Desenvolvimento do Software (Semana 10–12)

Disciplina: Projeto de Internet das Coisas
Curso: Tecnologia em Big Data no Agronegócio

🎯 Objetivos de Aprendizagem

- Programar o microcontrolador para coleta e envio de dados via rede ou Bluetooth.
- Desenvolver uma interface de visualização usando MIT App Inventor ou outras plataformas.
- Implementar automações e alertas com base nos dados lidos pelos sensores.

🧠 **Conteúdo Programático**

✅ **Semana 10: Coleta e Envio de Dados para a Nuvem ou App**

- Leitura de sensores (ex: DHT11, MQ-135, etc.).
- Envio de dados por WiFi com ESP8266WiFi, HTTPClient ou MQTT.
- Alternativa: envio de dados via Bluetooth para smartphone.
- Teste de conexão com App Inventor.

✅ **Semana 11: Interface de Visualização – MIT App Inventor**

- Introdução à lógica de blocos do MIT App Inventor.
- Criando interfaces móveis: rótulos, gráficos, botões, sliders.
- Comunicação com ESP8266 via Web ou com HC-05 via Bluetooth.
- Visualização em tempo real e controle remoto de atuadores (ex: relé, buzzer).
- Alternativas: dashboards como Blynk, ThingSpeak ou Grafana.

✅ **Semana 12: Automação e Alertas**

- Programação de regras automáticas no microcontrolador: ex. if (temperatura > 35) { ligaVentilador(); }
- Notificações sonoras (buzzer), visuais (LED), mensagens no app ou notificações push.
- Integração com interface no MIT App Inventor: envio de alertas ao app.

👩‍🏫 **Metodologia**

- Demonstração ao vivo da montagem de um app com MIT App Inventor.
- Aprendizagem Baseada em Projeto (ABP): grupos aplicam o que aprendem em seus projetos.
- Aprendizado colaborativo: grupos compartilham avanços e soluções.

🛠️ **Atividades Práticas**

- Configurar envio de temperatura para nuvem (ThingSpeak) **ou** via Bluetooth para o App.
- Criar app no MIT App Inventor com:
  - Leitura de dados em tempo real (ex: temperatura).
  - Botão de controle remoto (ex: ligar bomba d'água).
  - Tela de alerta visual (ex: temperatura fora do ideal).
- Testar automação de ações com base nos dados.
- Documentar e subir o app (.aia) e o código do Arduino no repositório.

📋 **Avaliação**

- Funcionalidade da comunicação entre app e ESP.
- Interface clara e funcional.
- Lógica de automação aplicada corretamente.

- Repositório GitHub com README explicativo.
- Demonstração prática do app funcionando.

📚 **Recursos Didáticos**

- MIT App Inventor (https://appinventor.mit.edu)
- Arduino IDE
- Kit de sensores (DHT11, buzzer, relé, LED, etc.)
- Módulo Bluetooth HC-05 ou módulo WiFi ESP8266
- Plataformas de nuvem opcionais: Blynk, ThingSpeak, Firebase

💡 Exemplo de Dashboard Agrícola

Painel	Tipo	Fonte
Temperatura	Linha	InfluxDB
Umidade do Solo	Linha	InfluxDB
Umidade Relativa	Gauge	InfluxDB
Sensor GPS	Mapa	JSON API
Estado da Bomba	Stat	MySQL

