



CASA CONECTADA, USO CONSCIENTE DA ENERGIA COM IoT.

Patricia Yumi Takeyama¹ Professor Wilian França Costa²

Universidade Presbiteriana Mackenzie (UPM)
Rua da Consolação, 930 Consolação, São Paulo - SP, 01302-907 – Brasil

yumi.takeyama@gmail.com

Abstract. *The Internet of Things (IoT) has transformed people's daily lives by offering them a way of communicating and interacting with machines, and the alignment with industry 4.0 has contributed to the development and expansion of this technology. It is estimated that by the year 2025 more than 1 trillion sensors will be connected to the internet. This number, although impressive, is made possible by the economic feasibility of connecting everything to the internet, in addition to intelligent sensors having a strong competitive market, thus reducing the cost to the final consumer.*

In current times where energy is an expensive product, looking for intelligent ways to reduce costs and preserve the environment with clean energy solutions is a step that leads society to conscious consumption.

keywords: *Internet of Things (IoT), Arduino, 4.0 Industry.*

Resumo. *A Internet das Coisas (IoT) transformou o cotidiano das pessoas oferecendo a elas uma forma de comunicação e interação com as máquinas, e o alinhamento com a industria 4.0 contribuiu para o desenvolvimento e expansão dessa tecnologia. Estima-se que até o ano 2025 mais de 1 trilhão de sensores estejam conectados na internet. Esse numero apesar de impressionante torna-se possível pela viabilidade econômica de conectar tudo a internet, além disso os sensores inteligentes tem um forte mercado competitivo reduzindo assim o custo para o consumidor final.*

Em tempos atuais onde a energia é um produto caro buscar formas inteligentes de reduzir custo e preservar o meio ambiente com soluções de energia limpa é um passo que leva a sociedade a um consumo consciente.

palavras-chaves: Internet das Coisas, Arduino, Indústria 4.0, MQTT

¹ Graduanda no Curso EAD de Análise e Desenvolvimento de Sistemas.

² Professor da disciplina de Objetos Inteligentes Conectados.

1. Introdução

O termo Internet of Things (IoT), traduzido para português como Internet das Coisas foi criado em 1999 por Kevin Ashton durante seu trabalho na *Procter & Gamble*. O objetivo de Ashton era atrair a atenção da alta administração para uma nova tecnologia chamada *Radio Frequency Identification* (RFID) ou identificação por rádio frequência, na época houve poucos investidores interessados.

O início da popularização da IoT foi a partir de 2010, e desde então grandes empresas passaram a investir em larga escala. Hoje a IoT não se limita ao conceito primário de rastreio de objeto, ela se refere a toda troca de informações, seja se cunho informativo, automatizado ou inteligente. (Stevan, 2018, p. 20)

Além disso o avanço da microeletrônica permitiu o desenvolvimento de microcomputadores, a VLSI (*Very Large Scale Integration*), integração em muito larga escala, caracteriza uma classe de dispositivos eletrônicos capazes de armazenar em um único invólucro milhares de pequenos componentes. Este dispositivo chamado de chip é usado na estrutura dos computadores modernos. (Monteiro, 2015, p. 23)

A IoT tem uma expressiva participação na Indústria 4.0, o uso da tecnologia sensorial, eletrodomésticos inteligentes, sustentabilidade, gerenciamento de energia é apenas uma parte do universo IoT, dificilmente o consumidor achará algo que não esteja conectado a internet.

A proposta deste trabalho será construir um protótipo de casa conectada onde o usuário poderá monitorar as luzes da casa por meio de um celular conectado a internet.

A plataforma utilizada será a Arduino, dentro desta plataforma será feita a configuração da placa e dos pinos de LED, bem como a comunicação da placa com o protótipo.

Para o controle das lâmpadas o usuário deverá baixar a aplicação MQTT DASH disponível apenas para dispositivos android.

Será utilizado o protocolo de comunicação *Message Queue Telemetry Transport* (MQTT), definido como um protocolo leve, seguro e de fácil implementação.

2. Materiais e Métodos

UNIVERSIDADE PRESBITERIANA MACKENZIE

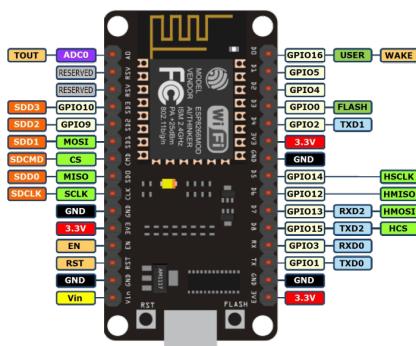
Faculdade de Computação e Informática

2.1 Definição dos materiais.

Este projeto está utilizando como referência as instruções do site *Curto Circuito*, já as descrições dos materiais utilizados e seus métodos estarão elencados a seguir.

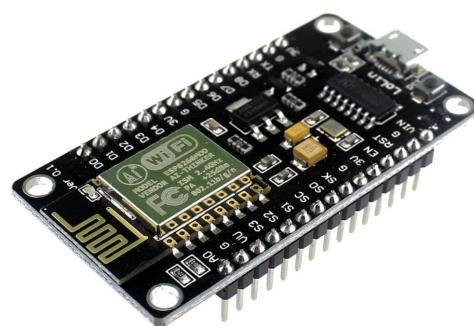
2.1.1 Placa NodeMCU V3 - ESP8266 - CH340

Figura 1: esquemático dos pinos do módulo



Fonte: site eletrogate

Figura 2: Placa ESP8266, NodeMCU, V3



Fonte: site eletrogate

Tabela 1: Datasheet da placa ESP8266

ESP 8266	Descrição
Padrões wireless	IEEE 802.11b, IEEE 802.11g, IEEE 802.11n
Faixa de frequência	2.4GHz
Taxa de transmissão	110 à 460 Mbps
Interface USB	CH340
Interface	Serial UART (Tx / Rx)
Segurança	WEP / WPA / TKIP / AES
Alimentação	4,0 à 9,0 VDC (conector Micro USB)
Tensão Lógica	3,3 VDC
Consumo	Min 70 mA (Standby) e Máx 220 mA (802.11b, CCK 1Mbps, Pout=+19.5dBm)
Conversor A/D	10 bits ADC e Vin 0 à 1 VDC
GPIO	11 portas
Dimensões	60 x 31 x 13 mm (C x L x A);
Peso	10 g

Fonte: Curto Circuito (adaptado)

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

2.1.2 Pino de Led

Tabela 2: Datasheet pino de led

Diâmetro	5 mm
Cor	Branco
Tensão	2 v
Corrente	20 mA

Fonte: Baú da eletrônica (adaptado)

Figura 3



Fonte: Usinainfo

2.1.3 Resistor 220 Ohm/s

Tabela 3: datasheet resistor

Resistência	220R
Potencia	1/4 W
Tolerância	5%

Fonte: Baú da eletrônica (adaptado)

Figura 4



Fonte: Usinainfo

2.1.4 Jumper MM

Tabela 4: datasheet jumper mm

Secção do fio condutor	AWG 24
Comprimento do fio	10 cm
Largura do conector	2,54 mm

Fonte: Filipeflop (adaptado)

Figura 5



Fonte: Usinainfo

2.1.5 Módulo sensor DHT22

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Tabela 5: datasheet sensor umidade e temperatura

Modelo	AM2302
Tensão de operação	3-5VDC (5,5VDC máximo)
Faixa de medição de umidade	0 a 100% UR
Faixa de medição de temperatura	-40° a +80°C
Corrente máxima durante uso	2,5mA
Corrente em stand by	100uA a 150 uA
Precisão de umidade de medição	± 0,5 °C
Resolução	0,1
Tempo de resposta	2s
Dimensões	25 x 15 7mm (sem terminais)

Figura 6

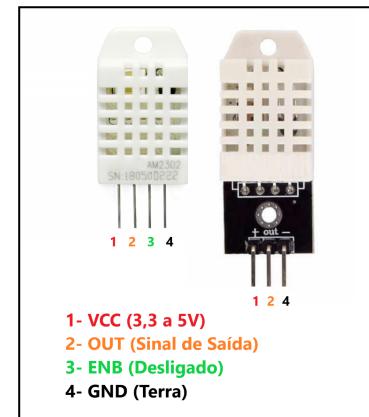


Figura 5 (Fonte Site Curto Circuito)

Fonte: Portal Vida de Silicio (adaptado)

2.2 Definição do Método

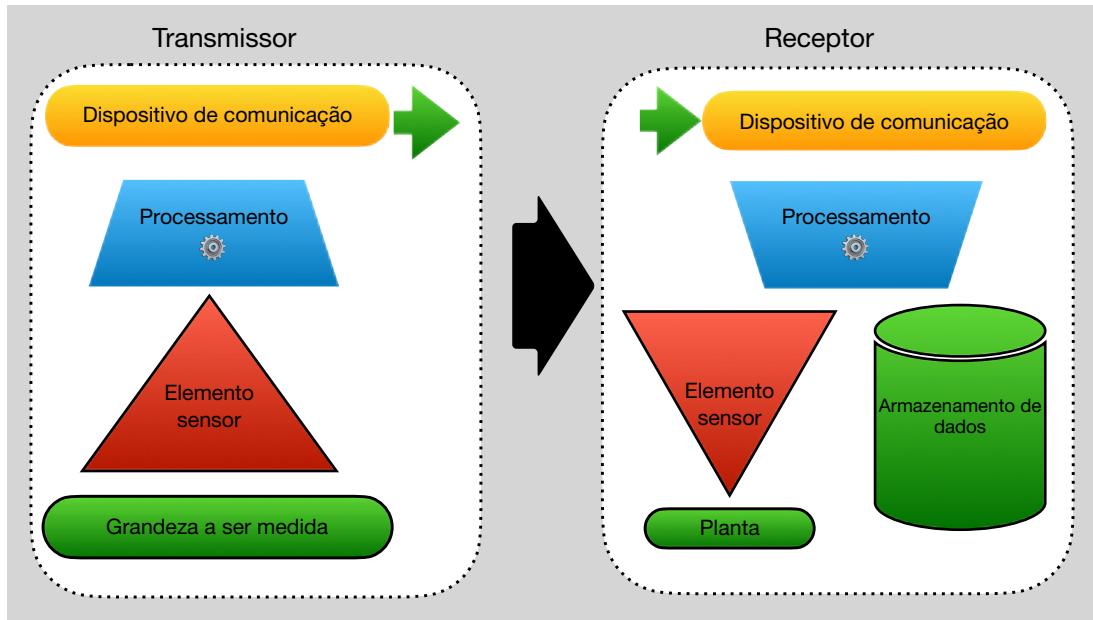
2.2.1 Funcionamento da IoT

Para que uma aplicação IoT funcione são necessários três elementos fundamentais:

- elemento sensor/transdutor:** ele é responsável pela identificação das grandezas físicas e conversão em sinais elétricos onde será adicionado a um circuito eletrônico.
- dispositivo de comunicação:** é responsável por receber o dado, adicionar no protocolo e transmiti-lo
- sistema microcontrolado (dedicado):** responsável por fazer o gerenciamento de dados que vem do sensor.

