apresentado na Figura 37.

Figura 37 - *Dashboard* Home Assistant na versão para *smartphone*



Fonte: Elaborado pelo autor.

Por fim, testou-se as automações variando as condições do local visando mostrar a resposta a o acionamento da lâmpada (acender) , microventilador (ligar) e disparar o buzzer. De forma que, para a lâmpada utilizou-se a lanterna do celular acesa diretamente sobre o sensor LDR e depois o “bloqueio a luz”, conforme apresentasse pouca luminosidade (sem luz) a lâmpada acenderia.

Para o sensor DHT-11, simulou-se a variação da temperatura do ambiente com o ar-condionado, cujo objetivo era que o microventilador fosse acionado automaticamente com o aumento da temperatura, ventilando o local. No caso do sensor de gás utilizou-se um isqueiro até que o alarme de incêndio (buzzer) fosse acionado, ou seja, alertando para o aumento do monôxido de carbono. Na aba histórico do Home Assistant, é possível observar gráficos das leituras dos sensores e dos atuadores, ao longo do tempo em que o sistema permaneceu ligado, como

ilustrado pelas Figuras 38 (a), (b) e (c). Na Figura 39 têm-se os gráficos da temperatura, ao longo do tempo, do sensor DHT-11 e a temperatura da bateria do smartphone utilizado, que é reconhecida automaticamente pelo Home Assistant quando utiliza-se o aplicativo no smartphone. Por se tratarem da mesma grandeza (temperatura em graus celsius), o Home Assistant automaticamente une os dados em um mesmo gráfico.

Figura 38 - Gráficos ao longo do tempo a) Umidade.b) Luminosidade.c) Monóxido de carbono

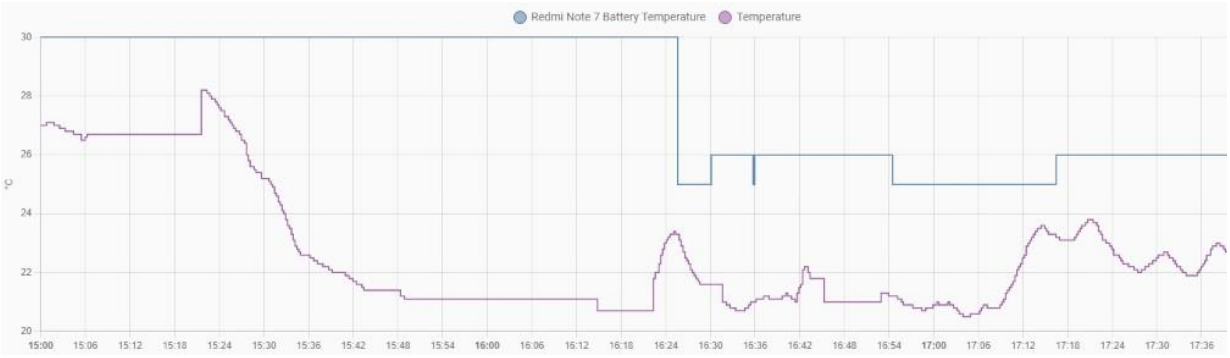
(a)

(b)

(c)

Fonte: Elaborado pelo autor.

Figura 39 - Gráficos da temperatuda da bateria do smartphone e a temperatura dada pelo sensor DHT11, ao longo do tempo



Fonte: Elaborado pelo autor.

Analisando os resultados de todos os testes realizados observa-se que foram bem sucedidos, não sendo o objetivo do trabalho aferir a precisão dos sensores. A ideia do módulo IoT e a sua arquitetura foram devidamente executadas com as implementações de hardware, protótipo e programas, envolvendo um amplo conhecimento e pesquisas. O arquivo de configuração completo, configuration.YAML, para o Home Assistant, encontra-se no Apêndice C.

7 CONCLUSÕES

Neste trabalho foi desenvolvido um modúlo didático IoT para automação residencial baseado no módulo wifi Esp8266 NodeMCU Esp-12E, usando sensores de temperatura, luminosidade, umidade e sensor de monóxido de carbono. Um interfaceamento é realizado entre o módulo didático e um sistema operacional com o objetivo de monitorar e atuar no ambiente de forma remota, por meio da Internet sem fio.

