IOT E AUTOMATIZAÇÃO DE IRRIGAÇÃO NA AGRICULTURA

Vladimir da Silva Junior, Andre Luis de Oliveira, Leandro Carlos Fernandes

Faculdade de Computação e Informática

Universidade Presbiteriana Mackenzie (UPM) - São Paulo, SP- Brazil

ABSTRACT. This article presents the possibility of using the Internet of Things (IOT) in agricultural production to automate soil moisture control, using sensors and a means of communication with the Internet by the MQTT protocol to turn irrigation on and off.

Keywords: Internet of Things; Automate; Irrigation.

RESUMO. Esse artigo apresenta a possibilidade de utilizar a internet das coisas (IOT) em produções agrícolas para automatizar o controle da umidade da terra, utilizando sensores e um meio de comunicação com internet via protocolo MQTT para ligar e desligar a irrigação.

Palavras-chave: Internet das coisas; Automatizar; Irrigação.

1. INTRODUÇÃO

A internet das coisas (IOT) refere-se a objetos, agregados de sensores e softwares, conectados através da internet com o objetivo de auxiliar o cotidiano doméstico e empresarial, tem-se previsto que em 2025 teriam 22 bilhões de dispositivos conectados (ORACLE, *s.d.*).

Entre as diversas aplicações do IOT tem-se como grande aliado sua utilização em um dos objetivos do ODS (Objetivos de Desenvolvimento Sustentável), como o sexto objetivo que trata a respeito da água potável e saneamento, principalmente o indicador 6.4.1 alteração da eficiência no uso da água ao longo do tempo (INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA, *s.d.*). Sendo exemplo, a utilização de dispositivos que identificam a umidade da terra para ligar e desligar a irrigação de forma automática, evitando desperdícios (INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA, *s.d.*).

2. MATERIAIS E MÉTODOS

Neste projeto serão necessários dois conjuntos de componentes, o primeiro conjunto será focado na umidade do solo, possuindo dois protoboard, no qual, será conectado um módulo de sensor ligado ao sensor de umidade e módulo Arduino Nodemcu esp8266 que enviará um sinal indicando o nível de umidade pelo protocolo MQTT. Sendo esses componentes ilustrados abaixo.

Figura 1. Protoboard. Fonte: https://www.usinainfo.com.br/protoboard/protoboard-400-pontos-para-montagem-de-projetos-2323.html.

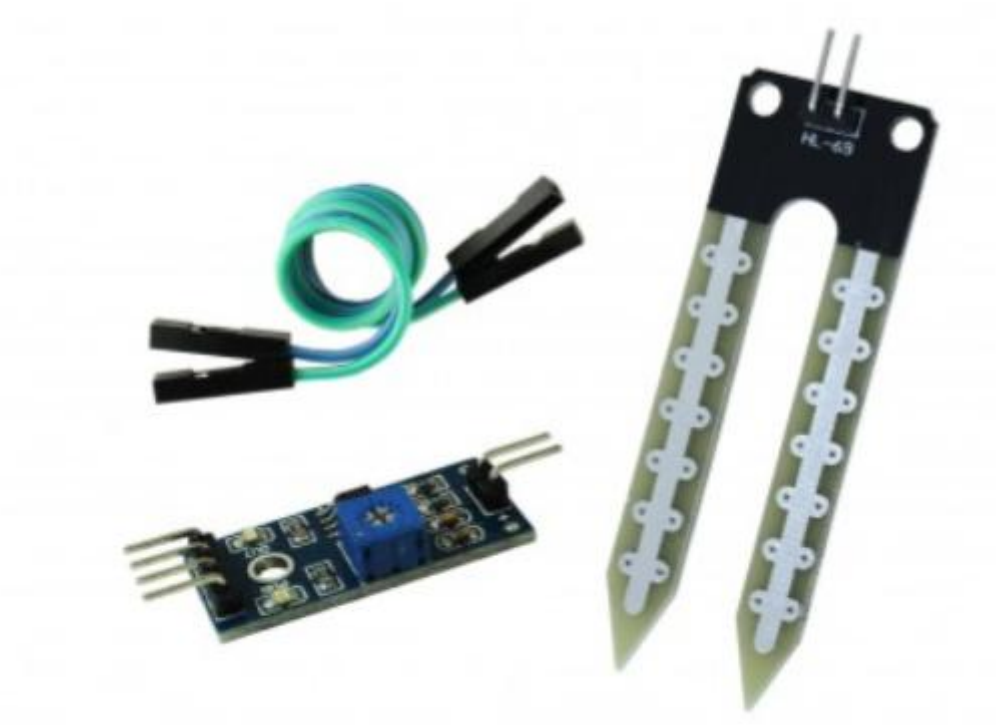

Figura 2. Sensor de umidade. Fonte: https://www.usinainfo.com.br/sensor-de-solo/sensor-de-umidade-de-solo-hl-69-para-arduino-2311.html.