Com isso é formado um dispositivo de transmissão e recepção de uma IoT. (Stevan, 2018, p.109).

Figura 7

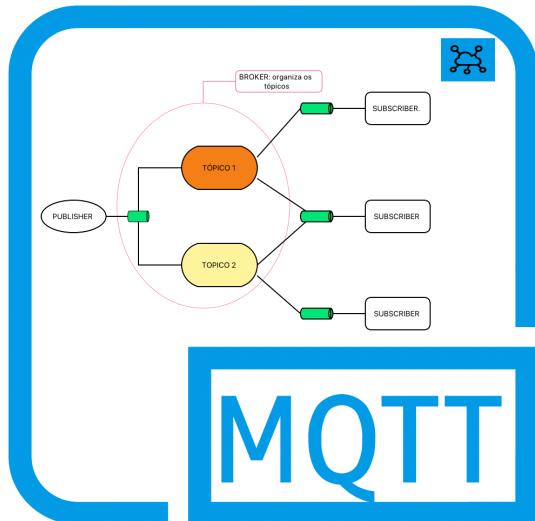


Esboço básico de um módulo de comunicação IoT. (Stevan, 2018, p. 113, Adaptado)

2.2.2 Protocolo MQTT

O MQTT (*Message Queue Telemetry Transport*) é um protocolo de rede para troca de mensagens que foi desenvolvido pela IBM e a Eurotech no final da década de 90. Este protocolo se tornou padrão nas aplicações IoT, a fácil implementação e a leveza é ideal para a comunicação remota entre dispositivos.

Figura 8



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Na figura 8 temos a representação do *Publish* (Publicador) e *Subscriber* (Subscrito), sendo que o mensageiro responsável por encaminhar e receber os dados é chamado de *Broker*, podendo esse ser hospedado na nuvem ou local (Monitoramento e Controle por aplicativo MQTT, Site Curto Circuito, 2020).

2.2.2.1 Broker

O Broker atua como um servidor intermediário da informação recebendo os dados enviados pelo sensores, tratando esses dados e encaminhando. É possível existir mais de um Broker em um sistema compartilhando os dados entre si, de acordo com o Cliente e seus requisitos.

2.2.2.2 Cliente

O Cliente está ligado a duas áreas de atuação: postagem e recebimento, onde ele tem a opção de escolher trabalhar apenas com uma delas ou com ambas, seja qual for a escolha o *Broker* deve ser o intermediador.

As tipos de mensagens principais são:

- *Connect*: tentar criar uma conexão com o Broker, aguarda a conexão ser estabelecida e começa a ouvir as mensagens.
- *Disconnect*: espera o cliente finalizar a ação, finaliza a conexão TCP/IP e para de ouvir as mensagens.
- *Publish*: retorna a informação que foi enviada ao cliente MQTT

2.2.2.3 Broker x Cliente

As informações recebidas pelo *Broker* são organizadas hierarquicamente de acordo com seus Tópicos e passadas a adiante. Isso quer dizer que cada dado que for captado e enviado para o *Broker* será parte apenas de um Tópico.

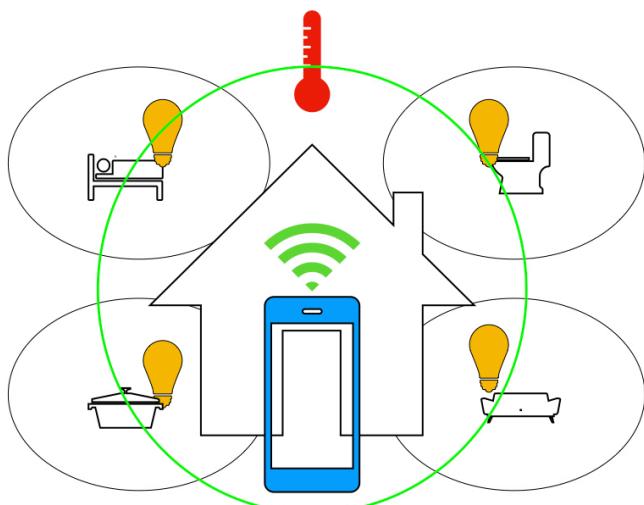
2.2.3 Conexão com o protocolo MQTT

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

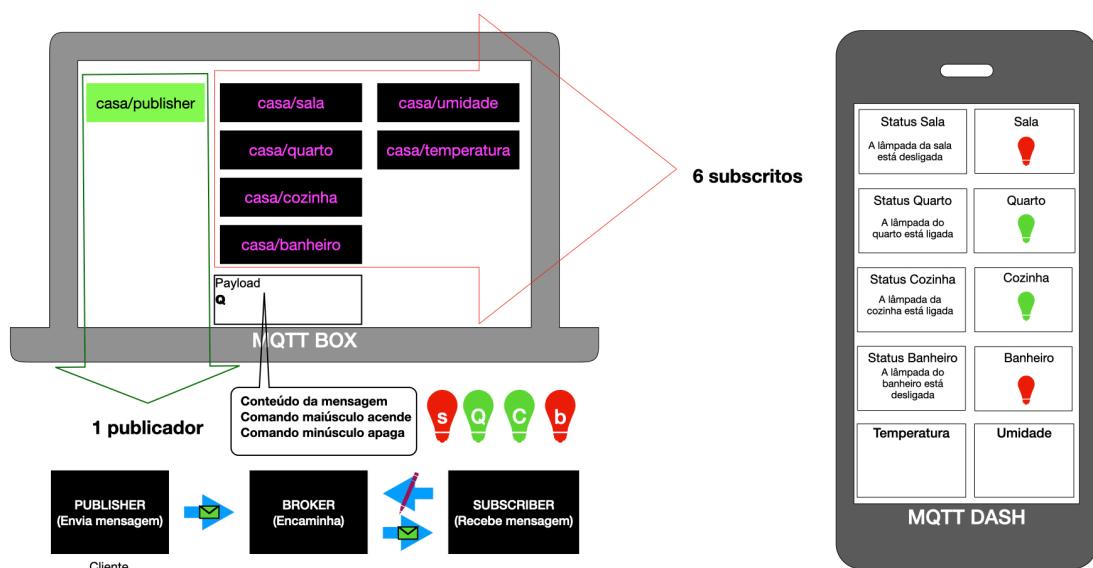
Na Figura 9 e 10 está representando uma casa com 4 cômodos onde cada lâmpada representa um Led e a sua comunicação se dará por meio do protocolo MQTT. Neste mesmo projeto será feita uma medição da temperatura e umidade da casa.

Figura 9



Fonte: elaborado pelo autor

Figura 10



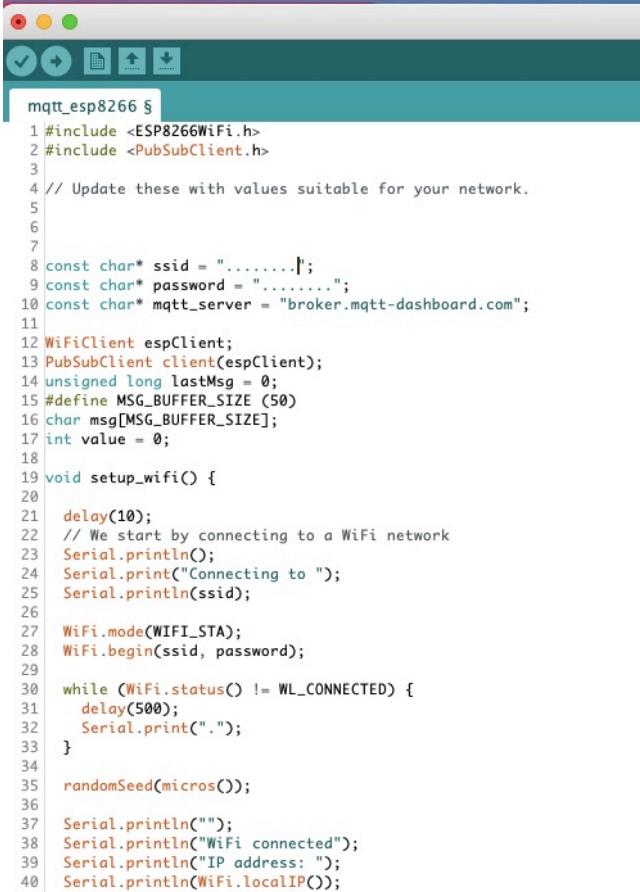
Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Abaixo, na figura 11, a codificação padrão para conectar uma máquina a outra. Esta codificação será inserida e gravada na placa.

