Intelligence plug: automatização da ação de acender e apagar lâmpadas através de uma página *web* usando os conceitos de Internet das Coisas

Italo Marvin Manhabosco, Michele Cristiani Barion

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Campus Hortolândia

(IFSP) 13183-250 - Hortolândia - SP - Brasil

italo.manhabosco@gmail.com, michele_barion@hotmail.com

Abstract. *The Internet of Things is an area that is increasingly gaining space in our daily life, connecting common devices with the internet that are often performed manually, such as turning on or off a light bulb. This example can be associated with a home security company, where a professional often needs to travel to the client's home to perform some action, such as closing an open door or turning off a light. Thus, the idea of the present work is to present a proposal for a web system that makes it possible to perform actions using the concepts of the Internet of Things, with low cost and easy implementation. In the course of this work, the implementation of all steps is addressed, as well as final tests that can be observed through a video available for consultation with system execution.*

Resumo. *Internet das Coisas é uma área que cada vez mais ganha espaço no nosso cotidiano interligando dispositivos comuns com a internet que muitas vezes são executados de maneira manual, como por exemplo, ligar ou desligar uma lâmpada. Esse exemplo pode ser associado a uma empresa de segurança residencial, a qual muitas vezes um profissional precisa se locomover até a casa do cliente para executar alguma ação, como fechar uma porta aberta ou apagar uma luz acesa. Assim, a ideia do presente trabalho é apresentar uma proposta de sistema web que possibilite executar ações usando os conceitos da Internet das Coisas, com baixo custo e que seja de fácil inplantação. No decorrer desse trabalho é abordado a implementação de todos os passos, além de testes finais que podem ser observados através de um vídeo disponibilizado para consulta com execução do sistema.*

1. Introdução

Com a saída das pessoas de suas residências no dia a dia, há possibilidade de deixarem ligadas as lâmpadas ou outros equipamentos em um período curto, mas também em um espaço maior de horas. Esse esquecimento pode acarretar um custo maior de energia elétrica, como também tendo possibilidade de trazer danos elétricos por alguma eventualidade que surja nesse período em que os equipamentos fiquem com a rede elétrica ativada.

Além disso, muitas pessoas estão optando em contratar empresas de segurança para assegurar uma maior proteção e preservação pessoal e patrimonial, sendo que em muitas residências, como em condomínios, há uso de câmeras para facilitar o acompanhamento em tempo real, tendo outras possibilidades como acionar alarmes à distância.

Considerando que muitas ações por essas empresas de segurança são executadas manualmente, pode ser aplicado o uso da Internet das Coisas para automatizar atividades desses profissionais. Como destaca o [Ministério das Comunicações 2021], *Internet of Things*

associa qualquer *hardware* que possa fazer a interação de uma determinada “coisa” com a Internet, tais como controlar eletrodomésticos, temperatura do piso, controlar cortinas, ventilação e iluminação”.

Desde 2021 o governo sancionou uma lei para promover incentivos e benefícios tributários à *IoT*. Segundo a Associação Brasileira de Automação Residencial e Predial (Aureside), o uso de dispositivos de IoT para casas inteligentes deve crescer 20% até 2023 [GOV.BR 2020].

Segundo [Bertoleti 2016], a automação residencial pode ser definida como um conjunto de tecnologias que ajudam na gestão e execução de tarefas domésticas cotidianas. A sua utilização tem por objetivo proporcionar um maior nível de conforto, comodidade e segurança além de um menor e mais racional consumo de energia”.

Associando empresas de segurança e Internet das Coisas, uma das tarefas que podem ser automatizadas é o gerenciamento da iluminação, caso houvesse o esquecimento por parte do cliente em deixá-la acesa.

Diante do exposto, o objetivo principal do trabalho é apresentar uma ferramenta *web* em que o usuário possa acionar as ações de uma tomada elétrica aplicando os conceitos de automação residencial e segurança residencial usando *hardware* de baixo custo. Enfatiza-se que a proposta terá a funcionalidade como um *plug* de tomada que gerencia as ações de ligar e desligar a lâmpada.

Para abordagem do objetivo exposto, o presente trabalho está organizado em oito seções, sendo: introdução, trabalhos correlatos, referencial teórico, metodologia, desenvolvimento, conclusão, relação das referências utilizadas tanto na pesquisa sobre as metodologias adotadas quanto das ferramentas que foram fundamentais para o desenvolvimento, além dos anexos contendo os códigos dos programas.

2. Trabalhos Correlatos

Os autores [Sousa e Francisco 2020], apresentam um modelo usando o conceito de Internet das Coisas com foco em automação residencial. O trabalho faz a aferição da temperatura de um determinado ambiente utilizando microcontrolador e um sensor de temperatura, para que possa ser futuramente adicionada uma funcionalidade que consiga alterar a temperatura do local e poder acionar alguma iluminação desejada.

Já os autores [Cruz e Santos Junior 2021] traz um outro modelo que é parecido ao do autor [Francisco e Sousa 2021], todavia diferenças são encontradas na aplicação, considerando que a proposta visa o controle e o monitoramento da temperatura de um protótipo de adega para vinhos através de um microcontrolador, relés e uma pastilha Peltier, diferente ao anterior que apenas aferiu a temperatura do ambiente.

[Cope 2021] é o terceiro trabalho correlato a ser apresentado, o qual traz um artigo publicado em um *site* americano, a qual ensina como publicar mensagens e assinar tópicos utilizando o protocolo MQTT, websockets e Javascript.

Assim, fazendo-se uma analogia entre a proposta deste trabalho e dos autores apresentados nessa seção, a funcionalidade futura do primeiro trabalho correlato que irá fazer o controle da iluminação, a ideia de utilizar relé junto com o NodeMCU do segundo artigo e

fazer uma conexão com um broker[1] MQTT utilizando *JavaScript* com *Websockets,* como foi proposto no terceiro artigo. Enfatiza-se que, um diferencial da proposta seja o baixo custo.

3. Referencial Teórico

Esta seção aborda a pesquisa bibliográfica realizada para gerar o embasamento teórico que fundamenta o uso de cada escolha para realização do presente trabalho.

3.1. Internet das Coisas

Como apresenta [Magrani 2018], Internet das Coisas ou *Internet of Things* (IoT) "pode ser caracterizado como um conjunto de tecnologias e protocolos associados que permitem que objetos se conectem a uma rede de comunicações e são identificados e controlados através desta conexão de rede".