Os conceitos de IoT tomaram conta do mercado de componentes, de forma que são muitos os dispositivos e recursos, redes confiáveis visando integrar os aparelhos de uma casa ou ambiente e atuar de forma a facilitar as tarefas, proporcionar segurança , o acesso a pessoa com alguma deficiência, comodidades, auxiliando na economia de gastos com energia, são alguns dos benefícios.

A plataforma de controle utilizada, o NodeMCU, apresenta características que facilitaram o seu uso, como softwares de desenvolvimento livres e amigáveis, entradas e saídas digitais, além do baixo custo.

Os sensores utilizados são circuitos integrados com tecnologia de montagem em superfície (em inglês Surface Mounted Device –SMD), cujos componentes eletrônicos são ultra-miniaturizados, portando apresentam boa repetibilidade e precisão em suas medidas. Na aquisição de dados pelos sensores com saídas analógicas foi utilizado um conversor analógico digital devido a quantidade mínima de portas analógicas na placa NodeMCU.

Um dos grandes desafios do projeto foi o desenvolvimento da lógica de programação, e o uso dos vários aplicativos visando à comunicação com a internet e a apresentação dos resultados em um monitor ou smartphone. Outro importante aprendizado envolveu os diferentes valores (mais baixos) de alimentação nos dispositivos modernos como os atuadores e o NodeMCU que visam diminuir o consumo e a dissipação. Essa questão exigiu mais estudos e o uso de reguladores de tensão para adequar os diferentes valores de fontes de alimentação.

Ao final do desenvolvimento do projeto e dos resultados apresentados, pode-se concluir que o módulo didático IoT implementado pode proporcionar experimentos importantes para alunos da graduação, em hardware e software, e tornar-se um grande aliado para o ensino em automação residencial, demonstrando com o seu uso a redução do consumo de energia elétrica ao implementar

automações com dias e horários de funcionamento pré-programados, evitando também que lâmpadas e ventiladores esquecidos fiquem ligados ao sair de casa. Para trabalhos futuros sugere-se a elaboração de outros módulos com mais sensores e funções implementadas como abertura /tranca portas , persianas, sensor de presença , etc, utilizando o mesmo planejamento de software e hardware.

8 REFERÊNCIAS BIBLIOGRÁFICAS

ALEXANDRE, Gerônimo Barbosa; SILVA, Wesley Matheus Araújo; SILVA, Tiago Eduardo da. **Sistema De Irrigação Automático Usando Tecnologias De Internet Das Coisas**. In: Congresso Brasileiro de Educação em Engenharia, 47.,2019, Fortaleza. Anais: Abenge, 2019. Disponível em: <http://www.abenge.org.br/cobenge/2019/anais.php>. Acesso em: 04 abr. 2019.

ARDUCORE. **Módulo Relé 5V 2 Canais.** 2021. Disponível em: <https://www.arducore.com.br/modulo-rele-5v-2-canais>. Acesso em: 21 nov. 2021.

ASHTON, Kevin. **That 'Internet of Things' Thing**: In the Real World Things Matter More than Idea. 2009. RFID Journal. Disponível em: <https://www.rfidjournal.com/articles/view?4986>. Acesso 04 abr. 2021.

ASSISTANT, H. **Hass.io.** [S.l.], 2019. Disponível em: <https://www.home-assistant.io/hassio/>. Aceeso em: 10 jul 2021

AVELAR, Camilo Esteves Mendes de. **Domótica aplicada no monitoramento de água utilizando comunicação MQTT e arquitetura de microsserviços: uma solução IoT.** 2019. 63 f. TCC (Graduação) - Curso de Engenharia de Controle e Automação, Universidade Federal de Ouro Preto - Ufop Escola de Minas, Ouro Preto, 2019.

BLOG ELETROGATE. **NodeMCU – ESP12: Guia completo – Introdução (Parte 1).** 2018. Disponível em: <https://blog.eletrogate.com/nodemcu-esp12-introducao-1/>. Acesso em: 28 ago. 2021

CANDIDO, Gradimilo. **Sensor De Gás MQ-135 E A Família De Sensores MQ**: Sensor de Gás MQ-135 e a família MQ de detectores de Gás. 2017. Disponível em: <https://portal.vidadesilicio.com.br/sensor-de-gas-mq-135/>. Acesso em: 28 ago. 2021.