Figura 11



```
mqtt_esp8266 §
1 #include <ESP8266WiFi.h>
2 #include <PubSubClient.h>
3
4 // Update these with values suitable for your network.
5
6
7
8 const char* ssid = ".....";
9 const char* password = ".....";
10 const char* mqtt_server = "broker.mqtt-dashboard.com";
11
12 WiFiClient espClient;
13 PubSubClient client(espClient);
14 unsigned long lastMsg = 0;
15 #define MSG_BUFFER_SIZE (50)
16 char msg[MSG_BUFFER_SIZE];
17 int value = 0;
18
19 void setup_wifi() {
20
21   delay(10);
22   // We start by connecting to a WiFi network
23   Serial.println();
24   Serial.print("Connecting to ");
25   Serial.println(ssid);
26
27   WiFi.mode(WIFI_STA);
28   WiFi.begin(ssid, password);
29
30   while (WiFi.status() != WL_CONNECTED) {
31     delay(500);
32     Serial.print(".");
33   }
34
35   randomSeed(micros());
36
37   Serial.println("");
38   Serial.println("WiFi connected");
39   Serial.println("IP address: ");
40   Serial.println(WiFi.localIP());
```

Fonte: captura tela Arduino IDE

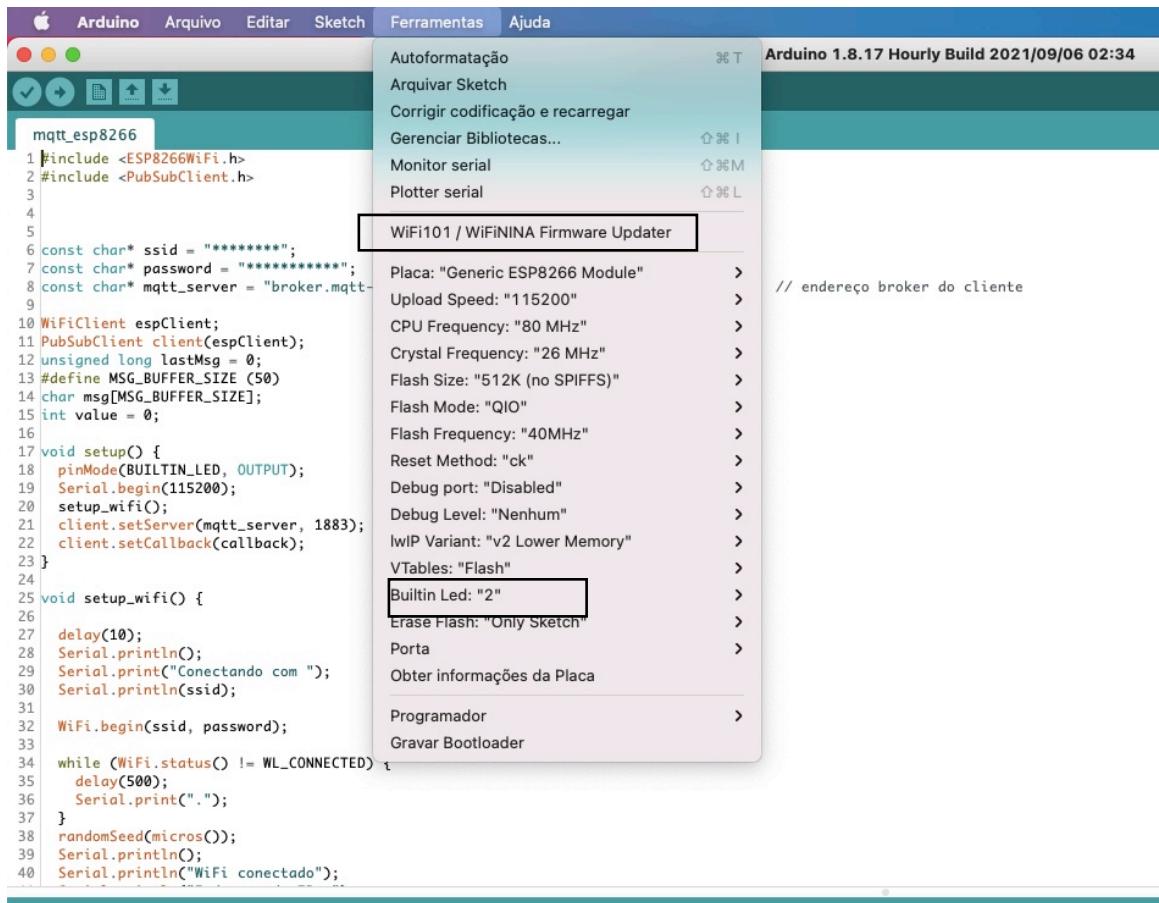
Após a inserção do código, figura 12, a placa deverá ser conectada no cabo USB e em seguida deverá ser selecionado o modelo e porta compatível.

O número da porta pode ser diferente de um computador para outro, para este caso foi selecionada a porta COM3.

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

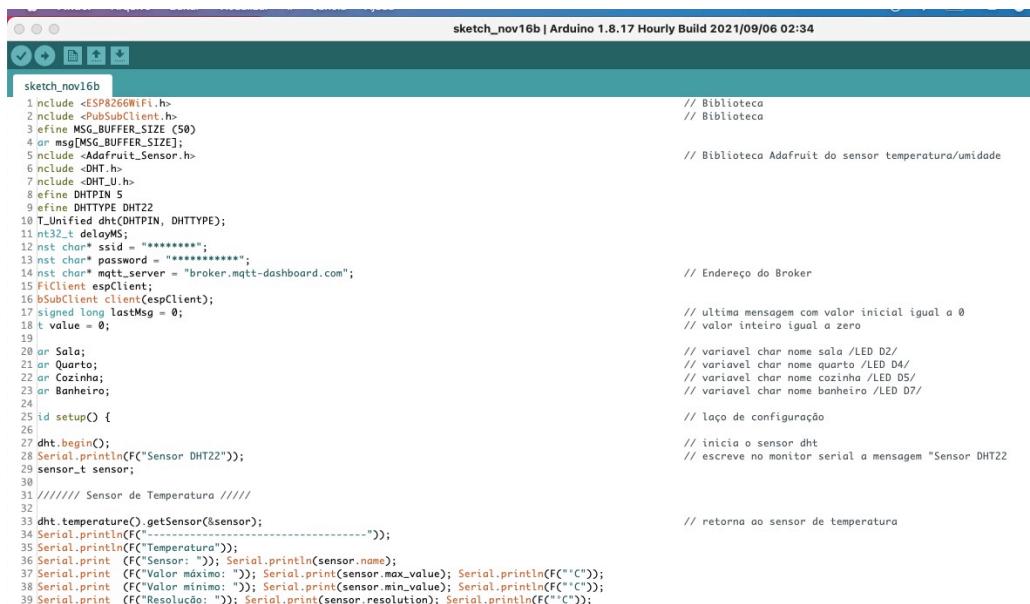
Figura 12



Fonte: captura de tela Arduino IDE

Nas figuras 13 a 18 estão as imagens capturadas da configuração dos pinos da sala, quarto, cozinha e banheiro.

Figura 13



Fonte: captura de tela Arduino IDE

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Figura 14

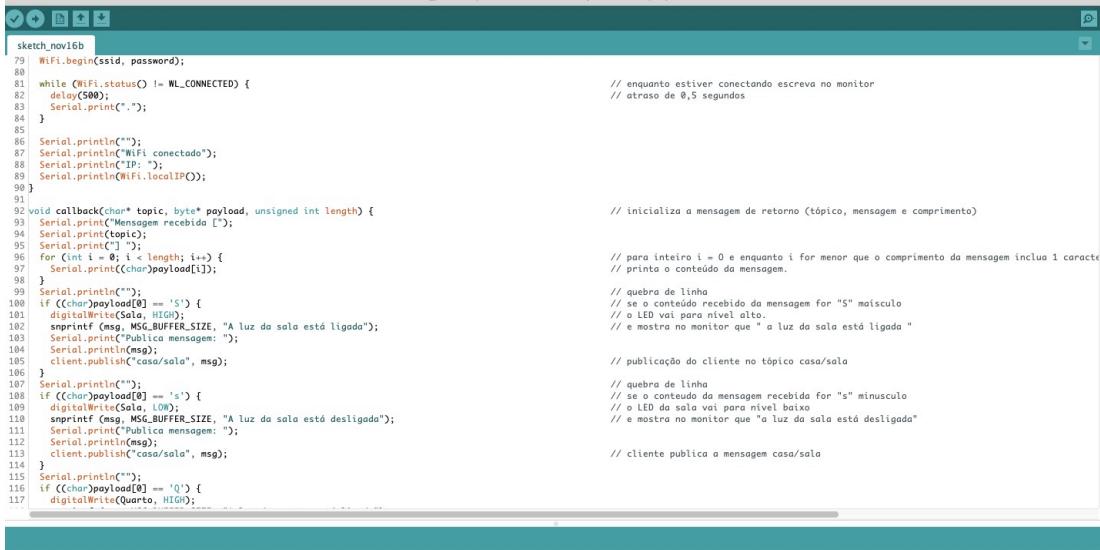


```
sketch_nov16b | Arduino 1.8.17 Hourly Build 2021/09/06 02:34

sketch_nov16b
39 Serial.print(F("Resolução: ")); Serial.print(sensor.resolution); Serial.println(F("%"));
40 Serial.println(F("-----"));
41
42 //---- Sensor de Umidade ----
43
44 dht.humidity()> getSensor(&sensor);
45 Serial.println(F("Umidade"));
46 Serial.print(F("Sensor: ")); Serial.println(sensor.name);
47 Serial.print(F("Valor máximo: ")); Serial.print(sensor.max_value); Serial.println(F("%"));
48 Serial.print(F("Valor mínimo: ")); Serial.print(sensor.min_value); Serial.println(F("%"));
49 Serial.print(F("Resolução: ")); Serial.print(sensor.resolution); Serial.println(F("%"));
50 Serial.println(F("-----"));
51
52 delayMS = sensor.min_delay / 1000;
53
54 Sala = 4;
55 Quarto = 2;
56 Cozinha = 14;
57 Banheiro = 13;
58
59 pinMode(Sala, OUTPUT);
60 pinMode(Quarto, OUTPUT);
61 pinMode(Cozinha, OUTPUT);
62 pinMode(Banheiro, OUTPUT);
63
64 Serial.begin(115200);
65
66 setup_wifi();
67 client.setServer(mqtt_server, 1883);
68 client.setCallback(callback);
69
70 }
71
72 void setup_wifi() {
73
74 delay(500);
75 Serial.println("");
76 Serial.print("Conectando com ");
77 Serial.println(ssid);
78 }
```