A partir dessa comunicação, possibilita a interação de computadores, *smartphones*, *Smart TV* e outros dispositivos que tenham conexão com a Internet, com objetos do meio físico, tais como geladeiras, lâmpadas, aparelhos de ar-condicionado, torradeiras, ventiladores e outros [MINISTÉRIO DAS COMUNICAÇÕES 2021]. Como cita os autores [Cebrian e Freitas 2021], alguns exemplos de aplicação da IoT seria a contagem do número de passos, frequência cardíaca, preferências de temperatura e iluminação, reconhecimentos de elementos biométricos ativos (voz) e passivos (impressão digital).

3.2. NodeMCU

NodeMCU é uma plataforma para prototipação relacionada à Internet das Coisas. Utiliza a linguagem Lua e também pode ser programada na IDE do Arduino. Tem como principais características: conexão wi-fi através do módulo ESP8266 12-E, *wireless* padrão 802.11, Micro USB, GPIO com funções de PWM, 12-C, SPI, tensão de operação de 4,5 á 9v, taxa de transferência de 110460800bps, onze pinos de entrada e saída, suporta 5 conexão TCP/IP [VELOSO *et al.* 2017]. Através da série de terminais (pinos) que ela traz, podem ser conectados diferentes componentes como botões, relés, leds e sensores para automatizar praticamente qualquer tipo de processo, ambiente ou atividade.

Na Figura 1 é apresentado o mapa geral dos pinos de entrada e saída do NODE.

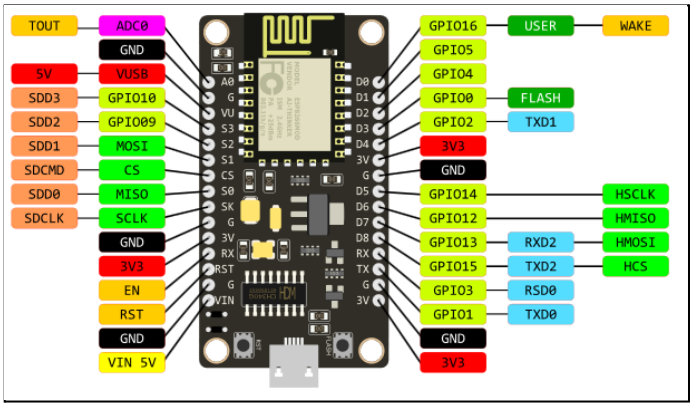

Figura 1. NodeMCU [CAPSISTEMAS 2022]

[1] Quanto ao conceito, *broker* pode ser apresentado como um servidor que recebe mensagens dos clientes e, em seguida, roteia essas mensagens para os clientes de destino relevantes. Considera-se que cliente é qualquer coisa que possa interagir com o *broker* e receber mensagens.

3.3. IDE Arduino

A IDE do Arduino é um ambiente de desenvolvimento integrado, ou seja, é um espaço onde tem tudo que precisa para programar uma placa baseada nessa plataforma escrevendo seus códigos de maneira rápida e eficiente [QUINTINO 2021].

3.4. HiveMQ

HiveMQ é um *broker* MQTT gratuito disponível em serviço de nuvem e uma plataforma de mensagens baseada em cliente projetada para a movimentação rápida, eficiente e confiável de dados de e para dispositivos IoT conectados. Ele usa o protocolo MQTT para envio instantâneo e bidirecional de dados entre seu dispositivo e seus sistemas corporativos.

A Figura 2 mostra um diagrama geral de funcionamento do *broker*, conforme apresenta o autor [HIVEMQ 2021]. Em outras palavras, o ícone representado por "*laptop*" ou "*mobile-device*" faz um "*subscribe*", ou seja, uma assinatura com um código único no "*MQTT-Broker*". Assim, deverá ser única pois outras pessoas com a mesma assinatura poderão interferir nos sensores. Ao lado esquerdo da Figura 2, o "*temperature sensor*" faz uma publicação "*publish 21°C*" no *broker,* utilizando como referência o *subscribe* feito pelo computador ou *smartphone* no primeiro passo, para que possa saber quem vai ser o destinatário. Assim, é preciso saber qual foi a assinatura de tópico utilizada por ele, e então os dispositivos recebem a mensagem direcionada pelo broker levando em conta todos esses parâmetros.

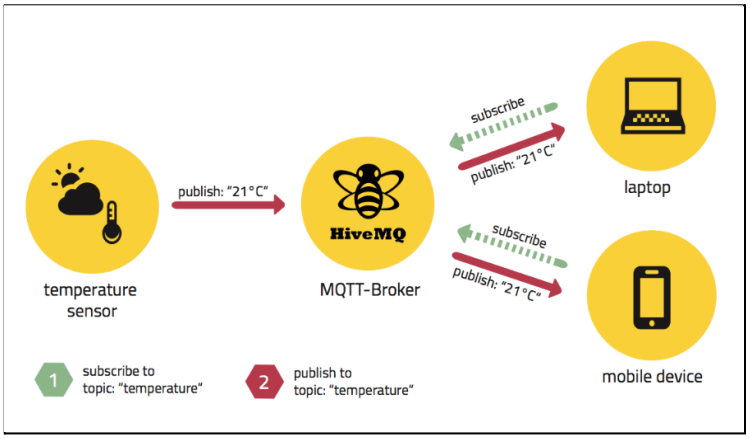

Figura 2. HiveMQ

3.5. Servidor 000webhost

É um ambiente de hospedagem gratuito patrocinado pela Hostinger, que oferece um ambiente seguro, ativo e sem custo algum. Uma das desvantagens que pode ser citada é o número de limitações sobre o número de sites hospedados [000WEBHOST 2021][2].

3.6. Sublime Text 3

É um editor HTML de código aberto, tendo uma interface com diferentes cores para facilitar a compreensão e construção dos códigos. Como apresenta [Sublime Text 2022], o programa

[2] O vídeo https://www.youtube.com/watch?v=XNy3JCzX5TY traz uma abordagem de como usar o servidor 000webhost para hospedagem de páginas.

tem sido um editor de texto muito escolhido pelos desenvolvedores por ser leve, simples e com interface agradável.

4. Metodologia

O presente trabalho segue uma metodologia de desenvolvimento com cinco etapas, sendo: (1) a escolha do *hardware* de baixo custo; (2) a procura do broker MQTT gratuito; (3) o desenvolvimento da página *web*; (4) o estabelecimento do servidor *web* e (5) a montagem do protótipo. Cada uma das etapas apresentadas seguem um desenvolvimento sequencial.