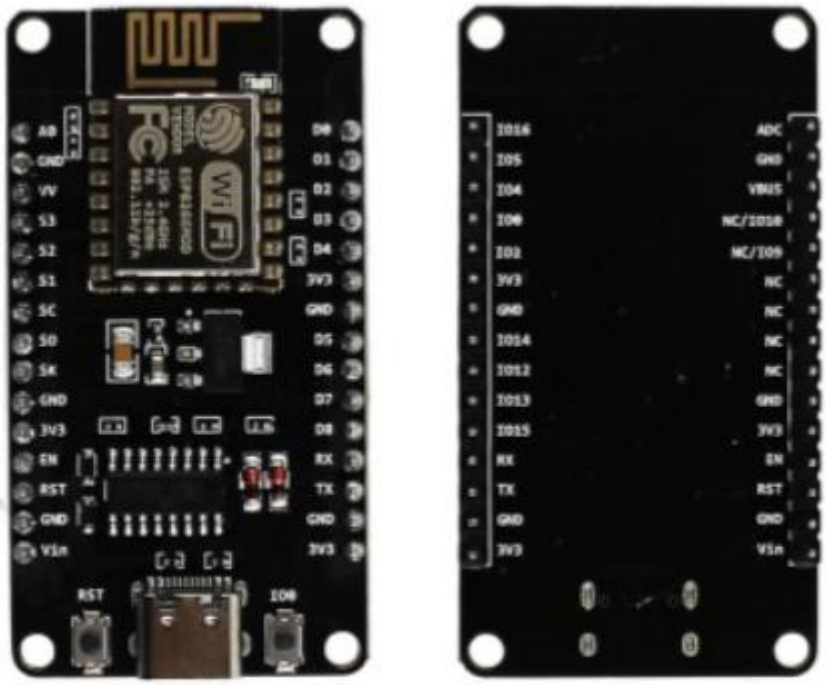

Figura 3. Nodemcu esp8266. Fonte: https://www.usinainfo.com.br/esp8266/nodemcu-v3-esp8266-esp-12e-iot-wifi-com-usb-c-8791.html.

No segundo conjunto de componentes tem-se outros dois protoboard que será conectado a um outro módulo Arduino Nodemcu esp8266 que receberá o sinal MQTT e um atuador relé que vai ligar/desligar a bomba da irrigação. Sendo estes componentes a seguir.

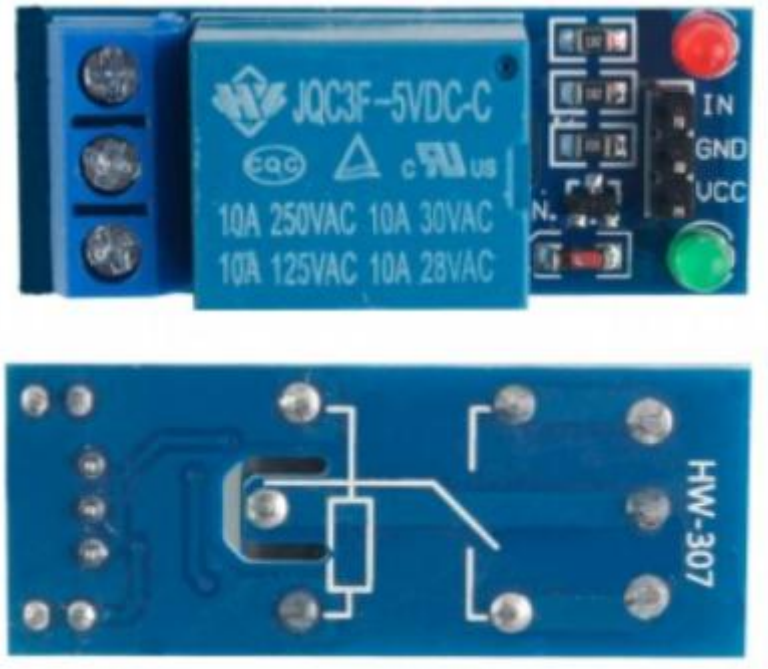

Figura 4 - Atuador relé. Fonte: https://www.usinainfo.com.br/rele-arduino/modulo-rele-5v-10a-1-canal-hw-307-8054.html.

