

**SESI/SENAI**  
**TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS**

Michel Antônio Vieira

Sandro Pinheiro

Marcelo Pinheiro

**SITUAÇÃO DE APRENDIZAGEM II – ETAPA I**

SANTA CATARINA – SC

2023

Michel Antônio Vieira

Sandro Pinheiro

Marcelo Pinheiro

## **SITUAÇÃO DE APRENDIZAGEM II – ETAPA I**

Trabalho apresentado à disciplina Internet das Coisas, como requisito parcial para obtenção de nota

Tutor: Osvaldo da Silva Neto.

SANTA CATARINA – SC

2023

## **SITUAÇÃO DE APRENDIZAGEM 2 – ETAPA I**

### **1)Tecnologias escolhidas: (explicação da plataforma ThingSpeak)**

#### **a) Descreva a plataforma IOT Thingspeak e cite exemplos de como explorá-la de outra forma.**

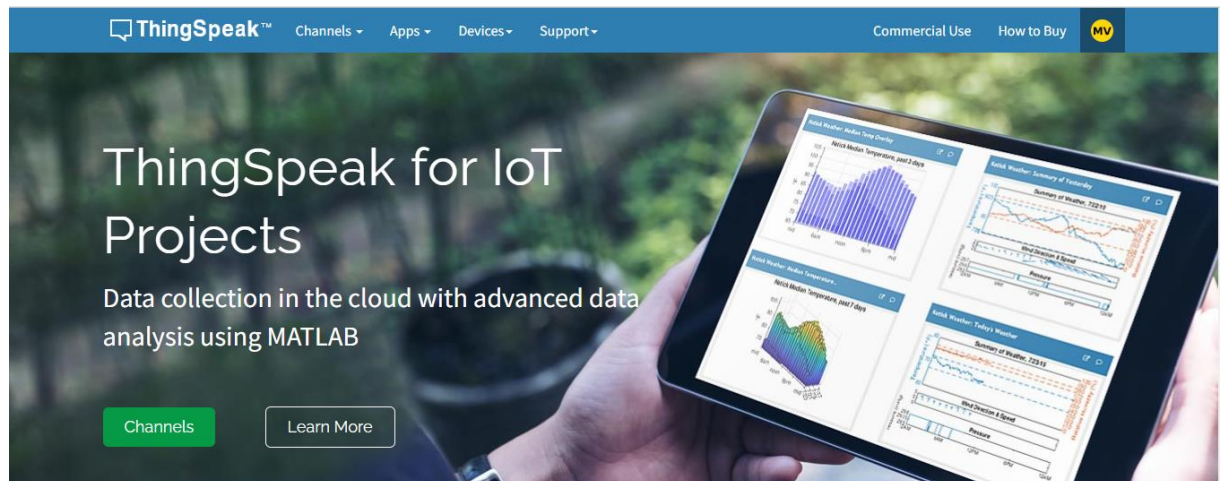
ThingSpeak trata-se de uma plataforma alocada na nuvem que tem por objetivo a coleta, análise e visualização de dados relacionados a dispositivos IoT (Internet das Coisas). A mesma foi criada pela MathWorks, que concebeu a plataforma numa estrutura simplificada para que usuários pudessem realizar tal processo de maneira fácil e ágil, podendo criar canais personalizados para diferentes dispositivos de IoT, visualizar dados em gráficos bem como tabelas em tempo real.

Além disso, pode-se explorar a plataforma de outras formas como o uso de recursos avançados para análise de dados, como cálculos de média, máxima e mínima, filtragem de dados, regras de notificação personalizadas que alertam os usuários sobre a alteração dos dados modificados.

Portanto, o ThingSpeak é uma plataforma muito útil para projetos de IoT em que é necessário coletar, armazenar e analisar dados em tempo real. Seu uso pode ser encontrado em aplicações que incluem o monitoramento ambiental, controle de processos industriais, monitoramento de saúde e agricultura de precisão, entre outros.

**b) documente os passos de configuração e desenvolvimento na integra**

1 – Criar a conta na plataforma ThingSpeak.



2 – Configurar os canais.



3 – Criar o canal.

# My Channels

New Channel

Search by tag



Name ↕	Created ↕	Updated ↕
Monitoramento Sensor <div>Private Public Settings Sharing API Keys Data Import / Export</div>	2023-05-01	2023-05-01 20:14

4 – Criar um nome para o canal a ser utilizado e cadastrar a quantidade de campos que serão puxados do Wokwi. No nosso caso utilizamos apenas dois campos denominados “temperatura” e “umidade”.