(1) Escolha do *hardware* de baixo custo: com base em pesquisa bibliográfica, tendo como fontes *sites* e livros da área de desenvolvimento dessa proposta, identificou que o melhor *hardware* associado a custo, tamanho e funcionalidades é o NodeMCU. Sendo assim, essa escolha parte das características quanto ao tamanho, processamento, uso de Wi-Fi, além do valor de aquisição. Enfatiza-se que esta etapa foi importante, pois todo o projeto parte do dispositivo escolhido. Considerando a Figura 1 que apresentou o NodeMCU, enfatiza-se que para este trabalho foram utilizados os pinos D2, D4, GND, 3.3V para comunicação com o módulo de mini relés. Adiciona-se como material o uso de uma fonte 5 volts para alimentação da placa NodeMCU e conexão com a Internet via Wi-Fi.

(2) Procura do broker MQTT gratuito: a segunda etapa parte da busca de um broker MQTT gratuito, levando em conta as vantagens e desvantagens de cada resultado da pesquisa. O HIVEMQ foi escolhido, pois apresenta menor fluxo de dados, menor número de dispositivos de outras pessoas conectadas, além da sua estabilidade. Outras opções pesquisadas eram locais, além de possuir muitas conexões ao mesmo tempo em que acabava dificultando para obter um identificador único com o *subscribe.* Essa segunda etapa do trabalho foi fundamental para que houvesse a definição de um *broker* que estivesse disponível gratuitamente, considerando o valor final da proposta.

(3) Desenvolvimento do código do *hardware* NodeMCU e da página *web*: nesta etapa foram definidas as duas IDEs para programação: para o NodeMCU foi utilizada a IDE do Arduino com o uso da linguagem Arduino Code. Para o *site* foi utilizada a IDE *Sublime Text 3,* com o uso das tecnologias/linguagens Javascript, Ajax, Jquery, HTML e CSS.

(4) Definição do servidor *web*: após buscas de *sites* que viabilizam servidores gratuitos, até mesmo considerando o custo médio final do projeto, foi definido o uso da plataforma “000webHost” por já ser utilizada nas aulas do curso por ser estável e fácil de usar. A versão da ferramenta apresentada pelo servidor é 4.7/5.

(5) Montagem do protótipo: consiste em definir um modelo que conecte a lâmpada ao relé e o relé ao NodeMCU. Já a parte virtual, considerando que a página *web* já estará ativa e hospedada, essa última fase irá integrar todos os passos anteriores para implantação e a fase de testes finais.

5. Desenvolvimento

Para apresentar uma visão geral do objetivo deste trabalho, a Figura 3 traz todas as etapas do funcionamento da proposta.

O processo se dá a partir de um computador ou smartphone (representados pelos números 1 e 2), conectado via Wi-Fi ou cabo de rede com a Internet. O usuário por meio de um desses dispositivos acessa a interface *web,* utilizando seu navegador(3), que estará hospedada no servidor *web* na nuvem (4). Em seguida ocorrerá uma comunicação com o servidor do *broker* MQTT, quem também utiliza serviço de nuvem(5) utilizando o protocolo

MQTT e JavaScript com WebSockets, a qual repassará as informações recebidas dos números (1) ou (2) ao NodeMCU (6) via protocolo MQTT diretamente sem WebSockets. Finaliza-se com os relés (7 e 8) recebendo suas respectivas instruções, fazendo com que o microcontrolador envie uma mensagem confirmando a entrada recebida. Esse processo será inverso do anterior, ou seja, envia ao broker (5), repassa para a página (3) e chega nos dispositivos (1) ou (2) a confirmação que foi recebida a mensagem pelo NodeMCU.

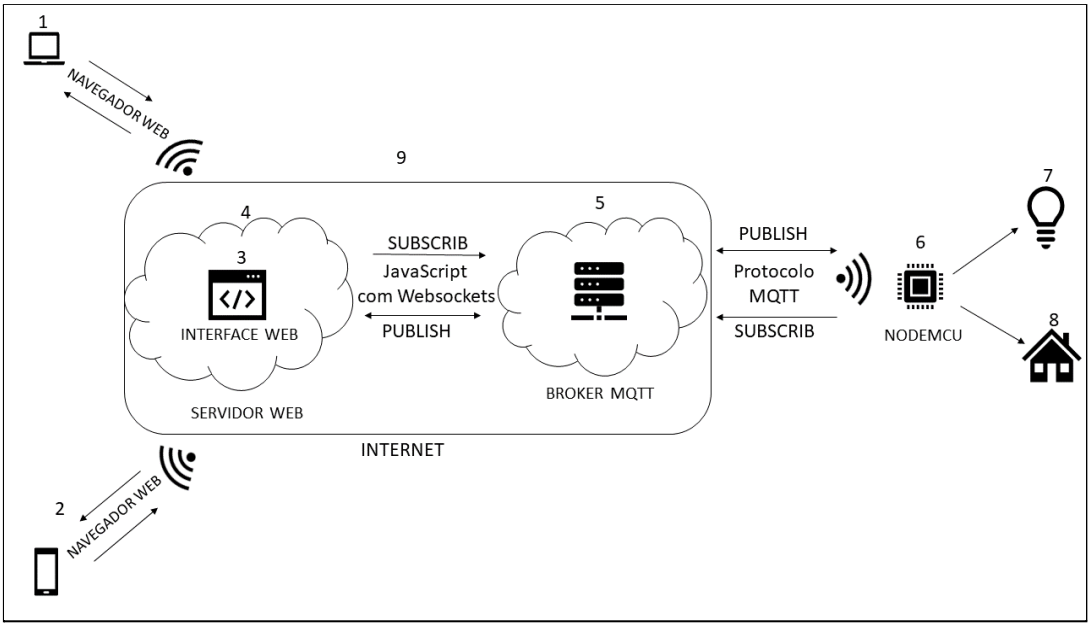

Figura 3. Esboço do projeto

Já a Figura 4 apresenta um esquema elétrico de ligação dos elementos envolvidos no projeto, a qual está numerada para explicação da mesma.

O Node (1) recebe uma tensão de 5 volts da fonte de alimentação (5) pelos cabos na cor cinza e verde, também enviando uma tensão de 3.3 volts na cor laranja e um aterramento compartilhado a fim de reforço e referência nas alimentações das placas, na cor verde, ao módulo de relés (2). O módulo recebe dois cabos de sinal, sendo D2 e D4, oriundos do microcontrolador (1) para determinar o acionamento dos relés (2). O módulo de tomadas (3) recebe alimentação de energia 127 volts (4) e o fechamento do circuito se dá pelo acionamento dos relés (2).