3. RESULTADOS

Para fazer a programação da lógica que será aplicada nos componentes será utilizado o Arduino IDE, com uma biblioteca “PubSubClient” que é preparada para a conexão e para a conexão dos componentes MQTT será utilizado o serviço de broker Eclipse IOT.

```
#include <PubSubClient.h>
#include <ESP8266WiFi.h>
```

Figura 5 - Adição da biblioteca PubSubClient e ESP8266. Fonte: autor.

No primeiro conjunto de dispositivos o sensor de umidade do solo mede o nível de umidade e manda para o Nodemcu esp8266, no qual de acordo com a lógica de

programação define qual o nível de umidade necessário para mandar um sinal por meio do MQTT para ligar ou não a bomba de irrigação.

Isso através da conexão via Wi-Fi informada no código para conexão ao MQTT.

```
//WiFi
const char* SSID = "*********";
const char* PASSWORD = "***********";
WiFiClient wifiClient;
```

Figura 6 - Conexão ao Wi-Fi. Fonte: autor.

E para conexão do MQTT é necessário informar com qual o Broker a ser utilizar e a porta para conexão. Nesse caso, será utilizado o Broker “iot.eclipse.org” e a porta “1883”.

```
//MQTT Server
const char* BROKER_MQTT = "iot.eclipse.org";
int BROKER_PORT = 1883;
```

Figura 7 - Conexão ao Broker e porta. Fonte: autor.

Depois de declarar as informações necessárias para conexão, pode declarar as funções a serem utilizadas na lógico para funcionamento do projeto.

```
//Declaração das funções
void mantemConexao();
void conectaWifi();
void conectaMQTT();
void enviaValores();
```

Figura 8 - Declaração de funções. Fonte: autor.

Na função “setup()”, necessário para o funcionamento do esp8266, é informado o pino de leitura do sensor de umidade do solo, a conexão do Wi-Fi e a conexão ao Broker MQTT.

```
void setup() {
  pinMode(pinSolo, INPUT);

  Serial.begin(115200);

  conectaWifi();
  MQTT.setServer(BROKER_MQTT, BROKER_PORT);
}
```

Figura 9 - Função setup. Fonte: autor.

Depois, na função “loop()”, outra função necessária para o funcionamento, mantém a conexão, envia o valor para o Broker MQTT e faz o loop no MQTT.

```
void loop() {
  mantemConexao();
  enviaValores();
  MQTT.loop();
}
```

Figura 10 - Função loop. Fonte: autor.

Em seguida, é feito a lógica nas funções declaradas anteriormente para o funcionamento do projeto dos componentes de leitura da umidade do solo, sendo essas funções a seguir.

```
void mantemConexao(){
  if(!MQTT.connected()){
    conectaMQTT();
  }

  conectaWifi();
}
```

Figura 11 - Função mantemConexao. Fonte: autor.

```
void conectaWifi(){
  if(WiFi.status() == WL_CONNECTED){
    return;
  }

  Serial.print("Conectando-se na rede: ");
  Serial.print(SSID);
  Serial.println(" Aguarde!");

  WiFi.begin(SSID, PASSWORD);
  while(WiFi.status() != WL_CONNECTED){
    delay(100);
    Serial.print(".");
  }

  Serial.println();
  Serial.print("Conectado com sucesso, na rede: ");
  Serial.print(SSID);
  Serial.print(" IP obtido: ");
  Serial.println(WiFi.localIP());
}
```

Figura 12 - Função conectaWifi. Fonte: autor.

```
void conectaMQTT(){
  while(!MQTT.connected()){
    Serial.print("Conectando ao Broker MQTT: ");
    Serial.println(BROKER_MQTT);
    if(MQTT.connect(ID_MQTT)){
      Serial.println("Conectado ao Broker com sucesso");
    }
    else{
      Serial.println("Não foi possível se conectar ao Broker!");
      Serial.println("Nova tentativa de conexão em 10s");
      delay(10000);
    }
  }
}
```

Figura 13 - Função conectaMQTT. Fonte: autor.

```
void enviaValores(){
  static int estadoSolo;
  static unsigned long debounceSolo;

  estadoSolo = analogRead(pinSolo);
  if ( (millis() - debounceSolo) > 30 ){
    if(estadoSolo < 500){
      //Ligar bomba de água
      MQTT.publish(TOPIC_PUBLISH, "1");
      Serial.println("Bomba ligada. Payload enviado.");

      debounceSolo = millis();
    }else if(estadoSolo > 500){
      //Desligar bomba de água
      MQTT.publish(TOPIC_PUBLISH, "0");
      Serial.println("Bomba desligada. Payload enviado.");

      debounceSolo = millis();
    }
  }
}
```

Figura 14 - Função enviaValores. Fonte: autor.