📊 Tabela Comparativa – Grafana vs ThingSpeak vs MIT App Inventor

Critério	**Grafana**	**ThingSpeak IoT**	**MIT App Inventor**
Tipo de ferramenta	Plataforma de visualização e monitoramento	Plataforma de IoT com armazenamento em nuvem	Plataforma de desenvolvimento de apps Android
Papel no ecossistema IoT	Criação de dashboards avançados para análise	Armazenamento, visualização básica e alertas	Interface móvel para controle/monitoramento
Fonte de dados	Externa (InfluxDB, MySQL, MQTT, etc.)	Própria (canal ThingSpeak)	Dados do app ou comunicação direta com sensores
Visualização de dados	Alta qualidade, interativo, personalizável	Gráficos simples, com histórico	Interface de usuário simples com botões/labels
Alertas	Avançados (condições complexas, multi-canal)	Simples (via ThingHTTP/IFTTT, baseado em limites)	Possível por lógica de programação com blocos
Controle de dispositivos	Via integração externa (API, MQTT, Webhook)	Via integração com ThingHTTP/IFTTT	Controle direto via Bluetooth ou Wi-Fi
Requer programação?	Sim, para consultas e integração com banco	Pouco (Arduino + HTTP ou MQTT)	Programação visual por blocos
Ideal para	Visualização e análise de grandes volumes de dados	Monitoramento remoto simples e projetos educacionais	Interfaces móveis simples e interativas para IoT
Dificuldade de uso	Média a alta	Baixa a média	Baixa (foco educacional)
Requisitos de hardware	Servidor próprio ou nuvem	Dispositivo com acesso à internet (ESP8266, etc)	Android + módulo Bluetooth/Wi-Fi
Plataforma/ Dispositivo alvo	Web	Web (nuvem)	Aplicativo Android
Custo	Gratuito / planos pagos para Grafana Cloud	Gratuito com limitações	Gratuito
Exemplo de uso no agro	Painel com sensores de solo, temperatura, mapa	Medição de umidade e temperatura em tempo real	Controle de irrigação via app Android com botão

🧠 **Considerações:**

- **Use o Grafana** quando quiser painéis profissionais com múltiplos sensores e análises.
- **Use o ThingSpeak** se precisar de um sistema de coleta simples, rápido e com suporte a MATLAB.
- **Use o MIT App Inventor** se quiser uma **interface direta com o usuário**, ideal para acionar ou monitorar sensores por celular.

📊 **Tabela Comparativa – Grafana vs ThingSpeak vs MIT App Inventor vs Blynk IoT**

Critério	Grafana	ThingSpeak IoT	MIT App Inventor	Blynk IoT
Tipo de ferramenta	Plataforma de visualização e análise	Plataforma de IoT com armazenamento em nuvem	Criador de apps Android com blocos visuais	Plataforma completa de IoT com app móvel e web
Papel no ecossistema IoT	Dashboards avançados e monitoramento	Armazenamento e visualização básica de dados	Interface de usuário para controle via celular	Visualização, controle e automação remota
Fonte de dados	Bancos externos (InfluxDB, MySQL, etc.)	Banco interno por canal	Dados locais no app ou por comunicação direta	Servidores Blynk (nuvem ou local)
Visualização de dados	Painéis ricos, personalizáveis	Gráficos simples, históricos	Interface simples com botões, rótulos, sliders	Visual moderno com gráficos, gauges, controles
Alertas	Avançados (multi-condições, canais diversos)	Básicos (via React e ThingHTTP)	Programação condicional básica	Notificações push e eventos com automações visuais
Controle de dispositivos	Externo via APIs ou scripts	ThingHTTP e integrações	Via Bluetooth/Wi-Fi com comandos por botão	Controle direto com widgets interativos
Requer programação?	Sim (consultas, integrações)	Baixo a médio (Arduino + API)	Não (blocos de programação visual)	Pouco – Widgets e templates prontos, lógica opcional
Ideal para	Grandes volumes de dados e análise histórica	Projetos educacionais e protótipos simples	Interface de apps móveis para projetos simples	Apps móveis/web funcionais para controle e monitoramento
Dificuldade de uso	Média a alta	Baixa a média	Muito baixa	Muito baixa a média (com app pronto)
Requisitos de hardware	Servidor próprio/nuvem + banco externo	Dispositivo com internet (ESP, Arduino)	Smartphone Android com Bluetooth ou Wi-Fi	ESP32, ESP8266, Arduino + conexão Wi-Fi
Plataforma/	Web	Web (dashboard	Android	Android, iOS e

Dispositivo alvo		online)	(aplicativo)	Web Dashboard
Custo	Gratuito com limite / versão cloud paga	Gratuito (limites de uso)	Gratuito	Gratuito com plano inicial / planos premium disponíveis
Exemplo de uso no agro	Monitorar sensores em lavoura, mapa de campo	Temperatura e umidade de solo com alertas	Controle de bomba de irrigação via botão no app	Monitorar e controlar sensores em tempo real com celular