CIRCUITOS-ELECTRICOS. **Componentes Electrónicos - Relé: qué es y cómo funciona**. 2021. Disponível em: <https://www.circuitos-electricos.com/rele-que-es-y-como-funciona/>. Acesso em 11 set. 2021.

COSTA, Rafael Almeida. **Casa inteligente com recurso a tecnologias open source**. 2018. 104 f. Dissertação (Mestrado) - Curso de Sistemas e tecnologias de Informação para as organizações, Instituto Politécnico de Viseu, Viseu, 2018.

DESTERRO ELETRICIDADE. **Casa inteligente: a IoT no futuro da automação residencial**. 2018. Disponível em: < https://www.desterroeletricidade.com.br/blog/sistema-de-seguranca/casa-inteligente-a-iot-no-futuro-da-automacao-residencial/>. Acesso em 22 mar. 2022.

DHT AUTOMAÇÃO. **Seu projeto de automação residencial**. Disponível em: <https://dhtautomacao.com.br/automacao-residencial/?campaign=16349574912&content=583615100528&keyword=sistema%2

0de%20automa%C3%A7%C3%A3o&gclid=CjwKCAjwuYWSBhByEiwAKd_n_gzzzfpvaB-GaO1HuKqcDo6MmRhXy_HDDzqGBKhLKx_vnfnCKnTqDhoClUIQAvD_BwE>. Acesso em : 02 jan 2022.

ELECTRONICWINGS. **DHT11 Sensor Interfacing with NodeMCU**. 2018. Disponível em: <https://www.electronicwings.com/nodemcu/dht11-sensor-interfacing-with-nodemcu>. Acesso em: 23 jun. 2021.

FILIPEFLOP. **Guia Iot para iniciantes em eletrônica: Tudo que você precisa saber para começar. 2018.** Disponível em: <https://www.filipeflop.com/blog/guia-iot-para-iniciantes-em-eletronica/>. Acesso em: 10 abr. 2021.

FILIPEFLOP. **Monitorando Temperatura e Umidade com o sensor DHT11**. 2013. Disponível em: <https://www.filipeflop.com/blog/monitorando-temperatura-e-umidade-com-o-sensor-dht11/>. Acesso em: 23 jun. 2021.

FILIPEFLOP. **Regulador de Tensão LM2596 Conversor DC-DC Step Down**. 2021. Disponível em: <https://www.filipeflop.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/>. Acesso em 01 set 2021.

FILIPEFLOP. **Sensor de Gás MQ-7 Monóxido de Carbono**. 2021 Disponível em: < https://www.filipeflop.com/produto/sensor-de-gas-mq-7-monoxido-de-carbono/>. Acesso em 01 set 2021.

FITZGERALD, A.E. et al., **Máquinas Elétricas**, McGraw-Hill do Brasil, 1975.

FREITAS, A. de P. P. Análise bibliométrica da produção científica sobre indústria 4.0. Trabalho de Conclusão de Curso - Universidade Federal de Uberlândia – UFU.2018

GEEKERING. **ESP8266 NodeMCU – Simple LDR Lux Meter.** 2021. Disponível em: <https://www.geekering.com/categories/embedded-sytems/esp8266/ricardocarreira/esp8266-nodemcu-simple-ldr-luximeter/> Acesso em: 30 ago. 2021.

GODOI, M. G. de; ARAÚJO, L. S. A INTERNET DAS COISAS: evolução, impactos e benefícios. **Revista Interface Tecnológica**, [S. l.], v. 16, n. 1, p. 19–30, 2019. Disponível em: <https://revista.fatectq.edu.br/index.php/interfacetecnologica/article/view/538>. Acesso em: 22 mar. 2022.