Fonte: captura de tela Arduino IDE

Figura 15



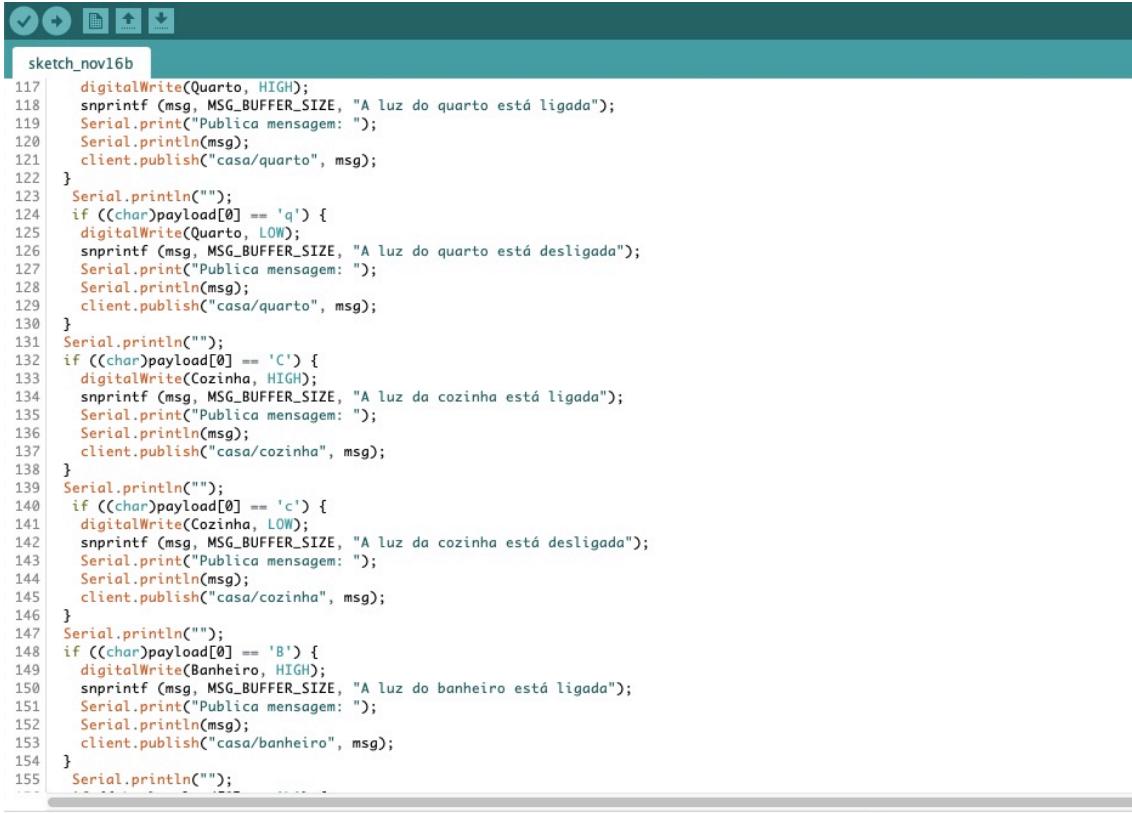
```
sketch_nov16b
79 WiFi.begin(ssid, password);
80 while (WiFi.status() != WL_CONNECTED) {
81   delay(500);
82   Serial.print(".");
83 }
84
85 Serial.println("");
86 Serial.println("WiFi conectado");
87 Serial.println("IP: ");
88 Serial.println(WiFi.localIP());
89
90 }
91
92 void callback(char* topic, byte* payload, unsigned int length) {
93   Serial.print("Mensagem recebida [");
94   Serial.print(topic);
95   Serial.print("]");
96   for (int i = 0; i < length; i++) {
97     Serial.print((char)payload[i]);
98   }
99   Serial.println("");
100  if ((char)payload[0] == 's') {
101    digitalWrite(Sala, HIGH);
102    sprintf(msg, MSG_BUFFER_SIZE, "A luz da sala está ligada");
103    Serial.print("Pública mensagem: ");
104    Serial.println(msg);
105    client.publish("casa/sala", msg);
106  }
107  Serial.println("");
108  if ((char)payload[0] == 's') {
109    digitalWrite(Sala, LOW);
110    sprintf(msg, MSG_BUFFER_SIZE, "A luz da sala está desligada");
111    Serial.print("Pública mensagem: ");
112    Serial.println(msg);
113    client.publish("casa/sala", msg);
114  }
115  Serial.println("");
116  if ((char)payload[0] == 'q') {
117    digitalWrite(Quarto, HIGH);
118  }
119 }
```

Fonte: Captura de tela Arduino IDE

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Figura 16



```
sketch_nov16b
117 digitalWrite(Quarto, HIGH);
118 sprintf (msg, MSG_BUFFER_SIZE, "A luz do quarto está ligada");
119 Serial.print("Publica mensagem: ");
120 Serial.println(msg);
121 client.publish("casa/quarto", msg);
122 }
123 Serial.println("");
124 if ((char)payload[0] == 'q') {
125 digitalWrite(Quarto, LOW);
126 sprintf (msg, MSG_BUFFER_SIZE, "A luz do quarto está desligada");
127 Serial.print("Publica mensagem: ");
128 Serial.println(msg);
129 client.publish("casa/quarto", msg);
130 }
131 Serial.println("");
132 if ((char)payload[0] == 'C') {
133 digitalWrite(Cozinha, HIGH);
134 sprintf (msg, MSG_BUFFER_SIZE, "A luz da cozinha está ligada");
135 Serial.print("Publica mensagem: ");
136 Serial.println(msg);
137 client.publish("casa/cozinha", msg);
138 }
139 Serial.println("");
140 if ((char)payload[0] == 'c') {
141 digitalWrite(Cozinha, LOW);
142 sprintf (msg, MSG_BUFFER_SIZE, "A luz da cozinha está desligada");
143 Serial.print("Publica mensagem: ");
144 Serial.println(msg);
145 client.publish("casa/cozinha", msg);
146 }
147 Serial.println("");
148 if ((char)payload[0] == 'B') {
149 digitalWrite(Banheiro, HIGH);
150 sprintf (msg, MSG_BUFFER_SIZE, "A luz do banheiro está ligada");
151 Serial.print("Publica mensagem: ");
152 Serial.println(msg);
153 client.publish("casa/banheiro", msg);
154 }
155 Serial.println("");
```

Fonte: captura de tela Arduino IDE

Figura 17



```
sketch_nov16b
155 Serial.println("");
156 if ((char)payload[0] == 'b') {
157 digitalWrite(Banheiro, LOW);
158 sprintf (msg, MSG_BUFFER_SIZE, "A luz do banheiro está desligada");
159 Serial.print("Publica mensagem: ");
160 Serial.println(msg);
161 client.publish("casa/banheiro", msg);
162 }
163 }
164 void reconnect() {
165 while (!client.connected()) {
166 Serial.print("Aguardando conexão MQTT...");
167 String clientId = "ESP8266Client";
168 clientId += String(random(0xffff), HEX);
169 if (client.connect(clientId.c_str())) {
170 Serial.println("conectado");
171 client.subscribe("casa/publisher");
172 } else {
173 Serial.print("Falhou, rc=");
174 Serial.print(client.state());
175 Serial.println("tente novamente em 5s");
176 delay(5000);
177 }
178 }
179 }
180 }
181 void loop() {
182 delay(delayMS);
183 sensors_event_t event;
184 dht.temperatureEvent(&event);
185 if (isnan(event.temperature)) {
186 Serial.println(F("Erro na leitura da temperatura"));
187 }
188 else {
189 Serial.print(F("Temperatura: "));
190 Serial.print(event.temperature);
191 Serial.println(F("°C"));
192 }
```

// laço de reconexão
// enquanto cliente conectando
// escrever no monitor serial aguardando conexão MQTT
// cria uma identidade tipo cliente aleatória

// se realizado a conexão
// escrever no monitor serial "conectado"
// cliente subscreve (casa/publisher)
// Caso o contrário
// imprime mensagem de falha
// imprime status do cliente

// atraso de 5 segundos

// laço de repetição
// atraso
// iniciar a medição do sensor
// inicia o sensor DHT
// caso não ocorra a leitura da temperatura
// imprimir a frase: erro na leitura da temperatura

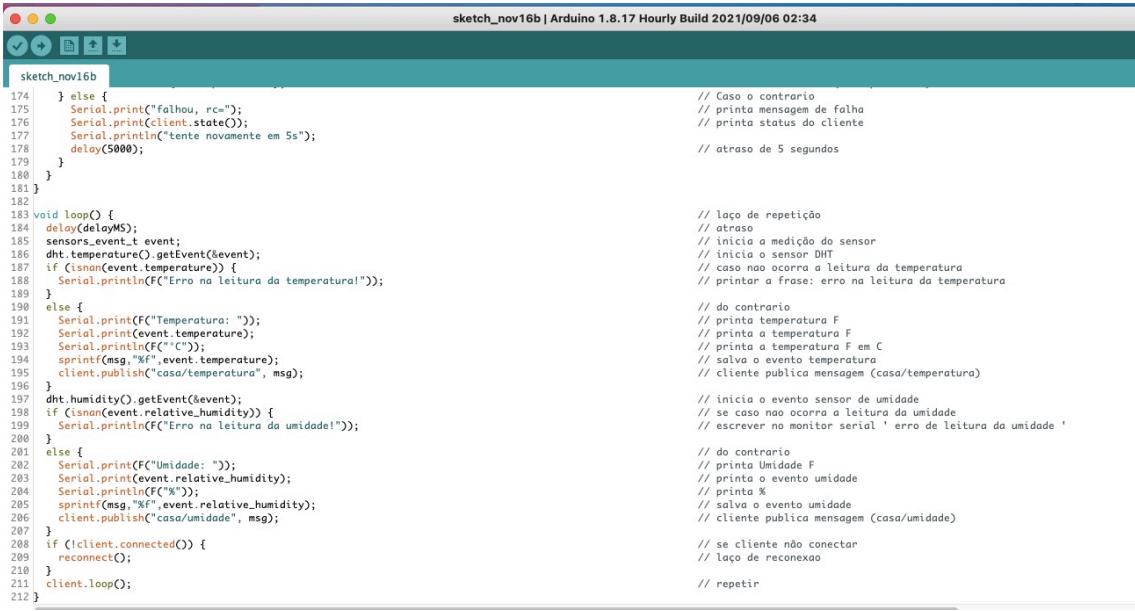
// do contrário
// printa temperatura F
// printa a temperatura F
// printa a temperatura F em C