O segundo conjunto de dispositivo possui um módulo Arduino Nodemcu esp8266 que recebe um sinal através do MQTT e dependendo do sinal aciona um relé para ligar ou desligar a bomba de irrigação.

Nesse conjunto, a inclusão da biblioteca PubSubClient, esp8266, conexão ao Wi-Fi, conexão ao Broker MQTT (alterando apenas o ID do conjunto esp8266), as funções declaradas (alterando para receber os pacotes), a função “setup()” (alterando o pino e adicionando a função “setCallback” no MQTT), a função “loop()” (excluindo a função enviaValores), função mantemConexao, conectaWiFi e conectaMQTT continuam iguais ou com algumas alterações, sendo essas a seguir.

```
#define ID_MQTT "BCI02"
#define TOPIC_PUBLISH "BCISolo"
PubSubClient MQTT(wifiClient);
```

Figura 15 - Alteração no ID. Fonte: autor.

```
//Declaração das funções
void mantemConexao();
void conectaWiFi();
void conectaMQTT();
void recebePacote(char* topic, byte* payload, unsigned int length);
```

Figura 16 - Alteração para receber pacotes. Fonte: autor.

```
void setup() {
  pinMode(pinRele, OUTPUT);

  Serial.begin(115200);

  conectaWiFi();
  MQTT.setServer(BROKER_MQTT, BROKER_PORT);
  MQTT.setCallback(recebePacote);
}
```

Figura 17 - Alteração função setup(). Fonte: autor.

```
void loop() {
  mantemConexao();
  MQTT.loop();
}
```

Figura 18 - Alteração função loop(). Fonte: autor.

```
void recebePacote(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;
  }

  if(msg == "0"){
    digitalWrite(pinRele, LOW);
  }
  if(msg == "1"){
    digitalWrite(pinRele, HIGH);
  }
}
```

Figura 19 - Função receber pacote. Fonte: autor.

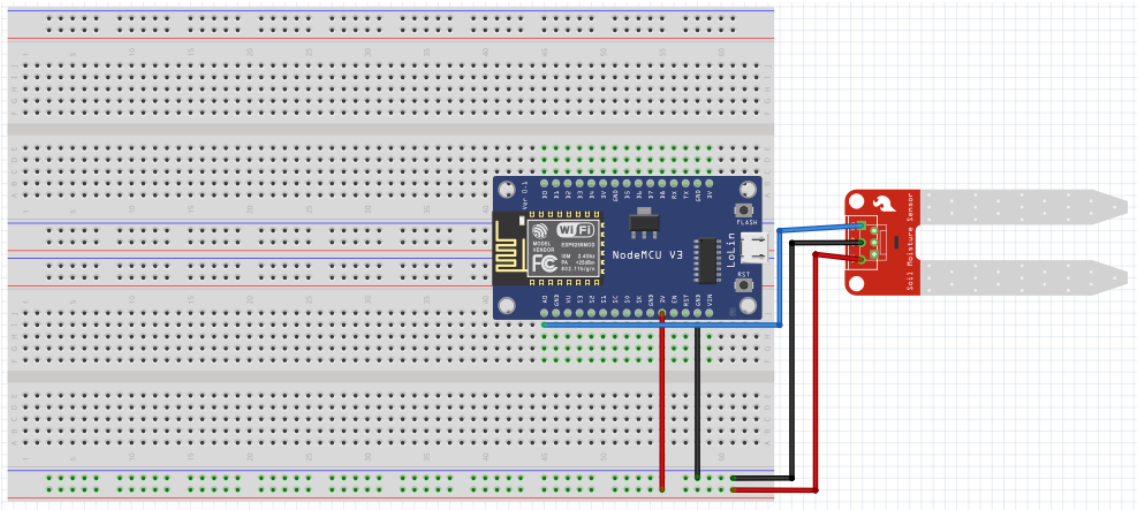

Figura 20 - Projeto sensor de umidade do solo. Fonte: autor.