GTA UFRJ. **LDR - Light Dependent Resistor**. 2021. Disponível em: < https://www.gta.ufrj.br/grad/01_1/contador555/ldr.htm#:~:text=Constitui%C3%A7%C3%A3o%20do%20LDR%20e%20suas,exposta%20%C3%A0%20incid%C3%AAncia%20luminosa%20externa.> Acesso em: 15 jun. 2021

HACKSTER.IO. **SmartMali**. 2017. Disponível em: <https://www.hackster.io/AzureDragon/smartmali-1fbdda> . Acesso em: 28 jul. 2021

HELPDIGITAL. **Revolução Digital:** A Internet das Coisas. 2017. Disponível em:

<http://helpdigitalti.com.br/blog/revolucao-digital-internet-das-coisas/>. Acesso em: 22 mar. 2022.

KODALI, R. K.; SORATKAL, S. R. **MQTT based home automation system using ESP8266**. IEEE Region 10 Humanitarian Technology Conference 2016, R10-HTC 2016 - Proceedings, 2017. IEEE, p. 1–5, 2017.

LABORATORY, E. **Getting Started with MQTT using Mosquitto**. 2018. Disponível em: <http://embeddedlaboratory.blogspot.com/2018/01/getting-started-with-mqtt-using.html>. Acesso em: 23 jan. 2022.

LIMA, Emanuel M. Santos; NOBRE, A. Ygo Magalhães; ALENCAR de, Rômulo A. Ellery. **Automação Residencial de Baixo Custo com Arduino Mega E Ethernet Shield.** Trabalho de Conclusão de Curso. Centro Universitário Estácio do Ceará, 2016.

MINDSTORM ENGINEERING. **Nodemcu ESP8266 – Lab 10 MQ-2 Gas Sensor.** 2021. Disponível em: <https://mindstormengg.com/nodemcu-esp8266-lab-9-mq-2-gas-sensor/> . Acesso em: 19 set. 2021.

OLIVEIRA, Ricardo Rodrigues. **Uso do Microcontrolador Esp8266 para automação residencial**. 2017. Trabalho de conclusão de curso (Graduação) - Escola Politécnica,Universidade Federal do Rio de Janeiro, [*S. l.*], 2017.

PUBSUBCLIENT. Arduino Client for MQTT. [S.l.], 2019. Disponível em: <https://github.com/knolleary/pubsubclient > . Acesso em: 27 set. 2021.

RANDOM NERD TUTORIALS. **ESP32 Relay Module – Control AC Appliances (Web Server).** 2019. Disponível em: <https://randomnerdtutorials.com/esp32-relay-module-ac-web-server/>. Acesso em: 21 nov. 2021.

SILVA, Leonardo José Nascimento. **Utilizando o Home Assistant e o NodeMCU para um modelo genérico de automação moderna.** 2019. 47 f. TCC (Graduação) - Curso de Ciência da Computação, Universidade Federal de Sergipe, Centro de Ciências Exatas e Tecnologia, São Cristóvão, 2019.

SOLHA, Matheus Coelho. **Construção de módulos baseados em Internet das coisas para coleta de dados e gerenciamento de informações**. 2018. 55 f. TCC (Graduação) - Curso de Ciência da Computação, Departamento de Computação, Universidade Estadual Paulista "Júlio de Mesquita Filho" Faculdade de Ciências - Campus Bauru, Bauru, 2018.

THINGSPEAK, Criar usuário. 2021. Disponível em: <https://thingspeak.com/>. Acesso em: 25 ago. 2021.

YUAN, M. **Getting to know MQTT**. 2017. IBM Developer. Disponível em: <https://developer.ibm.com/articles/iot-mqtt-why-good-for-iot/>. Acesso em: 23 jan. 2022.

APÊNDICE A

Tem-se na Tabela 6 uma lista com a família de sensores de gás MQ

Tabela 6 - Lista de sensores de gás- família MQ

Sensor	Sensível para
MQ-2	Detecção de gases inflamáveis: GLP, Metano, Propano, Butano, Hidrogênio, Álcool,Gás Natural, outros inflamáveis e fumaça.
MQ-3	Detecção de Álcool, Etanol e fumaça.
MQ-4	Detecção de Metano, Propano e Butano.
MQ-5	Detecção de GLP e gás natural
MQ-6	Detecção de gás GLP (Gás de Cozinha), Propano, Isobutano e Gás NaturalLiquefeito
MQ-7	Detecção do gás Monóxido de Carbono
MQ-8	Detecção do gás hidrogênio
MQ-9	Detecção de Monóxido de Carbono e gases inflamáveis
MQ-131	Detecção de ozônio
MQ-135	Detecção de Gás Amônia, Óxido Nítrico, Álcool, Benzeno, Dióxido de Carbono eFumaça
MQ-136	Detecção de gás Sulfídrico H2S
MQ-137	Detecção de gás Amônia
MQ-138	Detecção de n-hexano, benzeno, NH3, álcool, fumaça, CO, etc.