✅ **Destaque dos usos combinados:**

Solução	Melhor para...
Grafana + InfluxDB	Grandes projetos com análise e dashboards
ThingSpeak + Arduino	Projetos rápidos e protótipos educacionais
App Inventor + Bluetooth	Interação local e educativa via celular
Blynk + ESP32	Projetos robustos com controle remoto móvel

🔧 Ferramentas e Plataformas Relevantes para Projetos de IoT

Nome da Ferramenta	Descrição resumida	Tipo / Uso principal
Node-RED	Ambiente de programação visual para conectar dispositivos, APIs e serviços com fluxos lógicos.	Integração e automação
Firebase (Realtime DB)	Banco de dados em tempo real da Google, ideal para apps móveis interagirem com dispositivos IoT.	Armazenamento, controle e notificações
Home Assistant	Plataforma de automação residencial (e agrícola), com suporte a sensores, atuadores e dashboards.	Automação e controle local ou remoto
Cayenne	Plataforma IoT com interface gráfica para controle de sensores, gráficos e regras de automação.	Monitoramento e automação
Kaa IoT Platform	Plataforma escalável para integração de dispositivos IoT industriais e de grande porte.	Nuvem, gerenciamento de dispositivos
OpenHAB	Plataforma de automação open-source voltada para casas inteligentes, com suporte a dispositivos IoT.	Automação e dashboards
Ubidots	Plataforma de visualização e gerenciamento de dispositivos IoT com dashboards customizáveis.	Dashboards e análise de dados
Zabbix	Sistema de monitoramento de redes e dispositivos, adaptável para ambientes de IoT.	Monitoramento técnico e alertas
Mosquitto MQTT Broker	Servidor MQTT leve, ideal para comunicação entre dispositivos IoT de forma rápida e confiável.	Comunicação de dados
IFTTT (If This Then That)	Plataforma de automação de tarefas baseada em regras condicionais simples.	Ações automatizadas e notificações
Plotly Dash	Framework Python para construir dashboards interativos com dados IoT em tempo real.	Visualização avançada
Arduino IoT Cloud	Plataforma oficial da Arduino para controle e visualização remota de dispositivos conectados.	Nuvem + automação com ESP/Arduino
ThingsBoard	Plataforma open-source para gerenciamento, visualização e automação de dispositivos IoT.	Nuvem, dashboards e gerenciamento de devices

---

✅ **Sugestões de Combinações para Projetos no Agronegócio**

- **ESP32 + Node-RED + Grafana** → sistema local com visualização e automação
- **ESP8266 + Firebase + MIT App Inventor** → app móvel com dados em tempo real
- **Arduino + MQTT + Cayenne/Ubidots** → controle remoto com alertas por eventos
- **Arduino + Blynk + IFTTT** → solução rápida com app e automações integradas

- **Sensor de Solo + Arduino IoT Cloud** → plataforma oficial com dashboards prontos

Tabela Comparativa de Ferramentas IoT

Ferramenta	Descrição	Categoria Principal	Nível de Complexidade
Plotly Dash	Framework Python para visualizações interativas	Visualização	Alto
Arduino IoT Cloud	Plataforma da Arduino para visualização e controle	Visualização e Controle	Baixo
ThingsBoard	Plataforma open-source para gerenciamento de IoT	Nuvem e Gerenciamento	Alto

🧰 Outras Ferramentas e Plataformas para Projetos de IoT

Nome da Ferramenta	Descrição resumida	Tipo / Finalidade
Balena	Plataforma para deploy e gerenciamento remoto de dispositivos embarcados (Linux).	Gerenciamento de dispositivos IoT
Kibana	Visualização de dados da pilha ELK (Elasticsearch) com dashboards interativos.	Visualização e análise de logs/dados IoT
Domoticz	Plataforma leve de automação residencial com suporte a sensores e scripts.	Automação local
IoTivity	Framework open-source promovido pela Linux Foundation para padronizar IoT.	Integração e interoperabilidade
Tasmota	Firmware leve para ESP8266/ESP32 via MQTT e webserver, sem necessidade de codar.	Controle e comunicação via MQTT/Web
ESPHome	Firmware personalizável para ESPs, integrável ao Home Assistant, com YAML.	Controle de sensores/atuadores com automações
OpenRemote	Plataforma open-source para gerenciamento de IoT e edifícios inteligentes.	Plataforma de controle centralizado
Thingspeak MATLAB App	Extensão para análise mais profunda dentro do próprio ThingSpeak.	Análise e modelagem matemática
Microsoft Azure IoT Hub	Plataforma robusta para integrar, controlar e escalar dispositivos IoT.	Nuvem corporativa / gerenciamento em escala
AWS IoT Core	Serviço da Amazon Web Services para conectar e gerenciar dispositivos IoT.	Plataforma em nuvem para projetos de larga escala
Google Cloud IoT Core	(Descontinuado em 2023, mas ainda presente em projetos)	Plataforma de nuvem para dados de IoT