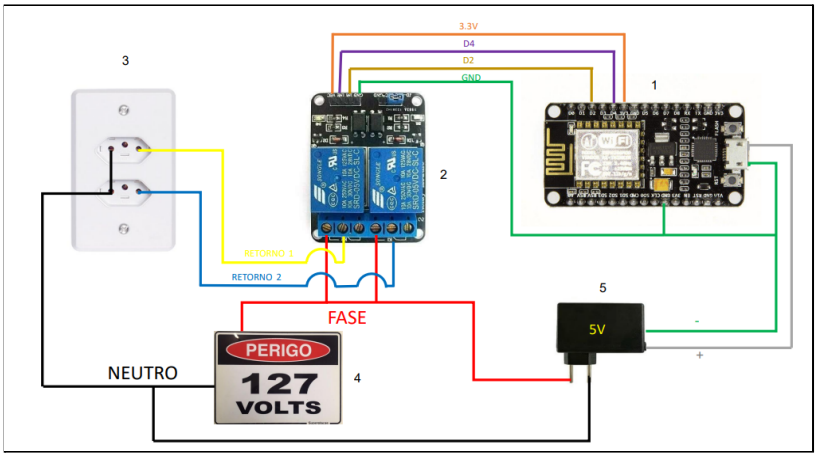

Figura 4. Estrutura elétrica do projeto

5.1. Implementação da página *web*

A Figura 5 traz um trecho do código da interface *web*, considerando a função que, ao ser requisitada por outra parte do código, realiza a comunicação entre a página e o microcontrolador NodeMCU.

```
90              function liga_lampada(){
91                  var topic = "italoMarvinEnvia";
92                  var msg = "L";
93
94                  document.getElementById("messages").innerHTML ="...";
95                  if (connected_flag==0){
96                      out_msg="<b>Não conectado ao servidor</b>"
97                      console.log(out_msg);
98                      document.getElementById("messages").innerHTML = out_msg;
99                      return false;
100                 }
101
102
103                 console.log(msg);
104
105
106                 message = new Paho.MQTT.Message(msg);
107                 message.destinationName = topic;
108                 mqtt.send(message);
109
110
111                 return false;
112             }
```

Figura 5. Trecho da implementação da função “liga_lampada”

A função “liga_lampada()” que está iniciando na linha 90 do programa contém duas variáveis local, “italoMarvinEnvia” e “L” que se encontram respectivamente nas linhas 91 e 92. Logo em seguida, a partir da linha 94 até a linha 100 ocorre um tratamento de erro, caso o servidor esteja desconectado. Se o servidor não estiver com seu status conectado, a função "*connected _flag*" que se encontra na linha 95 retornará o número 0, então a função enviará a mensagem ao console do navegador e a página *web* gerará a seguinte mensagem: "Não conectado ao servidor". Da linha 106 a 108 do código ocorre o envio da mensagem ao *broker* HiveMQ, com o valor contido na variável da linha 91, a qual informará o tópico que deve ser publicada a mensagem contida na variável da linha 92.

Para desativar a ação da função “liga_lampada”, foi criada a função “desliga_lampada”, conforme apresenta o trecho da Figura 6.

```
166
167             function desliga_lampada(){
168                 var topic = "italoMarvinEnvia";
169                 var msg = "D";
170
171
172                 document.getElementById("messages").innerHTML ="...";
173                 if (connected_flag==0){
174                     out_msg="<b>Não conectado ao servidor</b>"
175                     console.log(out_msg);
176                     document.getElementById("messages").innerHTML = out_msg;
177                     return false;
178                 }else{
179                     document.getElementById("messages").innerHTML ="topico:"+topic;
180                     document.getElementById("messages").innerHTML ="<br>Menssagem:"+msg;
181                 }
182
183
184                 console.log(msg);
185
186
187                 message = new Paho.MQTT.Message(msg);
188                 message.destinationName = topic;
189                 mqtt.send(message);
190
191
192                 return false;
193             }
194
```

Figura 6. Trecho da implementação da função “desliga_lampada”

São funcionalidades parecidas, todavia o diferencial está na mensagem enviada ao broker, ou seja, ao invés de enviar a mensagem “L” é enviada a mensagem “D” que significa para o código do microcontrolador executar a tarefa de desligar a lâmpada.

Ao abrir o endereço da página do programa em um navegador *web*, um trecho do código da página vai ser executado automaticamente. Está descrito logo abaixo na Figura 7 as funções automáticas. Iniciando a partir da linha 204 do código até a linha 210, encontram-se as variáveis mais importantes do sistema, considerando seus valores, sendo:

- *“connected_flag=0”* serve para saber se foi feita a conexão com o servidor, iniciando com valor 0 e depois, ao se conectar o valor, é alterado para 1 pela função “*onConnect*” na linha 57.

- “*mqtt*”, “reconnectTimeout” serve para auto reconectar em 2 segundos de espera.

- “*host*” é colocado o endereço do *broker* desejado.

- “*port*” é o número da porta de acordo com o *broker* escolhido.

- “*stopic*” que é o nome do tópico dado ao *subscribe* feito no *broker*.

- “*stopic2*” também é o tópico de *subscribe*, tendo como função receber as confirmações.

Pode ser observado na Figura 7, entre as linhas 214 a 225, a conexão com o *broker* utilizando as variáveis descritas acima.

```
202  <script type = "text/javascript"></script>
203      <script >
204          var connected_flag=0
205          var mqtt;
206          var reconnectTimeout = 2000;
207          var host="broker.hivemq.com";
208          var port=8000;//PORTA WEBSOCKETS
209          var stopic="italoMarvinEnvia";
210          var stopic2="italoMarvinEnviaConfirma";
211
212
213          //conectando ao servidor: host + porta
214          console.log("conectando a "+ host +" "+ port);
215          mqtt = new Paho.MQTT.Client(host,port,"clientjsaaa");
216          var options = {
217                          timeout: 3,
218                          onSuccess: onConnect,
219                          onFailure: onFailure,
220                      };
221
222          mqtt.onConnectionLost = onConnectionLost;
223          mqtt.onMessageArrived = onMessageArrived;
224          mqtt.onConnected = onConnected;
225          mqtt.connect(options);
226      </script>
```

Figura 7. Funções automáticas da interface web

O trecho do código que se encontra na Figura 8 apresenta as duas principais ações da ferramenta, sendo os botões ligar e desligar a lâmpada. Ao clicar no botão o evento “*onclick*” da linha 242 é direcionado à função “liga_lampada()” da linha 90 (implementação ilustrada na

[FIGURA 5]). Já o botão codificado na linha 246, direciona à função “desliga_lampada()” (implementação ilustrada na [FIGURA 6]).

```
239
240                 <h1>Lâmpada</h1>
241
242                 <button onclick="document.getElementById('myImage').src='on.jpg',liga_lampada()">
                    Liga</button>
243
244                 <img id="myImage" src="off.jpg" style="width:100px">
245
246                 <button onclick="document.getElementById('myImage').src='off.jpg',desliga_lampada(
                        )">Desliga</button>
247
248
```

Figura 8. Botão acionamento da lâmpada via interface web

O código completo da interface *web*, encontra-se disponível para consulta no [Anexo 1] deste trabalho.