Fonte: Candido, 2018

APÊNDICE B

Neste apêndice descreve-se o programa do NodeMCU utilizado nos testes finais do projeto.

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#include <Adafruit_ADS1X15.h>
Adafruit_ADS1115 ads;  /* Use this for the 16-bit version */

#define wifi_ssid "Bia"
#define wifi_password "bia123456"

#define mqtt_server "186.217.93.131"
#define mqtt_user "mqttuser"
#define mqtt_password "1234"

#define temperature_topic "sensor/temperature"
#define humidity_topic "sensor/humidity"
#define light_topic "sensor/light"
#define gas_topic "sensor/gas"

float val = 0;
float light_level = 0, sensorgas_level = 0;

//Buffer to decode MQTT messages
char message_buff[100];

long lastMsg = 0;
long lastRecu = 0;
bool debug = false;  //Display log message if True
```

```
#define DHTPIN D6
#define DHTTYPE DHT11

// Create abjects
DHT dht(DHTPIN, DHTTYPE);
WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(9600);
  pinMode(D0,OUTPUT);     //Buzzer
  pinMode(D8,OUTPUT);     //Lâmpada
  pinMode(D7,OUTPUT);     //Ventilador
  setup_wifi();           //Connect to Wifi network
  client.setServer(mqtt_server, 1883);    // Configure MQTT connexion
  client.setCallback(callback);           // callback function to execute when a
MQTT message
  dht.begin();
  ads.setGain(GAIN_ONE);        // 1x gain   +/- 4.096V  1 bit = 0.125mV
  ads.begin();
}

//Connexion WiFi
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(wifi_ssid);