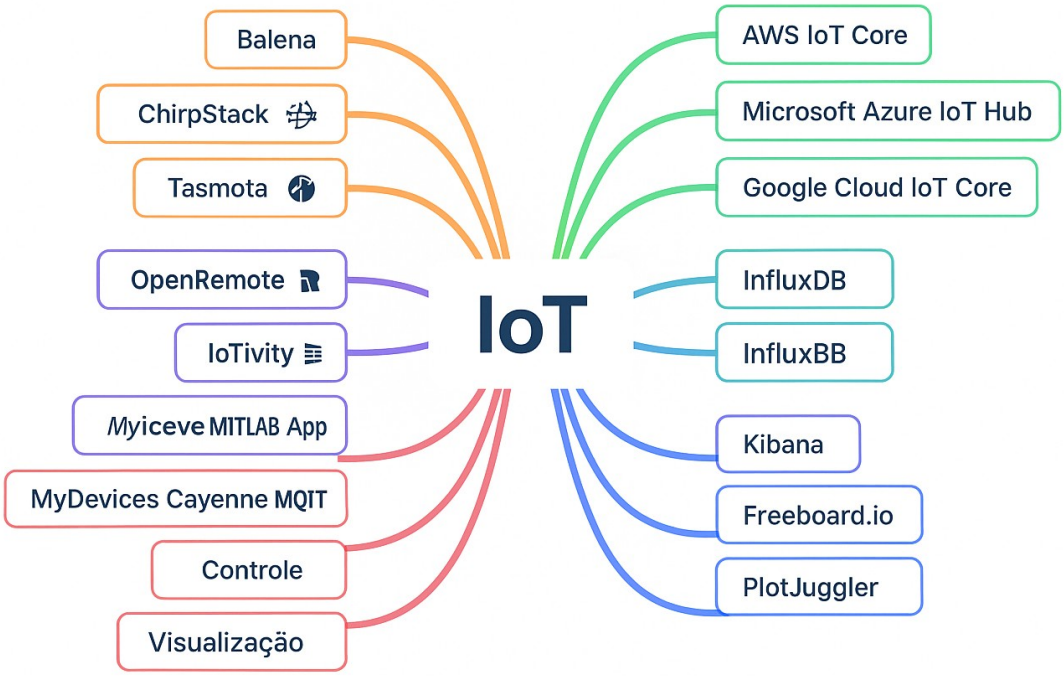
Nome da Ferramenta	Descrição resumida	Tipo / Finalidade
ChirpStack	Servidor LoRaWAN open-source para redes de sensores de longo alcance.	Comunicação LoRaWAN / Redes de sensores
MyDevices Cayenne MQTT	Extensão MQTT da Cayenne para dispositivos personalizados.	Comunicação leve com visualização via widgets
Freeboard.io	Dashboard web simples e open-source para visualização de dados em tempo real.	Visualização básica para protótipos rápidos
PlotJuggler	Ferramenta para análise de séries temporais e dados de sensores (ROS e IoT).	Análise e debug de dados em tempo real
InfluxDB	Banco de dados otimizado para séries temporais – ideal para sensores IoT.	Armazenamento de dados
Telegraf	Agente de coleta de métricas/sensores para InfluxDB (compatível com MQTT, JSON).	Coleta de dados de sensores e brokers
Eclipse Kura	Middleware de gateway IoT embarcado com suporte a protocolos industriais.	Edge computing / gateways industriais
Eclipse Ditto	Projeto que cria "gêmeos digitais" de dispositivos IoT.	Digital Twin / Sincronização

🔍 **Observações**

- Algumas dessas ferramentas são ideais para **prototipagem educacional** (como o Freeboard.io, ESPHome, Tasmota).
- Outras se encaixam melhor em **ambientes industriais ou corporativos** (como AWS IoT, Azure IoT Hub, Eclipse Kura).
- Também há muitas que trabalham **em conjunto** (ex: ESP com Tasmota → MQTT → Telegraf → InfluxDB → Grafana).