## Monitoramento Sensor

Channel ID: 2129836  
Author: mwa000030023584  
Access: Private

Receber e enviar dados para o Arduino com o objetivo de controlar LEDs através do monitoramento do sensor.

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

### Channel Settings

Percentage complete 50%

Channel ID 2129836

Name Monitoramento Sensor

Description objetivo de controlar LEDs através do monitoramento do sensor.

Field 1 temperatura ☒

Field 2 umidade ☒

Field 3 ☐

Field 4 ☐

Field 5 ☐

### Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

### Channel Settings

- **Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Fields:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.

The screenshot shows the 'Channel Settings' page for a ThingSpeak channel. The page has tabs for 'Private View', 'Public View', 'Channel Settings' (selected), 'Sharing', 'API Keys', and 'Data Import / Export'. The channel is named 'Machucados to Device' and has a 'Percentage complete' of 50%. The 'Channel ID' is 2126036. The 'Description' is 'Machucados to Device'. The 'Fields' section has 8 fields, with 'Field 1' and 'Field 2' being active. The 'Metadata' section has a text area for 'Tags'. The 'Link to External Site' is 'http://'. The 'Link to GitHub' is 'https://github.com/'. The 'Elevation' is set to '0.0'. The 'Show Channel Location' section has 'Latitude' and 'Longitude' set to '0.0'. The 'Show Video' section has 'Video URL' set to 'http://'. The 'Show Status' section has a checkbox for 'Show Status'. The 'Save Channel' button is at the bottom. A red arrow points to the 'Save Channel' button.

5 – Acessar o código elaborado na aula 01 e realizar as seguintes alterações.

6 – Incluir as bibliotecas

```
#include <ThingSpeak.h>
```

```
#include <WiFi.h>
```

7 – Definir as credenciais de acesso, no caso da placa Esp32, pode-se utilizar o nome do WiFi como “Wokwi-GUEST” e a senha “” que a placa emula uma conexão com a internet.

```
char* ssid = "Wokwi-GUEST";
char* password = "";
```

8 – Inserir a API de escrita no código para endereçar o caminho para registro da informação do ThingSpeak.

# Monitoramento Sensor

Channel ID: **2129836**

Author: mwa0000030023584

Access: Private

Receber e enviar dados para o Arduino com o objetivo de controlador LEDs através do monitoramento do sensor.

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

+ Add Visualizations

+ Add Widgets

Export recent data

## Channel Stats

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

## Write API Key

Key

TP65HJZ6EYDKFIL2

Generate New Write API Key

## Help

API keys enable you to write data to the channel. API keys are auto-generated when you create a channel.

## API Keys Settings

- **Write API Key:** Use this key to write data to the channel. If this key has been compromised, you should generate a new key.
- **Read API Keys:** Use these keys to read data from the channel. You can use these keys to feed data into charts, or to use in your own applications.

9 - Com a posse do código da API de escrita, basta defini-lo no código.

```
char* apiKey = "TP65HJZ6EYDKFIL2";  
unsigned long channelNumber = 2;
```

10 - Criar o objeto cliente para conexão WiFi.

```
// Inicializar o objeto WiFiClient  
WiFiClient client;
```

11 - na void setup() deve-se criar um loop para tentar realizar a conexão WiFi

```
// Conectar-se à rede Wi-Fi  
WiFi.begin(ssid, password);  
while (WiFi.status() != WL_CONNECTED) {  
    delay(1000);  
    Serial.println("Conectando à rede Wi-Fi...");  
}
```