5.2. Implementação do NodeMCU

Inicia-se a apresentação do código implementado do NodeMCU, com abordagem das variáveis que assinarão os tópicos no *broker* HiveMQ. Elas serão responsáveis em fazer a assinatura do tópico. Uma observação importante é que deve ser exatamente igual a assinatura feita pelo programa da interface *web* descrito na subseção 5.1.

```
5
6 #define TOPICO_SUBSCRIBE "italoMarvinEnvia"
7 #define TOPICO_PUBLISH   "italoMarvinEnviaConfirma"
8
```

Figura 9. Variáveis de assinatura de tópico do

Como o NodeMCU não é nativo do Arduino e foi optado por realizar sua programação pela IDE do Arduino, então deve ser feito o mapeamento dos pinos do Node para que o interpretador do Arduino possa saber exatamente qual pino está sendo referenciado no programa. A Figura 10 traz um trecho do código mostrando o mapeamento.

```
16 //defines - mapeamento de pinos do NodeMCU
17 #define D0    16
18 #define D1    5
19 #define D2    4
20 #define D3    0
21 #define D4    2
22 #define D5    14
23 #define D6    12
24 #define D7    13
25 #define D8    15
26 #define D9    3
27 #define D10   1
```

Figura 10. Mapeamento dos pinos do NodeMCU

Salienta-se a importância do microcontrolador estar conectado a rede Wi-Fi e também ao *broker*, precisando de seu endereço com a respectiva porta. A Figura 11, a partir das linhas 31 a 36 do código implementado, apresenta as variáveis que carregam esses valores.

```
29
30 // WIFI
31 const char* SSID = "dlink-632D";
32 const char* PASSWORD = "**********";
33
34 // MQTT
35 const char* BROKER_MQTT = "broker.hivemq.com"; //URL do broker MQTT
36 int BROKER_PORT = 1883; // Porta do Broker MQTT
37
```

Figura 11. Variáveis do wi-fi e do broker

A partir da linha 109 da Figura 12, há um tratamento de entrada de dados, ou seja:

- Se a mensagem recebida da página *web* for a letra "L", então o programa toma a decisão de enviar um sinal "LOW" ao pino "D4" da placa microcontroladora acionando o relé para que seja ligada a iluminação. Então na linha 112 altera a variável "EstadoSaida" para o valor 1, indicando que a lâmpada se encontra ligada.

- Se a mensagem recebida for a letra "D'', então o programa enviará ao pino "D4" o sinal "HIGH" que significa desligar, pois o módulo de relés trabalha com lógica invertida. Então a variável "EstadoSaida" volta a valer 0, informando que foi desligada a lâmpada.

```
109     if (msg.equals("L"))
110     {
111         digitalWrite(D4, LOW);
112         EstadoSaida = '1';
113     }
114
115     if (msg.equals("D"))
116     {
117         digitalWrite(D4, HIGH);
118         EstadoSaida = '0';
119     }
```

Figura 12. Função liga e desliga a lâmpada

O código completo da placa NodeMCU pode ser encontrado para consulta no [Anexo 2] deste artigo.

6. Conclusão

Para ilustrar a proposta em funcionamento, a [Figura 13] traz todo o ambiente implantado, considerando todos os passos definidos na metodologia.

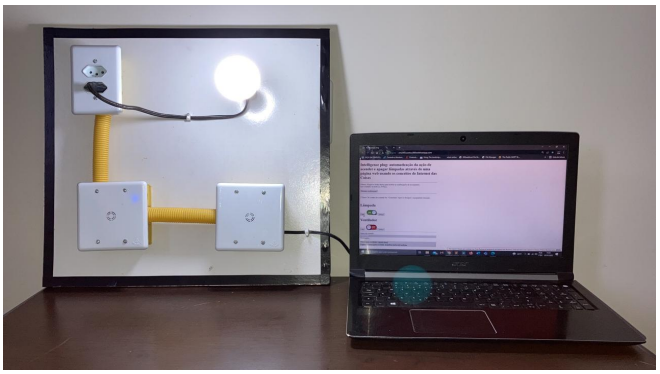

Figura 13. Ilustração da proposta implementada

Diante do objetivo proposto, o desenvolvimento foi satisfatório, considerando os testes que podem ser feitos através da ferramenta hospedada no *link* http://smarthousetcc.000webhostapp.com/. Diante do ambiente apresentado para implantação, observa-se que a ferramenta poderia auxiliar as empresas de segurança quanto ao monitoramento de um espaço físico. Também foi pensado sobre o valor ser acessível, utilizando *hardware* de baixo custo e plataformas gratuitas.

A execução da ferramenta pode ser observada através do *link* https://www.youtube.com/shorts/_HlvV0iMhsI.

Considerando o conteúdo das disciplinas do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, para este trabalho houve associação dos conceitos de rede de computadores, programação *web*, lógica de programação, engenharia de *software*, serviços de rede, além das informações compartilhadas no Grupo de Estudo e Pesquisa de Internet das Coisas (GEPIC).

7. Trabalhos futuros

Nesta seção serão apresentados alguns apontamentos observados no decorrer do desenvolvimento do trabalho, a qual podem ser definidos como futuras implementações:

- Avaliar melhorias na interface *web* ou buscar o desenvolvimento da mesma solução para dispositivos móveis.

- Adicionar mais sensores e funcionalidades, tais como temperatura, alerta de intruso, alerta de umidade e alerta de gases. Esses sensores devem estar associados a possíveis problemáticas levantadas no ambiente de desenvolvimento, tais como empresas de segurança.