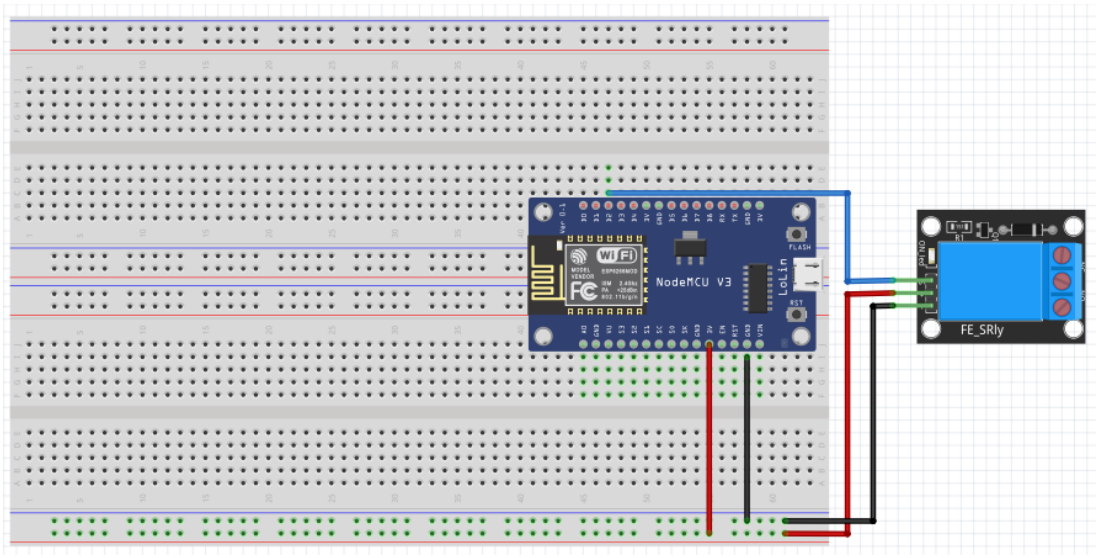

Figura 21 - Projeto ligar/desligar relé. Fonte: autor.

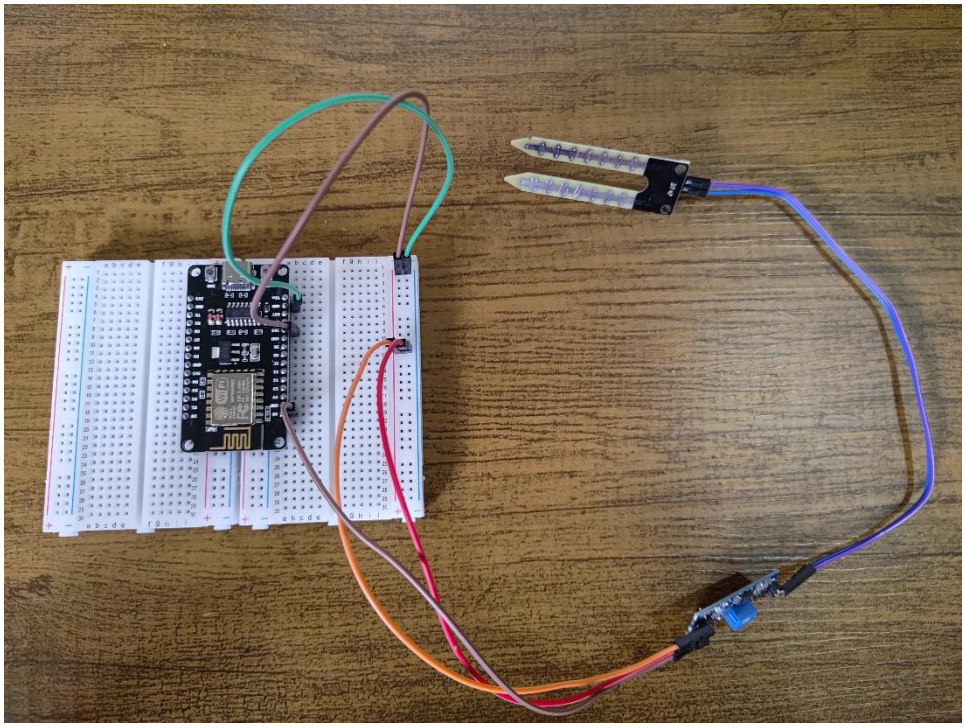

Figura 22 - Montagem do projeto sensor de umidade do solo. Fonte: autor.