12 - Agora será necessário inicializar o ThingSpeak

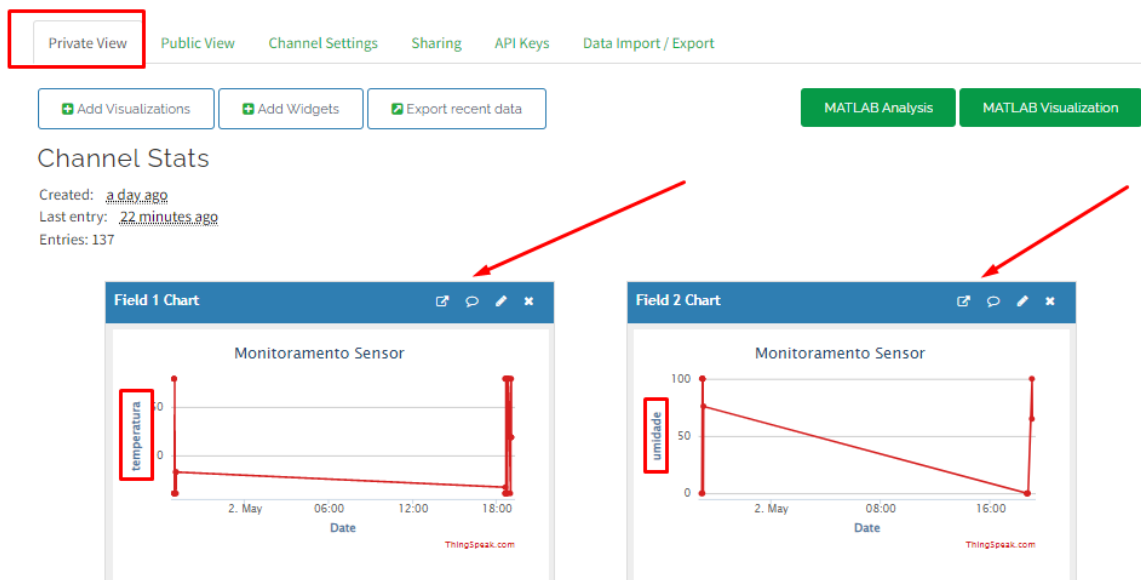
```
// Inicializar o ThingSpeak
```

```
ThingSpeak.begin(client);
```

13 – E por fim criar duas variáveis que receberão as informações do sensor e transmiti-las para o ThingSpeak.

```
float temperatura = dht.readTemperature();  
float umidade = dht.readHumidity();  
  
// Enviar os dados de temperatura e umidade para o ThingSpeak  
ThingSpeak.writeField(channelNumber, 1, temperatura, apiKey);  
ThingSpeak.writeField(channelNumber, 2, umidade, apiKey);
```

14 – Caso tudo ocorra de maneira correta, as informações serão recepcionadas pelo ThingSpeak e serão mostradas na aba Private View



## CÓDIGO NA INTEGRA

```
#include <DHT.h>  
#include <ThingSpeak.h>  
#include <WiFi.h>  
  
#define DHTPIN 4  
#define DHTTYPE DHT22  
  
DHT dht(DHTPIN, DHTTYPE);  
  
int buzzerPin = 23; //ok  
int ledVerde = 18;  
int ledAzul = 5;  
int relayPin = 2;  
int ledAmarelo = 22;  
  
// Definir suas credenciais Wi-Fi e informações do ThingSpeak
```



```

char* ssid = "Wokwi-GUEST";
char* password = "";
char* apiKey = "TP65HJZ6EYDKFIL2";
unsigned long channelNumber = 2;

// Inicializar o objeto WiFiClient
WiFiClient client;

void setup() {
    Serial.begin(9600);
    pinMode(buzzerPin, OUTPUT);
    pinMode(ledVerde, OUTPUT);
    pinMode(ledAmarelo, OUTPUT);
    pinMode(relayPin, OUTPUT);
    pinMode(ledAzul, OUTPUT);
    dht.begin();

    // Conectar-se à rede Wi-Fi
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Conectando à rede Wi-Fi...");
    }

    // Inicializar o ThingSpeak
    ThingSpeak.begin(client);
}

void loop() {
    float temperatura = dht.readTemperature();
    float umidade = dht.readHumidity();

    Serial.print("Temperatura: ");
    Serial.print(temperatura);
    Serial.print("C - Umidade: ");
    Serial.print(umidade);
    Serial.println("%");

    // Enviar os dados de temperatura e umidade para o ThingSpeak
    ThingSpeak.writeField(channelNumber, 1, temperatura, apiKey);
    ThingSpeak.writeField(channelNumber, 2, umidade, apiKey);

    if (temperatura > 35) {
        digitalWrite(ledVerde, HIGH);
        digitalWrite(relayPin, HIGH);
        digitalWrite(ledAmarelo, HIGH);
    } else {
        digitalWrite(ledVerde, LOW);
        digitalWrite(relayPin, LOW);
        digitalWrite(ledAmarelo, LOW);
    }

    if (umidade > 70) {
        digitalWrite(buzzerPin, HIGH);
    }
}

```

```
    digitalWrite(ledAzul, HIGH);  
  } else {  
    digitalWrite(buzzerPin, LOW);  
    digitalWrite(ledAzul, LOW);  
  }  
  
  delay(5000);  
}
```

## **2) Código desenvolvido (6 Pontos)**

**a) O código deverá ser publicado em um repositório público no github.**

**R:** [https://github.com/michelantoniovieira/SA2---trabalho\\_IOT](https://github.com/michelantoniovieira/SA2---trabalho_IOT)