Fonte: captura de tela Arduino IDE

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Figura 18



```
sketch_nov16b | Arduino 1.8.17 Hourly Build 2021/09/06 02:34

sketch_nov16b

174     } else {
175         Serial.print("failhou, recendo");
176         Serial.print(client.state());
177         Serial.println("tente novamente em 5s");
178         delay(5000);
179     }
180 }
181 }

182 void loop() {
183     delay(delayMS);
184     sensors_event_t event;
185     dht.temperature().getEvent(&event);
186     if (!isnan(event.temperature)) {
187         Serial.println(F("Erro na leitura da temperatura"));
188     } else {
189         Serial.printf("Temperatura: %f");
190         Serial.print(event.temperature);
191         Serial.println(F("%C"));
192         sprintf(msg, "%f", event.temperature);
193         client.publish("casa/temperatura", msg);
194     }
195     dht.humidity().getEvent(&event);
196     if (!isnan(event.relative_humidity)) {
197         Serial.println(F("Erro na leitura da umidade!"));
198     } else {
199         Serial.printf("Umidade: %f");
200         Serial.print(event.relative_humidity);
201         Serial.println(F("%"));
202         sprintf(msg, "%f", event.relative_humidity);
203         client.publish("casa/umidade", msg);
204     }
205     if (!client.connected()) {
206         reconnect();
207     }
208     client.loop();
209 }

// Caso o contrario
// printa mensagem de falha
// printa status do cliente
// atraso de 5 segundos

// laço de repetição
// atraso
// inicia a medição do sensor
// inicia o sensor DHT
// caso não ocorra a leitura da temperatura
// printar a frase: erro na leitura da temperatura

// do contrario
// printa temperatura F
// printa a temperatura F
// printa a temperatura F em C
// salva o evento temperatura
// cliente publica mensagem (casa/temperatura)

// inicia o evento sensor de umidade
// se caso não ocorra a leitura da umidade
// escrever no monitor serial ' erro de leitura da umidade '

// do contrario
// printa Umidade F
// printa o evento umidade
// printa %
// salva o evento umidade
// cliente publica mensagem (casa/umidade)

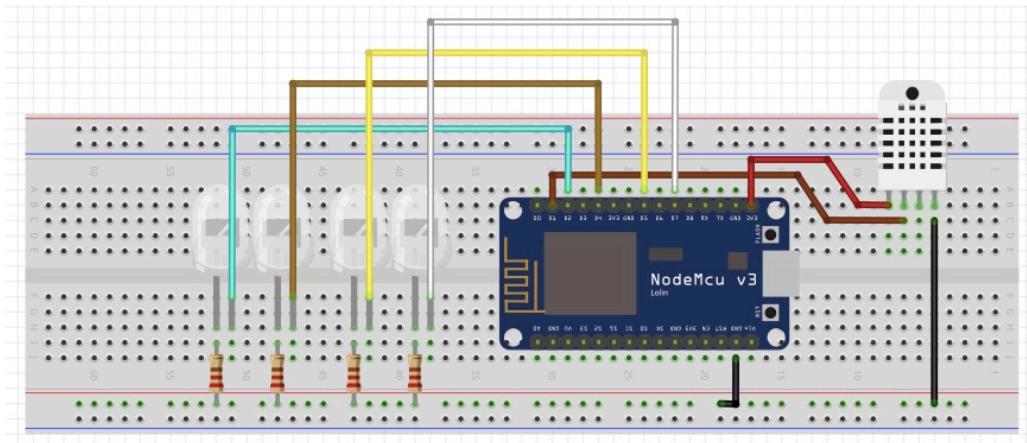
// se cliente não conectar
// laço de reconexão

// repetir
```

Fonte: captura de tela Arduino IDE

Na figura 19, o modelo de ligação dos componentes feito pela ferramenta *Fritzing*:

Figura 19



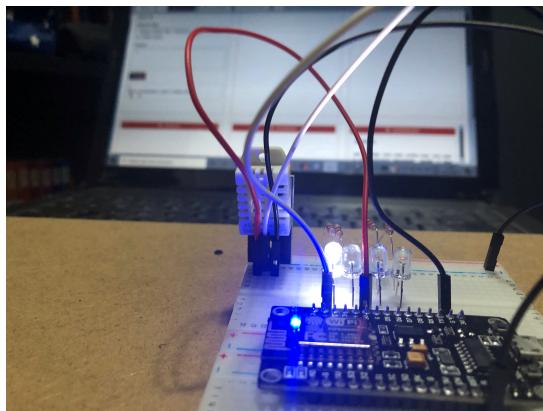
Fonte: site Curto Circuito (adaptado)

2.2.5 Imagens do projeto

UNIVERSIDADE PRESBITERIANA MACKENZIE

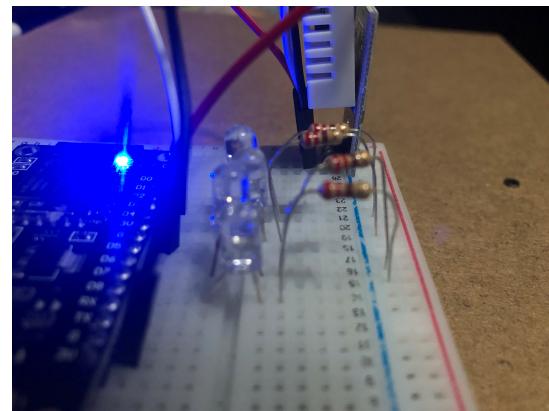
Faculdade de Computação e Informática

Figura 20: fotografia dos componentes montados



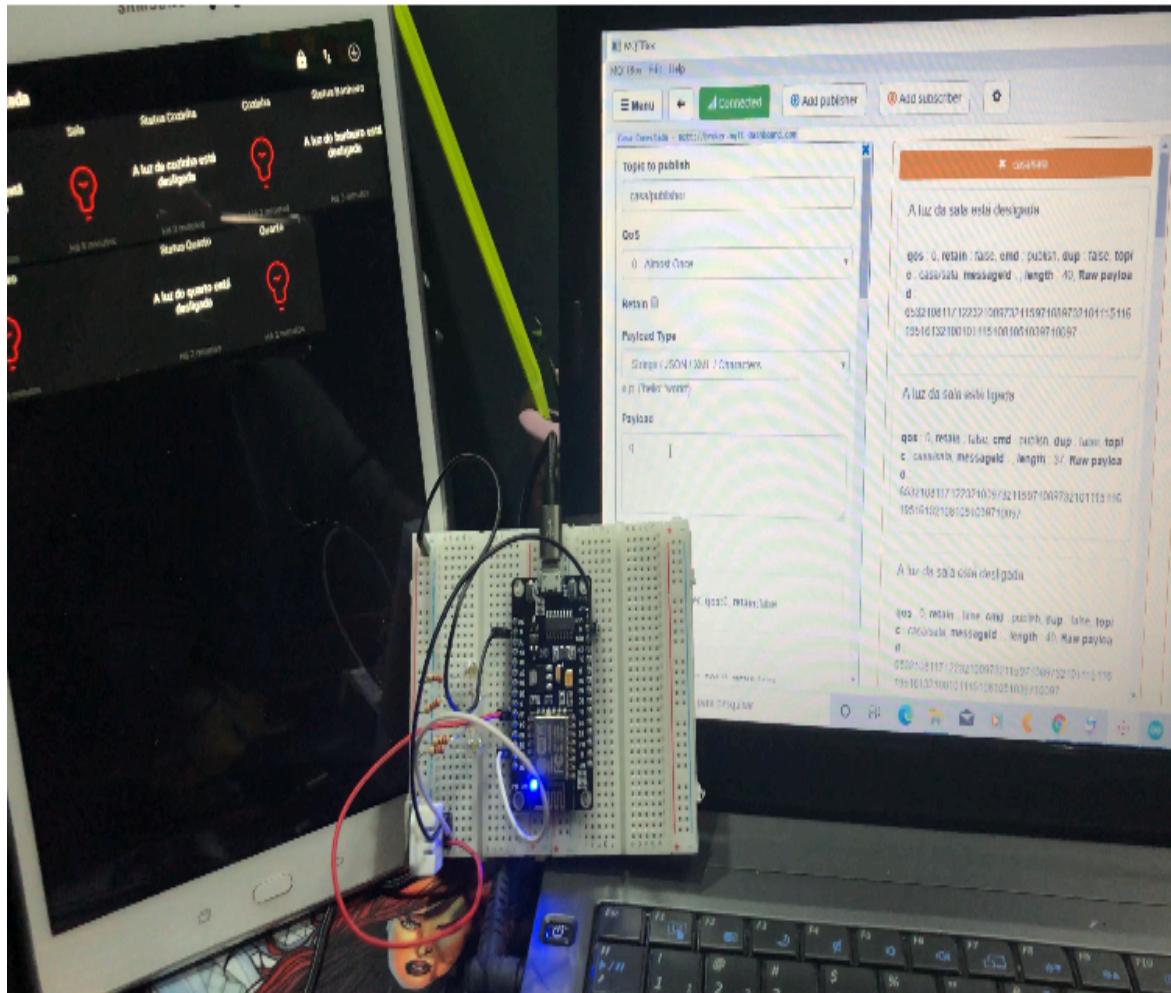
Fonte: elaborado pelo autor

Figura 21: foto mais detalhada dos componentes



Fonte: elaborado pelo autor

Figura 22: foto ilustrando o projeto pronto e em funcionamento.