🎨 **Significado das Cores no Mapa Mental**

Cor	Categoria Representada	Exemplos no mapa
🔵 Azul	**Visualização**	Grafana, Kibana, Freeboard.io, PlotJuggler
🔴 Vermelho	**Controle / Interface**	MIT App Inventor, MyDevices Cayenne MQTT
🟣 Roxo	**Automação / Integração**	OpenRemote, IoTivity
🟠 Laranja	**Comunicação / Firmware**	Tasmota, ChirpStack, Balena
🟢 Verde	**Plataformas de Nuvem (Cloud)**	AWS IoT Core, Azure IoT Hub, Google Cloud IoT, InfluxDB


IoT
Balena
ChirpStack
Tasmota
OpenRemote
IoTivity
Myiceve MITLAB App
MyDevices Cayenne MQIT
Controle
Visualização
AWS IoT Core
Microsoft Azure IoT Hub
Google Cloud IoT Core
InfluxDB
InfluxBB
Kibana
Freeboard.io
PlotJuggler

🌐 O que é o ThingSpeak?

ThingSpeak é uma plataforma online de código aberto para **armazenamento, visualização e análise de dados** provenientes de dispositivos IoT (Internet das Coisas). Foi desenvolvida inicialmente pela IoTWorks e atualmente é mantida pela **MathWorks**, a mesma empresa responsável pelo MATLAB.

Ela permite que dispositivos como **Arduino, ESP8266, ESP32** e outros microcontroladores conectados à internet possam **enviar dados para a nuvem**, onde esses dados podem ser **armazenados, visualizados em tempo real e analisados**.

🔧 Funcionalidades principais

1. **Canal de Dados (Channels):**
   - Cada canal suporta até **8 campos de dados** (ex: temperatura, umidade, pH etc.).
   - Armazena valores enviados por sensores com carimbo de data e hora.
   - Permite adicionar metadados e localização GPS.
2. **Visualização de Dados:**
   - Geração automática de **gráficos de linha, barra, medidores** e outras formas de visualização.
   - Atualização em tempo real com novos dados recebidos.
3. **Processamento com MATLAB:**
   - Permite usar scripts MATLAB diretamente na nuvem para **análise estatística, filtragem, predição e automações**.
4. **Ações baseadas em dados (React):**
   - É possível configurar **alertas e automações** quando certos valores forem atingidos.
   - Integrações com **Twitter, Email, ThingHTTP, MQTT**.
5. **APIs REST e MQTT:**
   - Pode enviar e receber dados usando **HTTP GET/POST** ou **protocolo MQTT**.
   - Fácil integração com **ESP8266**, **ESP32**, **Arduino Ethernet/WiFi Shield**, entre outros.

📈 Exemplo de uso em IoT no Agronegócio

Situação: Monitoramento de temperatura e umidade em uma estufa de morangos.

- O sensor DHT11 é conectado a um ESP8266.
- A cada 5 minutos, o ESP lê a temperatura e envia para um canal ThingSpeak usando HTTP.
- A plataforma gera gráficos em tempo real e, se a temperatura ultrapassar 35 °C:
  - Um **alerta por e-mail** é enviado.
  - Um comando pode ser enviado de volta ao ESP para **acionar um ventilador ou irrigação**.

💡 Vantagens do ThingSpeak

- **Gratuito** (até 3 milhões de mensagens por ano na conta free).
- Interface intuitiva.
- Ideal para projetos educacionais e protótipos rápidos.
- Integração com **MATLAB Analytics**.
- Fácil de configurar e usar com microcontroladores populares.