Referências

000Webhost. (2021). “Hospedagem de site grátis”. https://br.000webhost.com/. [Online: acessado em 02 de janeiro de 2022].

Bertoleti, P. (2016). “Controle e Monitoramento IoT com NodeMCU e MQTT”. https://www.filipeflop.com/blog/controle-monitoramento-iot-nodemcu-e-mqtt/. [Online: acessado em 09 de fevereiro de 2021].

Brasil, E. M. (2020). “Educa Mais Brasil - Bolsas de Estudo de até 70% para Faculdades – Graduação e Pós-graduação”. https://www.educamaisbrasil.com.br/enem/lingua-portuguesa/citacao-de-site. [Online: acessado em 07 de março de 2022].

CACPNRJ. (2019). “Guia sobre NodeMCU – Uma plataforma com características singulares para o seu projeto ioT”. https://capsistema.com.br/index.php/2019/11/24/guia-sobre-nodemcu-uma-plataforma-com-caracteristicas-singulares-para-o-seu-projeto-iot/. [Online: acessado em 15 de janeiro de 2022].

Cebrian, F. F.; Freitas, C. O. A. “Da Internet das Coisas à Internet do comportamento na proteção de dados”. http://trabalhoscidhcoimbra.com/ojs/index.php/anaiscidhcoimbra/article/view/345. [Online: acessado em 10 de agosto de 2021].

Cope, S. (2021). ”Using MQTT Over WebSockets with Mosquitto. Steve's Internet Guide”. http://www.steves-internet-guide.com/mqtt-websockets/. [Online: acessado em 23 de março de 2021].

Cruz, A. D.; Santos Junior, C. R. (2021). “Monitoramento e Controle de Adega Climatizada Utilizando Placa NodeMCU”. https://hto.ifsp.edu.br/cloud/s/3nWG4MZdD7teTxY. [Online: acessado em 15 de setembro de 2021].

GND Eletrônica. (2021). “Sensor de presença”. http://www.gnd.com.br/. [Online: acessado em 22 de novembro de 2021].

GOV.BR. (2020). “Governo Federal sanciona Lei que impulsionará a Internet das Coisas no país”. https://www.gov.br/mcom/pt-br/noticias/2020/dezembro/governo-federal-sanciona-lei-que-impulsionara-a-internet-das-coisas-no-pais. [Online: acessado em 18 de novembro de 2021].

HIVEMQ. (2021). “MQTT Broker”. https://www.hivemq.com/public-mqtt-broker/. [Online: acessado em 01 de dezembro de 2021].

Magrani, E. (2018). “A internet das Coisas”. http://eduardomagrani.com/livro-internet-da-coisas-2018/. [Online: acessado em 02 de dezembro de 2021].

Ministério das Comunicações. (2021). “Internet das Coisas:: um passeio pelo futuro que já é realidade no dia a dia das pessoas”. https://www.gov.br/mcom/pt-br/noticias/2021/marco/internet-das-coisas-um-passeio-pelo-futuro-que-ja-e-real-no-dia-a-dia-das-pessoas. [Online: acessado em 18 de novembro de 2021].

NSC. (2021). “Casa inteligente: automação residencial cada vez mais acessível ao bolso dos brasileiros”. https://www.nsctotal.com.br/noticias/casa-inteligente-automacao-residencial-cada-vez-mais-acessivel-ao-bolso-dos-brasileiros. [Online: acessado em 18 de março de 2022].

Quintino, E. C. (2021). “O que é IDE Arduino?” https://www.filipeflop.com/blog/o-que-e-ide-arduino/. [Online: acessado em 07 de novembro de 2021].

Sousa, M. C.; Oliveira, R. F. (2020). “Automação Residencial utilizando uma Plataforma Open-Source”. https://hto.ifsp.edu.br/cloud/s/pPYDMAR2HHibSyM. [Online: acessado em 15 de setembro de 2021].

Sublime Text. (2022). “A sophisticated text editor for code, markup and prose”. https://www.sublimetext.com. [Online: acessado em 01 de março de 2022].

Techtudo. (2022). “Sublime Text”. https://www.techtudo.com.br/tudo-sobre/sublime-text.html#:~:text=Sublime%20Text%20%C3%A9%20um%20editor. [Online: acessado em 18 de março de 2022].

Veloso, A. F. S.; Sousa, B. A.; Braz, A. R.; Rabelo, R. A. L.; Brito, E. M. (2017). “Prototipação com nodeMCU para Internet das Coisas em Smart Cities: Uma pesquisa”. https://docplayer.com.br/65162375-Prototipacao-com-nodemcu-para-internet-das-coisas-em-smart-cities-uma-pesquisa.html. [Online: acessado em 10 de novembro de 2021].

[ANEXO 1]

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
	<html xmlns="http://www.w3.org/1999/xhtml">
		<style>
			#messages
			{
				background-color:silver;
				font-size:3;
				font-weight:bold;
				line-height:140%;
			}
			#status
			{
				background-color:silver;
				font-size:4;
				font-weight:bold;
				color:black;
				line-height:140%;
			}

			#tamanho
			{
				height: 650px;
				width: 650px;
			}
		</style>
		<head>
		  <title>Automação e segurança residencial</title>
		    <meta name="viewport" content="width=device-width, initial-scale=1.0">
			<script src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.js"
type="text/javascript"></script>
			<script type = "text/javascript"
		    src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
		    <script type = "text/javascript">
				function onConnectionLost(){
					console.log("coneção prdida");
					document.getElementById("status").innerHTML = "Conexão
perdida";
					document.getElementById("messages").innerHTML ="Conexão
perdida";
					connected_flag=0;
				}
				function onFailure(message) {
					console.log("Falhou a conexão");
					document.getElementById("messages").innerHTML = "Falha na
conexão...refazendo";
				setTimeout(MQTTconnect, reconnectTimeout);
			}
				function onMessageArrived(r_message){
					out_msg="Menssagem recebida:
"+r_message.payloadString+"<br>";
					out_msg=out_msg+"Tópico da mensagem recebida:
"+r_message.destinationName;
					console.log("mensagem recebida: ",r_message.payloadString);
					console.log(out_msg);
					document.getElementById("messages").innerHTML =out_msg;
```