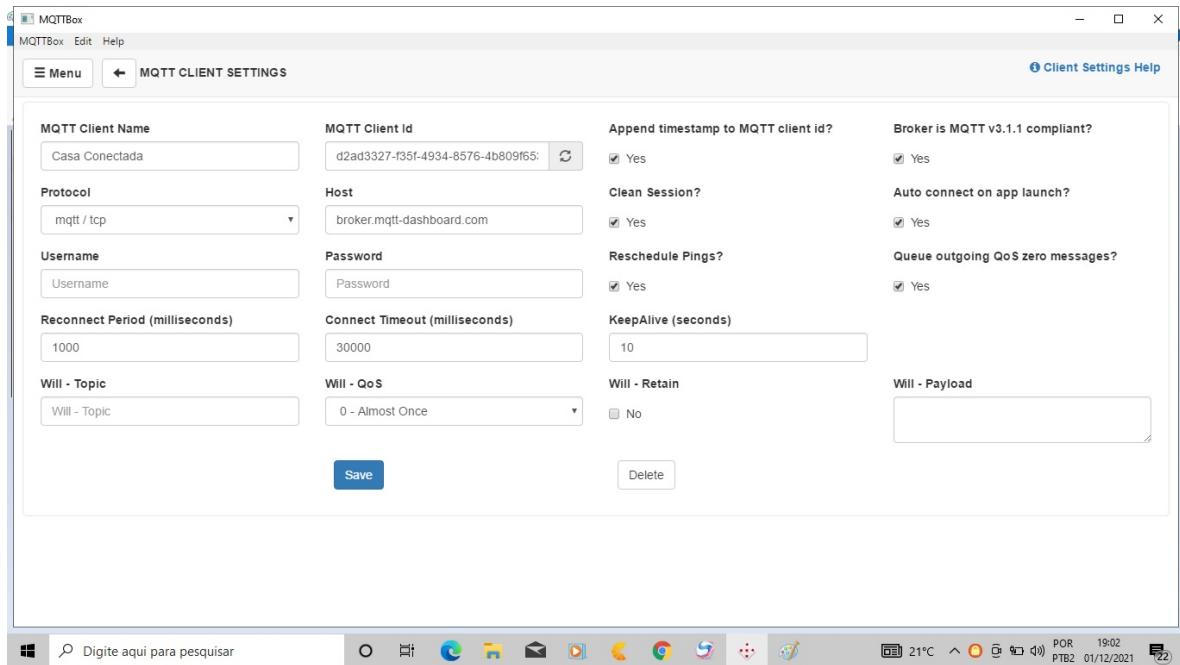


Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

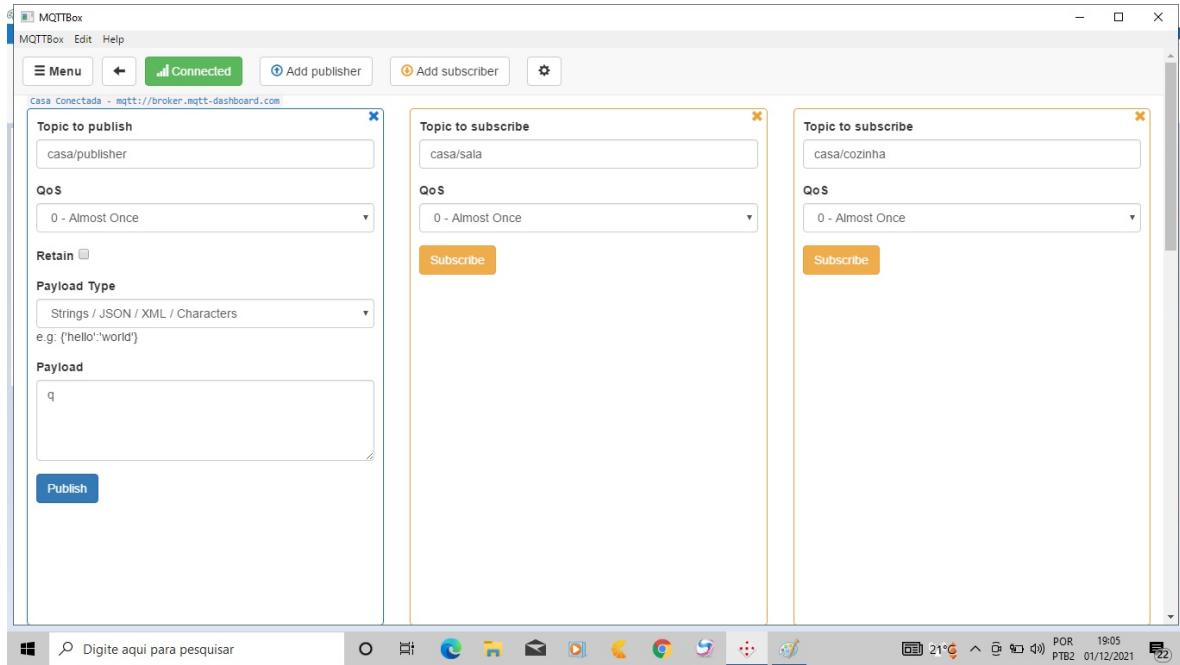
Faculdade de Computação e Informática

Figura 23: captura de tela da configuração do MQTTBox



Fonte: captura de tela MQTTBox

Figura 24: captura de tela demonstrando que a conexão foi estabelecida com sucesso

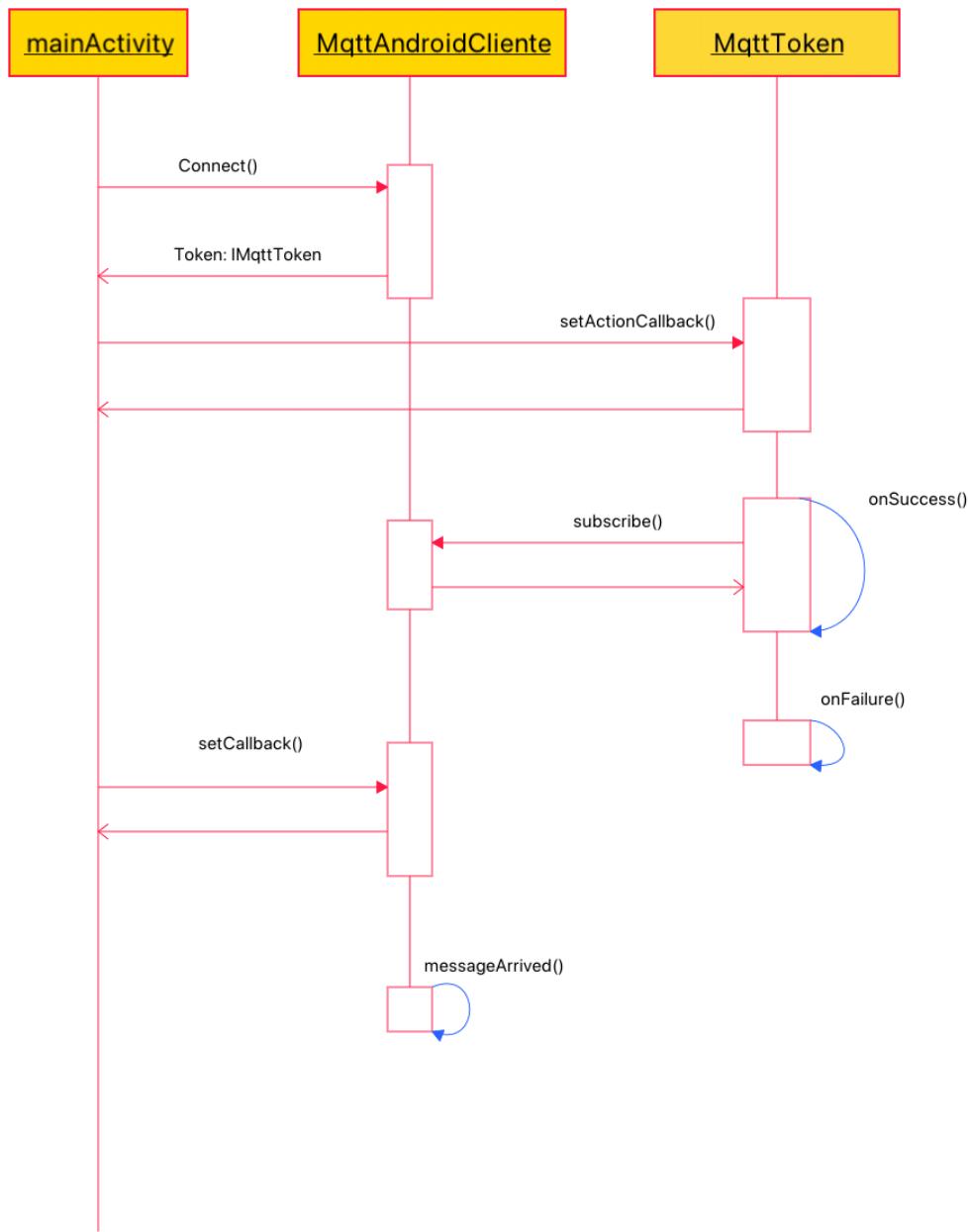


Fonte: captura de tela MQTTBox

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Figura 25: diagrama de sequencia do funcionamento do MQTT



Fonte: site USP

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

3. Resultados

Nesta sessão será apresentado os resultados dos testes realizados com o protótipo Casa Conectada. Foram realizados 4 testes para cada sensor/atuador. Os gráficos a seguir ilustram o comportamento do aplicativo ligando e desligando repetidas vezes.

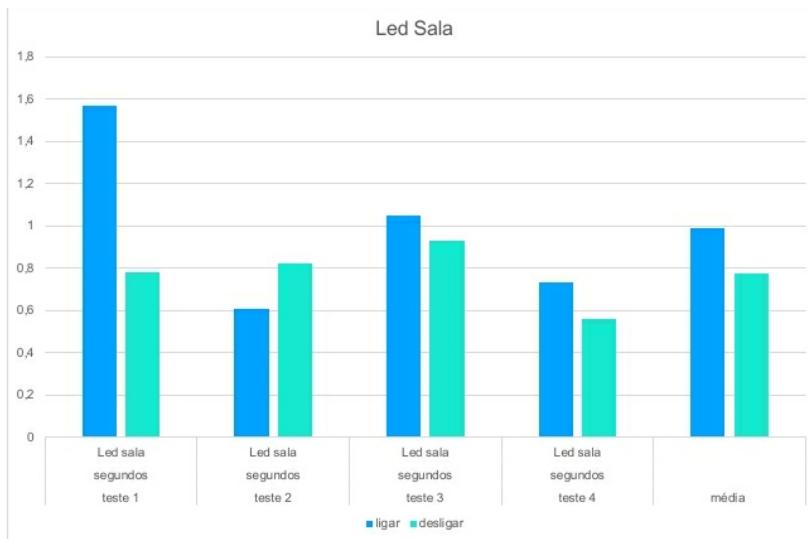
3.1 Testagem de tempo de resposta: comunicação MQTT DASH x LED.

Tabela 6

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led sala	1,57	0,78
teste 2	segundos	Led sala	0,61	0,82
teste 3	segundos	Led sala	1,05	0,93
teste 4	segundos	Led sala	0,73	0,56
		média	0,99	0,77

Fonte: elaborado pelo autor

Gráfico 1



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

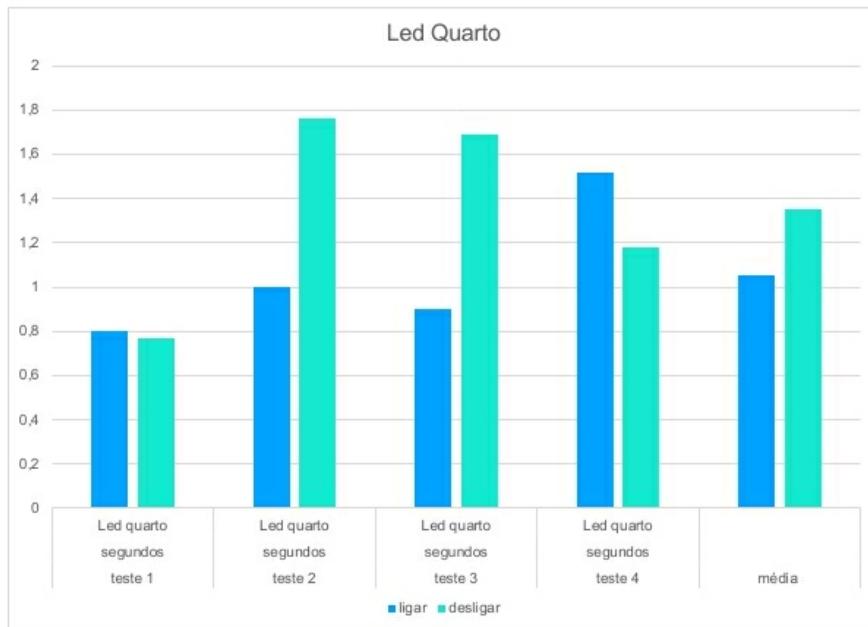
Faculdade de Computação e Informática

Tabela 7

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led quarto	0,80	0,77
teste 2	segundos	Led quarto	1,00	1,76
teste 3	segundos	Led quarto	0,90	1,69
teste 4	segundos	Led quarto	1,52	1,18
		média	1,06	1,35