📌 Limitações da versão gratuita

- Apenas **1 mensagem a cada 15 segundos por canal** (limite de taxa).
- Até **4 canais privados/públicos**.
- Sem suporte técnico prioritário.

📘 Modelo de Canal ThingSpeak

Objetivo: Monitorar dados de temperatura e umidade e acionar alertas automatizados no contexto de IoT para o Agronegócio.

1. 🏷️ Nome do Canal

Monitoramento Ambiental – Estufa de Tomate

2. 📝 Descrição

Canal criado para receber, armazenar e visualizar dados de sensores de temperatura e umidade em tempo real, com base em leituras feitas por um microcontrolador ESP8266. Utilizado para fins de análise ambiental e controle automático de sistemas de ventilação e irrigação.

3. 🌡️ Campos do Canal

Nº	Nome do Campo	Descrição	Unidade
1	Temperatura	Temperatura interna da estufa	°C
2	Umidade	Umidade relativa do ar na estufa	%
3	Estado do Ventilador	0 = Desligado / 1 = Ligado	Binário
4	Estado da Irrigação	0 = Desligado / 1 = Ligado	Binário

4. 🌐 Privacidade

- ✅ Tornar o canal **privado** (padrão), ou
- 🔓 Tornar o canal **público** se quiser mostrar os dados para terceiros ou visitantes sem login.

5. 📍 Localização

- Latitude e Longitude da fazenda/estufa (opcional).
- Exemplo:
  - **Latitude**: -21.12345
  - **Longitude**: -47.98765

6. 🔗 Tags (palavras-chave)

iot, estufa, temperatura, umidade, big data, agronegócio, esp8266

7. 🧪 Opções Avançadas

a) Notificações e Reações (React)

- Crie uma **Trigger** para enviar um alerta:
  - Se **Temperatura > 35 °C**, enviar notificação via ThingHTTP ou Tweet.
  - Se **Umidade < 40%**, acionar sistema de irrigação.

b) Integrações

- Integre com **MATLAB Analysis** para análises mais complexas.
- Use **ThingHTTP + IFTTT** para enviar notificações para o celular ou e-mail.

8. 🔐 Chaves de API

Você deve guardar estas duas chaves com cuidado:

- **Write API Key**: usada para **enviar** dados do ESP8266.

- **Read API Key**: usada para **ler** dados (em caso de dashboards externos).

Exemplo de chave fictícia:

Write API Key: ABCD1234EFGH5678

9. 📈 Visualizações Sugeridas

- Gráfico de linha para Temperatura e Umidade.
- Medidores (gauges) para valores instantâneos.
- Tabela de registros históricos.
- Mapa com localização (se ativado).

10. 📲 Link para Acesso (exemplo)

https://thingspeak.com/channels/123456

Exemplo de código Arduino para ESP8266 (NodeMCU ou Wemos D1) que envia **temperatura e umidade** do sensor **DHT11** para um **canal ThingSpeak** com os campos:

- **Campo 1** – Temperatura (°C)
- **Campo 2** – Umidade (%)

✅ 1. Bibliotecas Necessárias

Antes de rodar o código, instale estas bibliotecas no Arduino IDE:

- ESP8266WiFi (já vem com o core do ESP8266)
- DHT sensor library (by Adafruit)
- ThingSpeak (by MathWorks)

Vá em: **Sketch > Include Library > Manage Libraries**, e procure por elas.

📦 2. Materiais

- ESP8266 (ex: NodeMCU)
- Sensor DHT11 ou DHT22
- Resistor de 10kΩ (se necessário para o DHT11)
- Jumpers
- Acesso Wi-Fi