```
}
function onConnected(recon,url){
	console.log(" in onConnected " +reconn);
}
function onConnect() {
  // Once a connection has been made, make a subscription and send a
message.
	document.getElementById("messages").innerHTML ="Conectado
em: "+host +"<br>na porta:"+port;
	connected_flag=1
	document.getElementById("status").innerHTML = "Conectado";
	console.log("on Connect "+connected_flag);
 }
function sub_topics(){
	document.getElementById("messages").innerHTML ="...";
	if (connected_flag==0){
		out_msg="<b>Não conectado ao servidor, portanto não
posso fazer subscribe</b>"
		console.log(out_msg);
		document.getElementById("messages").innerHTML =
out_msg;
		return false;
	}else {
		out_msg="<b> SUBSCRIBE Autorizado!!!</b>"
		console.log(out_msg);
		document.getElementById("messages").innerHTML =
out_msg;
	}
	var stopic= "italoMarvinEnvia"
	var stopic2="italoMarvinEnviaConfirma"
	console.log("Fazendo SUBSCRIBE no"+stopic);
	console.log("Fazendo SUBSCRIBE no"+stopic2);
	mqtt.subscribe(stopic);
	mqtt.subscribe(stopic2);
	return false;
}
function liga_lampada(){
	var topic = "italoMarvinEnvia";
	var msg = "L";
	document.getElementById("messages").innerHTML ="...";
	if (connected_flag==0){
		out_msg="<b>Não conectado ao servidor</b>"
		console.log(out_msg);
		document.getElementById("messages").innerHTML =
out_msg;
		return false;
	}
	console.log(msg);
	message = new Paho.MQTT.Message(msg);
	message.destinationName = topic;
	mqtt.send(message);
	return false;
}
function liga_ventilador(){
	var topic = "italoMarvinEnvia";
	var msg = "on";
	document.getElementById("messages").innerHTML ="...";
	if (connected_flag==0){
		out_msg="<b>Não conectado ao servidor</b>"
```

```
                                        console.log(out_msg);
                                        document.getElementById("messages").innerHTML =
out_msg;
                                        return false;
                                }
                                console.log(msg);
                                message = new Paho.MQTT.Message(msg);
                                message.destinationName = topic;
                                mqtt.send(message);
                                return false;
                        }
                        function desliga_ventilador(){
                                var topic = "italoMarvinEnvia";
                                var msg = "off";
                                document.getElementById("messages").innerHTML ="...";
                                if (connected_flag==0){
                                        out_msg="<b>Não conectado ao servidor</b>"
                                        console.log(out_msg);
                                        document.getElementById("messages").innerHTML =
out_msg;
                                        return false;
                                }
                                console.log(msg);
                                message = new Paho.MQTT.Message(msg);
                                message.destinationName = topic;
                                mqtt.send(message);
                                return false;
                        }
                        function desliga_lampada(){
                                var topic = "italoMarvinEnvia";
                                var msg = "D";
                                document.getElementById("messages").innerHTML ="...";
                                if (connected_flag==0){
                                        out_msg="<b>Não conectado ao servidor</b>"
                                        console.log(out_msg);
                                        document.getElementById("messages").innerHTML =
out_msg;
                                        return false;
                                }else{
                                        document.getElementById("messages").innerHTML
="topico:"+topic;
                                        document.getElementById("messages").innerHTML
="<br>Menssagem:"+msg;
                                }
                                console.log(msg);
                                message = new Paho.MQTT.Message(msg);
                                message.destinationName = topic;
                                mqtt.send(message);
                                return false;
                        }
                  </script>
                </head>
                <body id="tamanho">
                  <h1>Automação e segurança residencial com Internet das Coisas</h1>
                        <script type = "text/javascript"></script>
                                <script >
                                        var connected_flag=0
                                                var mqtt;
                                           var reconnectTimeout = 2000;
```

```
                                        var host="broker.hivemq.com";
                                        var port=8000;//PORTA WEBSOCKETS
                                        var stopic="italoMarvinEnvia";
                                        var stopic2="italoMarvinEnviaConfirma";
                                        //conectando ao servidor: host + porta
                                        console.log("conectando a "+ host +" "+ port);
                                        mqtt = new Paho.MQTT.Client(host,port,"clientjsaaa");
                                        var options = {
                                                timeout: 3,
                                                onSuccess: onConnect,
                                                onFailure: onFailure,
                                        };
                                        mqtt.onConnectionLost = onConnectionLost;
                                        mqtt.onMessageArrived = onMessageArrived;
                                        mqtt.onConnected = onConnected;
                                        mqtt.connect(options);
                                </script>
                                        <hr>
                                        1ºPasso: Clique no botão abaixo para receber as
confirmações de acionamento,<br>
                                        caso contrário vá direto ao 2º Passo.
                                        <br><br>
                                        <input type="button" value="Receber confirmaçoes"
onclick="sub_topics()">
                                        <hr><br>
                                        2º Passo: Se o status de conexão for: "Conectado" ligue ou
desligue o equipamento desejado.
                                        <br><br>
                                        <h1>Lâmpada</h1>
                                        <button
onclick="document.getElementById('myImage').src='on.jpg',liga_lampada()">Liga</button>
                                        <img id="myImage" src="off.jpg" style="width:100px">
                                        <button
onclick="document.getElementById('myImage').src='off.jpg',desliga_lampada()">Desliga</button>
                                        <h1>Ventilador</h1>
                                        <button
onclick="document.getElementById('myImage2').src='on.jpg',liga_ventilador()">Liga</button>
                                        <img id="myImage2" src="off.jpg" style="width:100px">
                                        <button
onclick="document.getElementById('myImage2').src='off.jpg',desliga_ventilador()">Desliga</button>
                                        <hr>
                                Status de conexão:
                                <div id="status"> </div>
                                Mensagens:
                                <p id="messages"></p>
                </body>
        </html>
```

[ANEXO 2]

```
#include <ESP8266WiFi.h> // Importa a Biblioteca ESP8266WiFi
#include <PubSubClient.h> // Importa a Biblioteca PubSubClient

#define TOPICO_SUBSCRIBE "italoMarvinEnvia"
#define TOPICO_PUBLISH   "italoMarvinEnviaConfirma"

#define ID_MQTT  "HomeAut"     //id mqtt (para identificação de sessão)
                        //IMPORTANTE: este deve ser único no broker (ou seja,
                        //            se um client MQTT tentar entrar com o mesmo
                        //            id de outro já conectado ao broker, o broker
                        //            irá fechar a conexão de um deles).