Fonte: elaborado pelo autor

Gráfico 2



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

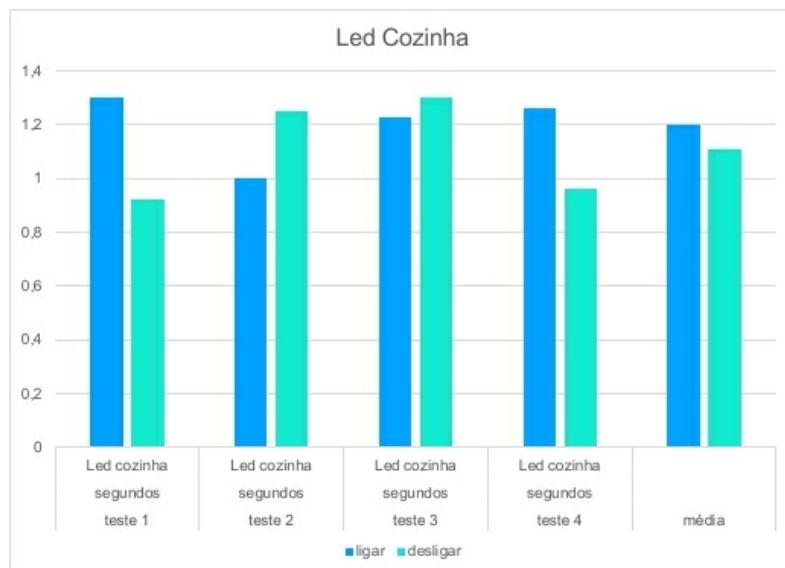
Faculdade de Computação e Informática

Tabela 8

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led cozinha	1,30	0,92
teste 2	segundos	Led cozinha	1,00	1,25
teste 3	segundos	Led cozinha	1,23	1,30
teste 4	segundos	Led cozinha	1,26	0,96
		média	1,20	1,11

Fonte: elaborado pelo autor

Gráfico 3



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

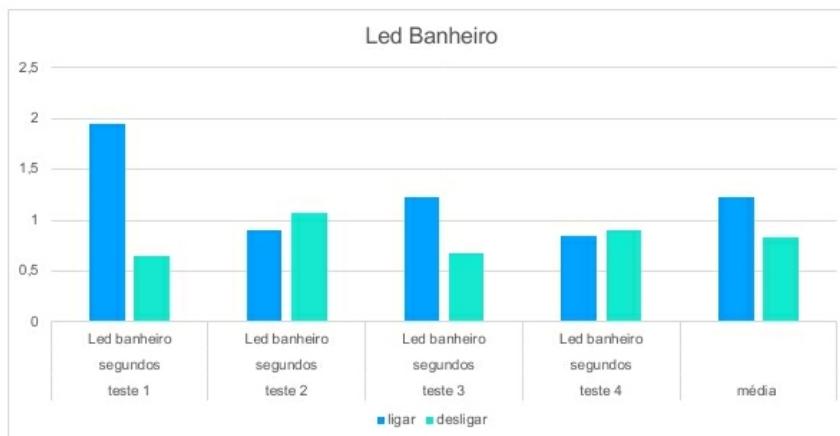
Faculdade de Computação e Informática

Tabela 9

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led banheiro	1,95	0,65
teste 2	segundos	Led banheiro	0,90	1,07
teste 3	segundos	Led banheiro	1,23	0,68
teste 4	segundos	Led banheiro	0,85	0,90
		média	1,23	0,83

Fonte: elaborado pelo autor

Gráfico 4



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

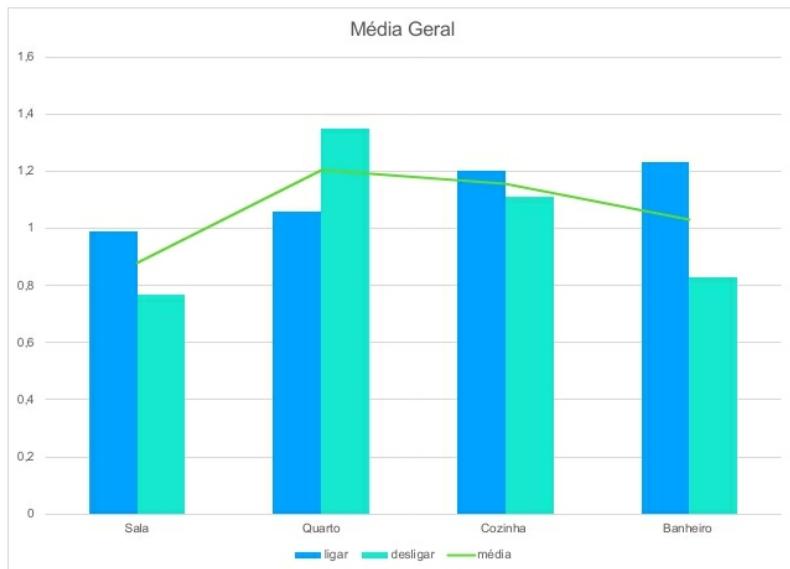
Faculdade de Computação e Informática

Tabela 10

	ligar	desligar	média
Sala	0,99	0,77	0,88
Quarto	1,06	1,35	1,21
Cozinha	1,20	1,11	1,16
Banheiro	1,23	0,83	1,03

Fonte: elaborado pelo autor

Gráfico 5



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

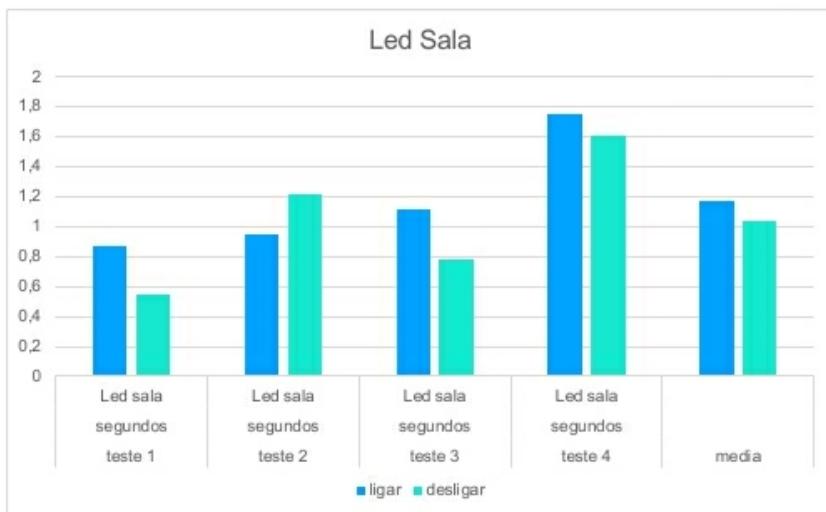
Faculdade de Computação e Informática

Tabela 11

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led sala	0,87	0,55
teste 2	segundos	Led sala	0,95	1,21
teste 3	segundos	Led sala	1,12	0,78
teste 4	segundos	Led sala	1,75	1,60
		média	1,17	1,04

Fonte: elaborado pelo autor

Gráfico 6



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

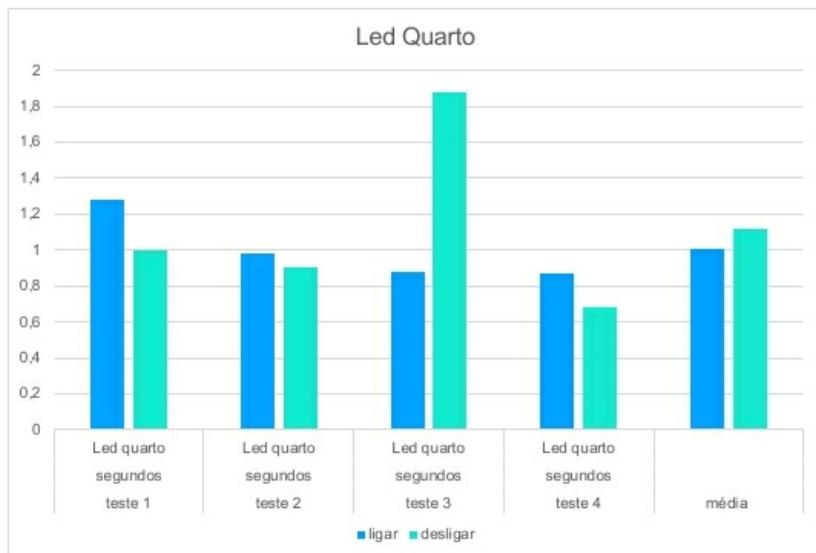
Faculdade de Computação e Informática

Tabela 12

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led quarto	1,28	1,00
teste 2	segundos	Led quarto	0,98	0,90
teste 3	segundos	Led quarto	0,88	1,88
teste 4	segundos	Led quarto	0,87	0,68
		média	1,00	1,12

Fonte: elaborado pelo autor

Gráfico 7



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

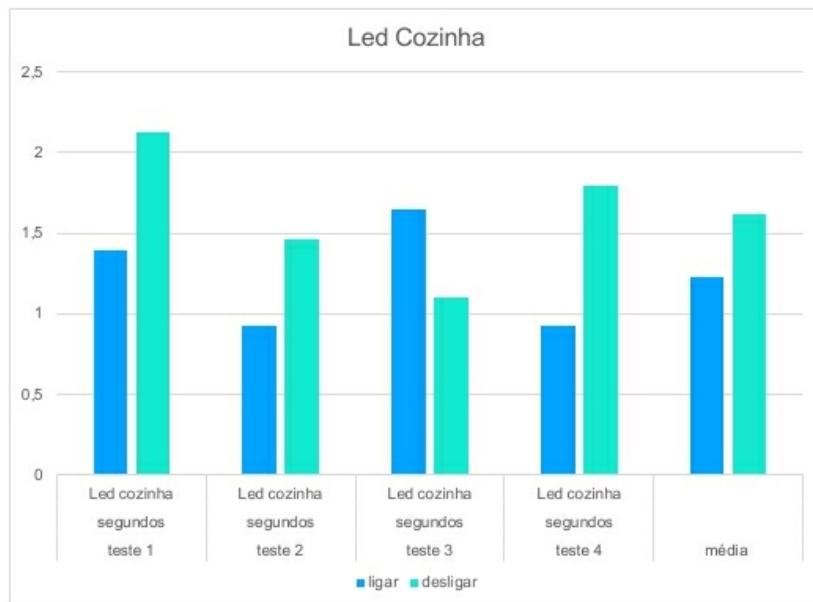
3.2 Testagem de tempo de resposta: comunicação MQTT BOX x LED