💻 3. Código Arduino e comentários

```
#include <ESP8266WiFi.h>
#include "DHT.h"
#include "ThingSpeak.h"
// Credenciais Wi-Fi
const char* ssid = "SEU_WIFI";
const char* password = "SENHA_DO_WIFI";

// Configuração do ThingSpeak
unsigned long myChannelNumber = 1234567; // 🔁 Substitua pelo número do seu canal
const char* myWriteAPIKey = "SUA_WRITE_API_KEY"; // 🔁 Substitua pela sua chave

WiFiClient client;

// Sensor DHT
#define DHTPIN D4       // Pino conectado ao DHT (ex: GPIO2 no NodeMCU)
#define DHTTYPE DHT11   // Troque para DHT22 se for o caso
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(115200);
  dht.begin();

  WiFi.begin(ssid, password);
```

```
  Serial.print("Conectando ao Wi-Fi");
   while (WiFi.status() != WL_CONNECTED) {
   delay(1000);
   Serial.print(".");
  }
   Serial.println("\nWi-Fi conectado!");
  ThingSpeak.begin(client);
}
void loop() {
  float temperatura = dht.readTemperature();
  float umidade = dht.readHumidity();
  // Verificação de leitura válida
  if (isnan(temperatura) || isnan(umidade)) {
   Serial.println("Falha ao ler o sensor DHT!");
   return;
  }
  Serial.println("Enviando para ThingSpeak...");
  Serial.print("Temperatura: "); Serial.print(temperatura); Serial.println(" °C");
  Serial.print("Umidade: "); Serial.print(umidade); Serial.println(" %");
  // Envio dos dados para os campos do canal
  ThingSpeak.setField(1, temperatura);
  ThingSpeak.setField(2, umidade);
  // Grava os dados no canal
  int response = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  if (response == 200) {
   Serial.println("Dados enviados com sucesso!");
  } else {
   Serial.print("Erro ao enviar: ");
   Serial.println(response);
  }
  delay(15000); // Espera de 15s (limite da conta gratuita)
}
```

🧪 **4. Testes**

1. Abra o **Monitor Serial** (baud 115200) para acompanhar o envio.
2. No ThingSpeak, vá até o canal → "Private View" → veja os gráficos atualizando.
3. Se quiser testar localmente, pode abrir também um dashboard com Node-RED ou Blynk.

✅ Como Criar um Canal no ThingSpeak

🔗 Acesse a plataforma

1. Vá até: https://thingspeak.com
2. Clique em **Sign In** (se já tiver conta) ou em **Sign Up** (para criar uma conta gratuita da MathWorks).

> 💡 Dica: o ThingSpeak usa a mesma conta do MATLAB/MathWorks. Pode usar uma conta Google para facilitar.

🧭 Criando o Canal

1. Após login, clique no menu **Channels > My Channels**
2. Clique em **New Channel**

Preenchendo os dados do canal

Preencha os campos do formulário:

Campo	O que colocar
Name	Ex: Monitoramento Estufa
Description	Ex: Monitoramento de temperatura e umidade em tempo real.
Field 1	Marque e coloque Temperatura (°C)
Field 2	Marque e coloque Umidade (%)
Field 3 até 8	(opcional) campos adicionais
Latitude/ Longitude	(opcional) localização geográfica da fazenda ou estufa
Elevation	Altitude (opcional)
Metadata	(opcional) para informações técnicas
Tags	Ex: iot, temperatura, umidade, agronegócio, esp8266
URL do Imagem	(opcional) link de imagem do projeto (ex: logo ou foto da fazenda)

🔒 Privacidade

- **Public**: qualquer pessoa pode ver o canal.
- **Private (padrão)**: apenas você verá os dados.

Pode mudar depois em *Channel Settings*.

✅ Finalizar

- Clique em **Save Channel** no final da página.

🔐 Obtenha suas Chaves API

1. Após criar, vá até a aba **API Keys**
2. Copie a chave de escrita (Write API Key) — será usada no Arduino.
    - Exemplo: AB12C34D5E6F7G8H
3. Também há uma **Read API Key** (para apps lerem os dados).

📊 Visualizando os dados

1. Acesse a aba **Private View** ou **Public View**
2. Os gráficos aparecem automaticamente com os dados recebidos.

🌱 Neste exemplo pode-se:

- Enviar dados do Arduino/ESP para o canal.
- Ver gráficos de temperatura/umidade.
- Configurar **Reações (React)** para alertas.
- Usar **ThingHTTP + IFTTT** para acionar mensagens, apps ou e-mails.

O que é o Grafana.

O ChatGPT disse:

O **Grafana** é uma plataforma de código aberto usada para **visualização e monitoramento de dados** em tempo real. Ele permite a criação de **dashboards interativos** que exibem dados de várias fontes, como bancos de dados SQL, APIs, sistemas de IoT, servidores, sensores e muito mais.