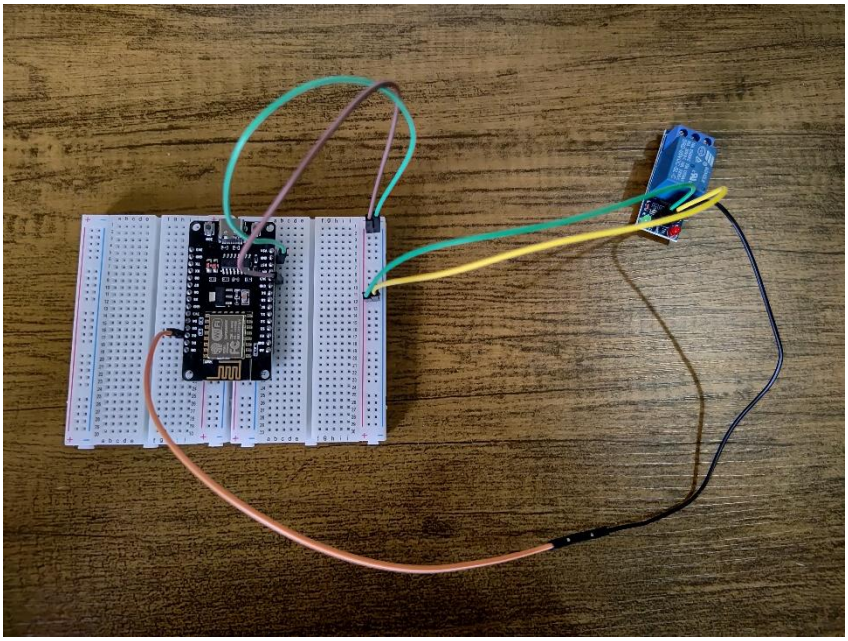

Figura 23 - Montagem do projeto ligar/desligar relé. Fonte: autor.

Para conferir a montagem e funcionamento do projeto completo junto com os hardwares e o código desenvolvido para o projeto apresentado, segue o link do vídeo: “https://www.youtube.com/watch?v=-FiEcneWgq8”.

O código apresentado no projeto está disponível no GITHUB no link: “https://github.com/VladimirSJr/projetoSensorDeUmidade”.

4. CONCLUSÃO

Conforme explicado no vídeo disponibilizado no link anterior, com a falta de alguns componentes necessários para o funcionamento completo do projeto apenas o conjunto que analisa a umidade do solo e envia a informação para ligar ou desligar a bomba de água está completo.

Contudo, é possível sim aplicar o projeto na prática para controlar a irrigação de forma autônoma, com o sensor medindo a umidade e ligando a bomba a distância utilizando o Wi-Fi e um servidor MQTT para realizar a transferência das informações.

No desenvolvimento desse projeto os principais problemas encontrados foram entender o funcionamento dos componentes e como conectá-los, para isso, foram necessárias diversas pesquisas e estudos para compreender e desenvolver a ideia.

5. REFERÊNCIAS

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **Indicador 6.4.1 – alteração da eficiência no uso da água ao longo do tempo.** *s.d.* Disponível em: https://odsbrasil.gov.br/objetivo6/indicador641. Acesso em: 07 mar. 2025.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. **Objetivo 6 – garantir disponibilidade e manejo sustentável da água e saneamento para todos.** *s.d.* Disponível em: https://odsbrasil.gov.br/objetivo/objetivo?n=6. Acesso em: 07 mar. 2025.

MQTT – Como comunicar 2 dispositivos via internet – Vídeo #7 [por] Flávio Guimarães. [S.l.: s.n], 2018. 1 vídeo (14:10 min). Publicado pelo canal de Brincando com ideias. Disponível em: https://www.youtube.com/watch?v=nWXKUSiEPhY. Acesso em: 01 abr. 2025.

ORACLE. **O que é IOT?** *s.d.* Disponível em: https://www.oracle.com/br/internet-of-things/. Acesso em: 06 mar. 2025.

Sistema de irrigação automática via WIFI – IOT (ESP8266) [por] Elian Vitor. [S.l.: s.n], 2021. 1 vídeo (15:44 min). Publicado pelo canal de Duo Tech Talk. Disponível em: https://www.youtube.com/watch?v=SUotikMeldM. Acesso em: 01 abr. 2025.