Tabela 13

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led cozinha	1,39	2,13
teste 2	segundos	Led cozinha	0,93	1,46
teste 3	segundos	Led cozinha	1,65	1,10
teste 4	segundos	Led cozinha	0,93	1,79
		média	1,23	1,62

Fonte: elaborado pelo autor

Gráfico 8



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

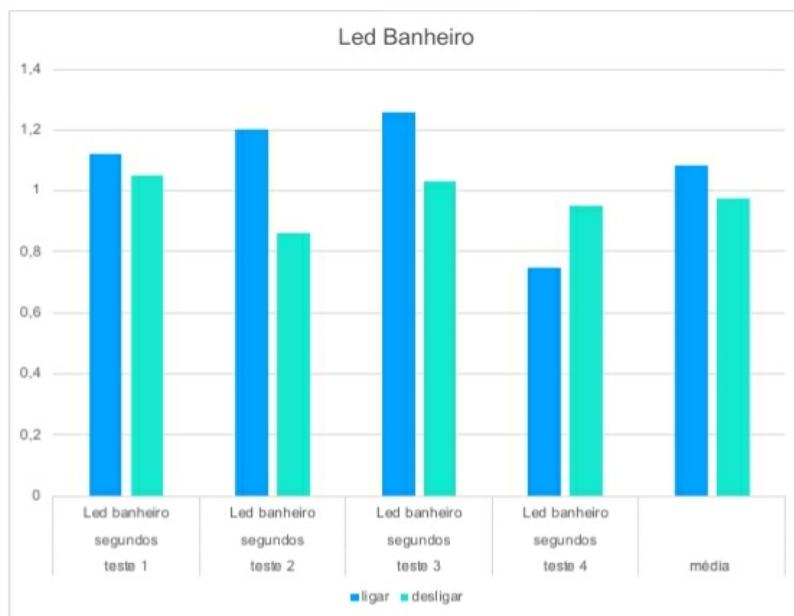
Faculdade de Computação e Informática

Tabela 14

	num/medida	sensor/atuador	ligar	desligar
teste 1	segundos	Led banheiro	1,12	1,05
teste 2	segundos	Led banheiro	1,20	0,86
teste 3	segundos	Led banheiro	1,26	1,03
teste 4	segundos	Led banheiro	0,75	0,95
		média	1,08	0,97

Fonte: elaborado pelo autor

Gráfico 9



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

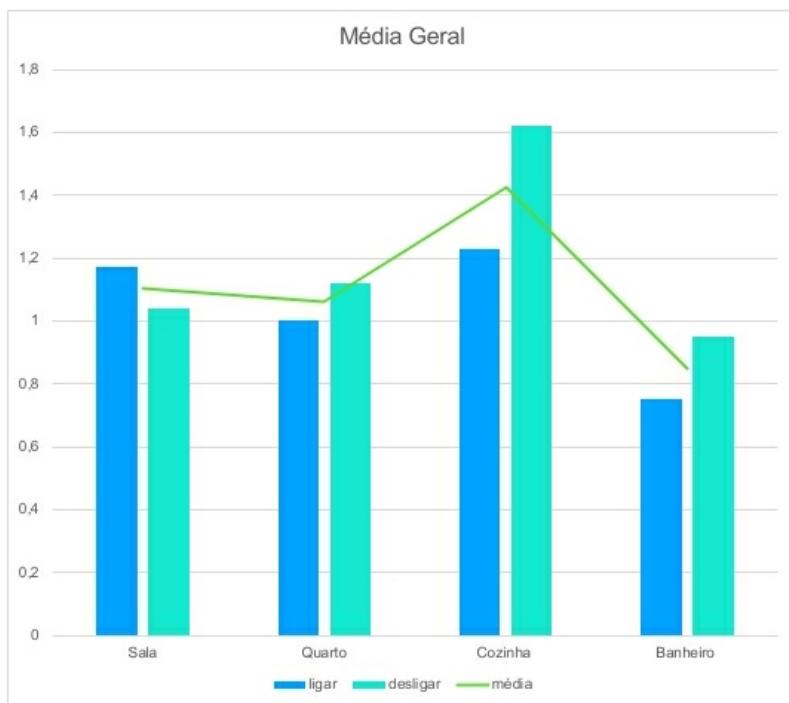
Faculdade de Computação e Informática

Tabela 15

	ligar	desligar	média
Sala	1,17	1,04	1,11
Quarto	1,00	1,12	1,06
Cozinha	1,23	1,62	1,43
Banheiro	0,75	0,95	0,85

Fonte: elaborado pelo autor

Gráfico 10



Fonte: elaborado pelo autor

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Como observado nas tabelas anteriores o tempo médio de resposta do dispositivo foi menor que 2 segundos, já o sensor de umidade e temperatura tem como tempo de resposta 2 segundos, essa informação pode ser vista na tabela 5 deste artigo.

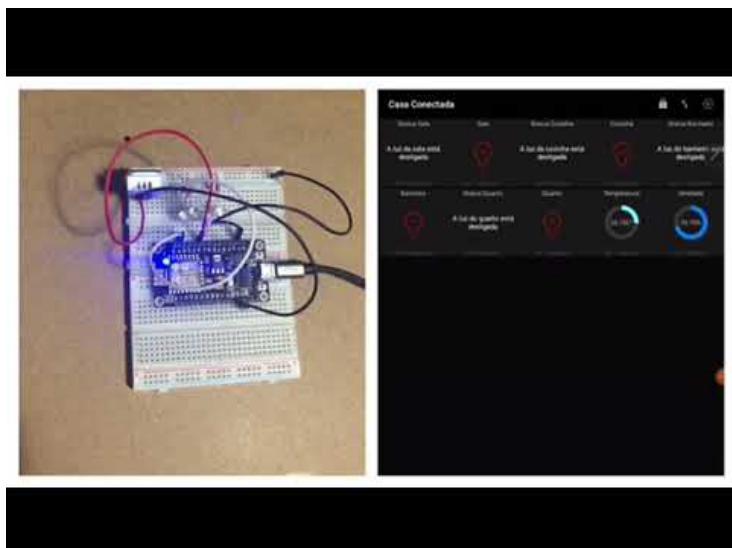
O dispositivo IoT pode ser controlado por meio do MQTTDash, uma aplicação para quem possui celulares ou tablets Android. Abaixo a imagem do MQTTDash já em funcionamento.

Figura 26: aplicação MQTTDash ilustrando o status das luzes.



Fonte: Captura de tela MQTTDash

O video com a demonstração do funcionamento do protótipo encontra-se upado no YouTube, o link para acesso é o:



<https://www.youtube.com/watch?v=X7s6NiOTvOQ>

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

As documentações, gráficos e código-fonte usados para o projeto encontra-se neste repositório do github <<https://github.com/yumitakeyama/casaconectada>>.

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

4.0 Conclusão

1. Os objetivos propostos foram alcançados?

Objetivo do projeto era a criação de um protótipo cuja a comunicação deveria se realizada pelo protocolo MQTT, além de ter a presença de um atuador e um sensor. A comunicação do projeto foi atendida como o esperado bem como presença do sensor, no entanto a presença do atuador ficou pouco evidente. Sendo assim os objetivos do projeto foram parcialmente atendidos

2. Quais são os principais problemas enfrentados e como foram resolvidos?

O principal problema encontrado foi na comunicação com o protocolo MQTT, pois em um projeto inicial, diferente deste apresentado, a plataforma utilizada era diferente da requerida tornando complicada uma adaptação e sendo assim mudei o projeto. Neste projeto atual (e no anterior) um dos pontos que achei complicado foi a conexão das peças por serem muito delicadas eu danifiquei algumas, além de ter queimado a placa, mas esses problemas foram resolvidos com a compra de novos componentes.

3. Quais são as vantagens e desvantagens do projeto?

Neste projeto é possível apontar vantagens como a administração do uso da energia, interface acessível para usuários, portabilidade e comodidade.

Já desvantagem é que para implementar necessitaria de lâmpadas inteligentes e este acessório tem um preço elevado.

4. O que deveria/poderia ser feito para melhorar o projeto?

Como melhoria poderia buscar formas de diminuir o custo do projeto, pois apesar de gerar economia de energia, o custo de implantação é alto.

UNIVERSIDADE PRESBITERIANA MACKENZIE

Faculdade de Computação e Informática

Referências Bibliográficas

MONTEIRO, Mario Antonio. **Introdução à Organização de Computadores**, Rio de Janeiro: LTC, 2015.

SCHWAB, Klaus. **A Quarta Revolução Industrial**, São Paulo: Edipro, 2016.

STEVAN, Sergio Luiz Jr. **IoT Internet das Coisas: Fundamentos e aplicações em Arduino e NodeMCU**, São Paulo: Erica, 2018.

CURTOCIRCUITO, Monitoramento e controle por aplicativo MQTT, Sao Paulo.

Disponível em <<https://www.curtocircuito.com.br/blog/monitoramento-e-controle-por-aplicativo-mqtt>> Acesso em 10 de novembro 2021.

LUECH , Knud Lasse. Why the Internet of Things is called Internet of Things: definition, history, disambiguation, **IoT Analytics**, Hamburg: [s.n.], 2014. Disponível em <<https://iot-analytics.com/internet-of-things-definition/>> Acesso em 07 setembro 2021.

UFRJ, O que é RFID. **Grupo de Teleinformática e Automação**, Rio de Janeiro: [s.n.]. Disponível em <https://www.gta.ufrj.br/grad/07_1/rfid/RFID_arquivos/o%20que%20e.htm> Acesso em 07 setembro 2020.

GTA UFRJ, MQTT. Rio de Janeiro:[s.n.]. Disponível em <<https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/mqtt/>> Acesso em 10 novembro 2021

THE ENGINEERING PROJECTS. **Datasheet - NodeMCU**, Paquistão:[s.n]. Disponível em <<https://www.theengineeringprojects.com/Downloads/Datasheet/NodeMCU-V3.pdf>> Acesso em 02 dezembro 2021.

ELECTROS SCHEMATICS. **Datasheet - DHT22**, China:[s.n]. Disponível em <<https://www.electroschematics.com/wp-content/uploads/2015/02/DHT22-datasheet.pdf>> Acesso em 02 dezembro 2021.

USP, Arquitetura de IoT para cidades inteligentes, São Paulo: [s.n], 2016. Disponível em <https://edisciplinas.usp.br/pluginfile.php/4207480/mod_resource/content/1/SlideTCC1.pdf> Acesso em 02 dezembro 2021