🧠 **Principais características do Grafana:**

- **Visualização em tempo real**: gráficos de linhas, gauges, mapas de calor, tabelas e outros.
- **Dashboards personalizáveis**: crie painéis completos com filtros, alertas e múltiplas fontes de dados.
- **Fontes de dados variadas**: integra com **InfluxDB**, **MySQL**, **PostgreSQL**, **Prometheus**, **Elasticsearch**, **MQTT**, **CSV**, e muito mais.
- **Alertas**: envia notificações por e-mail, Slack, Telegram ou outras plataformas quando determinados valores ultrapassam limites definidos.
- **Multiusuário**: permite colaboração em equipe com controle de permissões.

🌾 **Aplicações no Agronegócio com IoT**

No contexto agrícola, o Grafana pode ser utilizado para:

- Monitorar **temperatura, umidade do solo, luminosidade**, etc.
- Visualizar o histórico de dados captados por sensores.
- Criar painéis de controle para **estufas, irrigação, armazenamento, silos**.
- Gerar alertas em caso de condições críticas (ex: solo seco, temperatura elevada).

🛠️ **Exemplo de uso prático**

Um **ESP8266** envia dados de sensores para um banco de dados **InfluxDB**. O Grafana, conectado ao InfluxDB, exibe gráficos em tempo real da **temperatura e umidade da plantação**, com alertas visuais e notificações automáticas.

📍 PASSO A PASSO – Criar Dashboard no Grafana para IoT Agrícola

✅ 1. Acesse o Grafana

Você tem duas opções:

a) Usar o Grafana Cloud (mais fácil para começar)

- Acesse: https://grafana.com
- Clique em **Start for Free**
- Crie uma conta com email ou GitHub/Google
- Após login, será criado um **workspace gratuito**

b) Instalar localmente (avançado)

- Baixe e instale o Grafana: https://grafana.com/grafana/download
- Após instalação, acesse: http://localhost:3000
    - Login padrão: **admin / admin**

✅ 2. Criar uma Fonte de Dados (Data Source)

O Grafana não armazena dados, ele só **lê de bancos externos**. Exemplos comuns para IoT:

Banco de Dados	Tipo
InfluxDB	Para séries temporais (ideal para sensores)
MySQL/PostgreSQL	Dados tabulares
Prometheus	Métricas em tempo real
MQTT + Telegraf	Para dados de sensores MQTT
CSV ou JSON	Usando plugins ou APIs REST

Exemplo com InfluxDB:

1. Vá até o menu lateral → **Connections > Data Sources**
2. Clique em **InfluxDB**
3. Preencha:
    - **URL**: http://localhost:8086 (ou da nuvem)
    - **Database name**: nome do seu banco
    - **Token/Usuário e senha**, se necessário
4. Clique em **Save & Test**

✅ 3. Criar um Novo Dashboard

1. Menu esquerdo → clique em **+ Create > Dashboard**
2. Clique em **Add New Panel**
3. Configure:
    - Escolha a **fonte de dados**
    - Insira a **consulta** (ex: SELECT temperature FROM sensors WHERE time > now() - 24h)
    - Escolha o **tipo de gráfico** (linha, gauge, barra, mapa)

✅ 4. Personalize os Painéis

- Para cada painel:
    - Defina **título** (ex: "Temperatura")
    - Escolha **cor**, **alertas visuais**, limites (ex: min=10, max=40)
    - Adicione **descrições e ícones**
- Pode usar:
    - **Line Chart** para temperatura/umidade

    - **Gauge** para indicadores rápidos
    - **Image/Map** com localização de sensores
    - **Stat Panel** para mostrar valores fixos

✅ **5. Adicionar Alertas**

1. No painel → clique em **Alert > Create Alert Rule**
2. Configure a condição (ex: temperatura > 35)
3. Defina o canal de notificação (Slack, e-mail, Telegram, Webhook)
    - Vá em **Alerting > Contact points** para configurar

✅ **6. Salvar e Compartilhar**

- Clique em **Save Dashboard**
- Pode compartilhar:
    - Link direto (público ou com login)
    - Incorporar em páginas web (iframe)
    - Exportar como PDF/PNG