  WiFi.begin(wifi_ssid, wifi_password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

```
  }

  Serial.println("");
  Serial.println("WiFi OK ");
  Serial.print("=> ESP8266 IP address: ");
  Serial.print(WiFi.localIP());
}

//Reconnexion
void reconnect() {

  while (!client.connected()) {
    Serial.print("Connecting to MQTT broker ...");
    if (client.connect("ESP8266Client", mqtt_user, mqtt_password)) {
      Serial.println("OK");
    } else {
      Serial.print("KO, error : ");
      Serial.print(client.state());
      Serial.println(" Wait 5 secondes before to retry");
      delay(5000);
    }
  }
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  long now = millis();
  // Send a message every minute DHT11
  if (now - lastMsg > 1000 * 15) {
    lastMsg = now;
```

```
  // Read humidity
  float h = dht.readHumidity();
  // Read temperature in Celcius
  float t = dht.readTemperature();

sensorgas_level = ads.readADC_SingleEnded(3);
light_level = ads.readADC_SingleEnded(0);


  // Oh, nothing to send
  if ( isnan(t) || isnan(h)) {
    Serial.println("KO, Please chez DHT sensor !");
    return;
  }

  if ( debug ) {
    Serial.print("Temperature : ");
    Serial.print(t);
    Serial.print(" | Humidity : ");
    Serial.println(h);
  }
  client.publish(temperature_topic, String(t).c_str(), true);       // Publish
temperature on temperature_topic
  client.publish(humidity_topic, String(h).c_str(), true);      // and humidity
  client.publish(light_topic, String(light_level).c_str(), true);    // Publish light
on light_topic
  client.publish(gas_topic,    String(sensorgas_level).c_str(),    true);         //
Publish gas on gas_topic
}


if (now - lastRecu > 100 ) {
  lastRecu = now;
  client.subscribe("homeassistant/switch1"); //Lâmpada
```

```
      client.subscribe("homeassistant/switch2"); //Ventilador
      client.subscribe("homeassistant/switch3"); //Buzzer
    }
  }

  // MQTT callback function
  //  D'après  http://m2mio.tumblr.com/post/30048662088/a-simple-example-arduino-mqtt-m2mio
  void callback(char* topic, byte* payload, unsigned int length) {
    String topicStr = topic;
   if (topicStr == "homeassistant/switch1") {
    int i = 0;
    if ( debug ) {
      Serial.println("Message recu =>  topic: " + String(topic));
      Serial.print(" | longueur: " + String(length,DEC));
    }
    // create character buffer with ending null terminator (string)
    for(i=0; i<length; i++) {
      message_buff[i] = payload[i];
    }
    message_buff[i] = '\0';

    String msgString = String(message_buff);
    if ( debug ) {
      Serial.println("Payload: " + msgString);
    }

    if ( msgString == "ON" ) {
      digitalWrite(D8,LOW);
    } else {
      digitalWrite(D8,HIGH);
    }
  }
   if (topicStr == "homeassistant/switch2"){
```

```
  int i = 0;
  if ( debug ) {
    Serial.println("Message recu =>  topic: " + String(topic));
    Serial.print(" | longueur: " + String(length,DEC));
  }
  // create character buffer with ending null terminator (string)
  for(i=0; i<length; i++) {
    message_buff[i] = payload[i];
  }
  message_buff[i] = '\0';

  String msgString = String(message_buff);
  if ( debug ) {
    Serial.println("Payload: " + msgString);
  }

  if ( msgString == "ON" ) {
    digitalWrite(D7,LOW);
  } else {
    digitalWrite(D7,HIGH);
  }

}

if (topicStr == "homeassistant/switch3"){
  int i = 0;
  if ( debug ) {
    Serial.println("Message recu =>  topic: " + String(topic));
    Serial.print(" | longueur: " + String(length,DEC));
  }
  // create character buffer with ending null terminator (string)
  for(i=0; i<length; i++) {
    message_buff[i] = payload[i];
  }
```

```
  message_buff[i] = '\0';

  String msgString = String(message_buff);
  if ( debug ) {
    Serial.println("Payload: " + msgString);
  }

  if ( msgString == "ON" ) {
    digitalWrite(D0,HIGH);
  } else {
    digitalWrite(D0,LOW);
  }

}
}
```

APÊNDICE C

Neste apêndice descrevem-se as configurações do Home Assistant no arquivo configuration.YAML, utilizado nos testes finais.

```
# Configure a default setup of Home Assistant (frontend, api, etc)
default_config:

# Text to speech
tts:
  - platform: google_translate

group: !include groups.yaml
automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml

mqtt:
  broker: 192.168.0.104    #if broker installed on the same computer than Home-Assistant
  port: 1883          #by default
  client_id: home-assistant-1
  keepalive: 60
  username: "mqttuser" #optional
  password: '1234' #optional;
  protocol: 3.1       #by default

sensor:
- platform: mqtt
  state_topic: "sensor/temperature"
  name: "Temperature"
```

```
  qos: 0
  unit_of_measurement: "°C"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/humidity"
  name: "Humidity"
  qos: 0
  unit_of_measurement: "percent"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/light"
  name: "Light Level"
  qos: 0
  unit_of_measurement: "lux"
  #value_template: '{{ payload }}'


- platform: mqtt
  state_topic: "sensor/gas"
  name: "CO Level"
  qos: 0
  unit_of_measurement: "ppm"
  #value_template: '{{ payload }}'

switch 1:
  platform: mqtt
  name: "Lâmpada 1"
  command_topic: "homeassistant/switch1"
  payload_on: "ON"
  payload_off: "OFF"
```

```
  optimistic: true   # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'

switch 2:
  platform: mqtt
  name: "Ventilador"
  command_topic: "homeassistant/switch2"
  payload_on: "ON"
  payload_off: "OFF"
  optimistic: true   # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'

switch 3:
  platform: mqtt
  name: "Buzzer"
  command_topic: "homeassistant/switch3"
  payload_on: "ON"
  payload_off: "OFF"
  optimistic: true   # Defina como true para manter o estado
  qos: 0
  retain: true
  value_template: '{{ value.x }}'
```