//defines - mapeamento de pinos do NodeMCU
#define D0    16
#define D1    5
#define D2    4
#define D3    0
#define D4    2
#define D5    14
#define D6    12
#define D7    13
#define D8    15
#define D9    3
#define D10   1

// WIFI
const char* SSID = "dlink-632D";
const char* PASSWORD = "mnvca09382";

// MQTT
const char* BROKER_MQTT = "broker.hivemq.com"; //URL do broker MQTT
int BROKER_PORT = 1883; // Porta do Broker MQTT

//Variáveis e objetos globais
WiFiClient espClient; // Cria o objeto espClient
PubSubClient MQTT(espClient); // Instancia o Cliente MQTT passando o objeto espClient
char EstadoSaida = '0';  //variável que armazena o estado atual da saída da lâmpada
char EstadoSaidaFun = '0'; //  variável que armazena o estado atual da saída do ventilador
//Prototypes
void initSerial();
void initWiFi();
void initMQTT();
void reconectWiFi();
void mqtt_callback(char* topic, byte* payload, unsigned int length);
void VerificaConexoesWiFIEMQTT(void);
void InitOutput(void);

// Implementação das funções.
void setup()
{
   //inicializações:
   InitOutput();
   initSerial();
   initWiFi();
   initMQTT();
```

```
}

//Função que inicializa comunicação serial com baudrate em 115200.
void initSerial()
{
   Serial.begin(115200);
}

//Função que inicializa e conecta-se na rede WI-FI desejada.
void initWiFi()
{
   delay(10);
   Serial.println("------Conexao WI-FI------");
   Serial.print("Conectando-se na rede: ");
   Serial.println(SSID);
   Serial.println("Aguarde");
   reconectWiFi();
}

//Função que inicializa parâmetros de conexão MQTT(endereço do broker, porta e seta função de callback).
void initMQTT()
{
   MQTT.setServer(BROKER_MQTT, BROKER_PORT);   //informa qual broker e porta deve ser conectado
   MQTT.setCallback(mqtt_callback);            //atribui função de callback (função chamada quando qualquer
informação de um dos tópicos subescritos chega)
}
//Função: função de callback
//        esta função é chamada toda vez que uma informação de
//        um dos tópicos subescritos chega).
void mqtt_callback(char* topic, byte* payload, unsigned int length)
{
   String msg;
   //obtem a string do payload recebido
   for(int i = 0; i < length; i++)
   {
      char c = (char)payload[i];
      msg += c;
   }
   //toma ação dependendo da string recebida:
   //verifica se deve colocar nivel alto de tensão na saída D2 OU D4:
   //IMPORTANTE: o relé é acionado com lógica invertida (ou seja,
   //enviar HIGH para o output faz o relé DESLIGAR / enviar LOW faz o relé LIGAR).
   if (msg.equals("L"))
   {
      digitalWrite(D4, LOW);
      EstadoSaida = '1';
   }
   if (msg.equals("D"))
   {
      digitalWrite(D4, HIGH);
      EstadoSaida = '0';
   }
    if (msg.equals("on"))
   {
      digitalWrite(D2, LOW);
      EstadoSaidaFun = '1';
   }
    if (msg.equals("off"))
   {
```

```
      digitalWrite(D2, HIGH);
      EstadoSaidaFun = '0';
   }
}

//Função que reconecta-se ao broker MQTT (caso ainda não esteja conectado ou em caso de a conexão cair)
//       em caso de sucesso na conexão ou reconexão, o subscribe dos tópicos é refeito.
void reconnectMQTT()
{
   while (!MQTT.connected())
   {
      Serial.print("* Tentando se conectar ao Broker MQTT: ");
      Serial.println(BROKER_MQTT);
      if (MQTT.connect(ID_MQTT))
      {
         Serial.println("Conectado com sucesso ao broker MQTT!");
         MQTT.subscribe(TOPICO_SUBSCRIBE);
      }
      else
      {
         Serial.println("Falha ao reconectar no broker.");
         Serial.println("Havera nova tentatica de conexao em 2s");
         delay(2000);
      }
   }
}
//Função que reconecta-se ao WiFi.
void reconectWiFi()
{
   //se já está conectado a rede WI-FI, nada é feito.
   //Caso contrário, são efetuadas tentativas de conexão.
   if (WiFi.status() == WL_CONNECTED)
      return;
   WiFi.begin(SSID, PASSWORD); // Conecta na rede WI-FI.
   while (WiFi.status() != WL_CONNECTED)
   {
      delay(100);
      Serial.print(".");
   }
   Serial.print("Conectado com sucesso na rede ");
   Serial.print(SSID);
   Serial.println("IP obtido: ");
   Serial.println(WiFi.localIP());
}
//Função que verifica o estado das conexões WiFI e broker MQTT.
//       Em caso de desconexão (qualquer uma das duas), a conexão
//       é refeita.
void VerificaConexoesWiFIEMQTT(void)
{
   if (!MQTT.connected())
      reconnectMQTT(); //se não há conexão com o Broker, a conexão é refeita.
      reconectWiFi(); //se não há conexão com o WiFI, a conexão é refeita.
}/Função que envia ao Broker o estado atual do output
void EnviaEstadoOutputMQTT(void)
{
   if (EstadoSaida == '0')
     MQTT.publish(TOPICO_PUBLISH, "Desligado lamp");
     delay(1000);
   if  (EstadoSaida == '1')
```

```
      MQTT.publish(TOPICO_PUBLISH, "Ligado lamp");
      delay(1000);
     if (EstadoSaidaFun == '0')
      MQTT.publish(TOPICO_PUBLISH, "Desligado fun");
      delay(1000);
     if (EstadoSaidaFun == '1')
      MQTT.publish(TOPICO_PUBLISH, "Ligado fun");
      delay(1000);
    Serial.println("- Estado da saida enviado ao broker!");
    delay(2000);
}
//Função que inicializa o output em nível lógico baixo.
void InitOutput(void)
{
    //IMPORTANTE: o Led já contido na placa é acionado com lógica invertida (ou seja,
    //enviar HIGH para o output faz o Led apagar / enviar LOW faz o Led acender).
    pinMode(D4, OUTPUT);
    digitalWrite(D4, HIGH);
    pinMode(D2,OUTPUT);
    digitalWrite(D2,HIGH);
}
//programa principal
void loop()
{
    //garante funcionamento das conexões WiFi e ao broker MQTT.
    VerificaConexoesWiFIEMQTT();
    //envia o status de todos os outputs para o Broker no protocolo esperado.
    EnviaEstadoOutputMQTT();
    //keep-alive da comunicação com broker MQTT.
    MQTT.loop();
}
```