# Projeto Integrador - IoT

Tema: Monitoramento Ambiental e Controle Remoto em Salas de Aula Inteligentes

## Objetivo Geral

Desenvolver um sistema de monitoramento ambiental para salas de aula, capaz de registrar temperatura e umidade em tempo real, além de acionar um LED de forma automatizada ou remota, promovendo conforto térmico e eficiência energética dentro do conceito de Cidades Inteligentes.

## Objetivos Específicos

- Integrar sensores de temperatura e umidade ao ESP32.

- Publicar os dados em tempo real em uma planilha Google Sheets.

- Exibir os dados de forma gráfica para análise.

- Acionar um LED em situações críticas ou de forma manual.

- Aplicar práticas de prototipagem com protoboard, resistores e LEDs.

## Recursos de IoT Utilizados

• ESP32: Microcontrolador Wi-Fi responsável pela comunicação e controle

• Sensor (DHT11/DHT22): Coleta dados de temperatura e umidade

• LED + Resistor: Sinalização visual (atuador)

• Protoboard + Jumpers: Montagem dos circuitos

• Google Sheets: Armazenamento e visualização dos dados em nuvem

• Internet / Wi-Fi: Conexão para envio de dados

## Resultados Esperados

- Dados publicados em tempo real.

- Gráficos atualizados automaticamente na nuvem.

- LED acendendo em situações específicas (por exemplo, se temperatura > 30°C).

- Base para futuras automações na escola.

## Publicação de Dados com Google Sheets

1. Criar uma nova planilha e acessar Extensões > Apps Script.

2. Substituir o código existente por:

function doGet(e) {  
 var sheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("Dados");  
 if (!sheet) {  
 sheet = SpreadsheetApp.getActiveSpreadsheet().insertSheet("Dados");  
 sheet.appendRow(["Data", "Hora", "Temperatura", "Umidade", "Nome"]);  
 }  
 var temperatura = e.parameter.temperatura;  
 var umidade = e.parameter.umidade;  
 var nome = e.parameter.nome;  
 var data = Utilities.formatDate(new Date(), "GMT-3", "dd/MM/yyyy");  
 var hora = Utilities.formatDate(new Date(), "GMT-3", "HH:mm:ss");  
 sheet.appendRow([data, hora, temperatura, umidade, nome]);  
 return ContentService.createTextOutput("Dados recebidos com sucesso");  
}

3. Salvar e implantar como aplicativo da web, permitindo acesso público.

4. Copiar a URL gerada para ser usada no ESP32.

## Código Arduino - Envio de Dados

#include <WiFi.h>  
#include <HTTPClient.h>  
  
const char\* ssid = "SUA\_REDE\_WIFI";  
const char\* password = "SENHA\_WIFI";  
  
String nome = "Ana Luisa";  
const char\* serverName = "https://script.google.com/macros/s/SEU\_LINK\_AQUI/exec";  
  
void setup() {  
 Serial.begin(115200);  
 WiFi.begin(ssid, password);  
 while (WiFi.status() != WL\_CONNECTED) {  
 delay(1000);  
 Serial.println("Conectando ao WiFi...");  
 }  
 Serial.println("Conectado!");  
}  
  
void loop() {  
 if (WiFi.status() == WL\_CONNECTED) {  
 HTTPClient http;  
 String temperatura = "28.5";  
 String umidade = "65";  
 String url = String(serverName) + "?temperatura=" + temperatura + "&umidade=" + umidade + "&nome=" + nome;  
 http.begin(url);  
 int httpResponseCode = http.GET();  
 if (httpResponseCode > 0) {  
 String response = http.getString();  
 Serial.println("Resposta: " + response);  
 } else {  
 Serial.print("Erro na requisição: ");  
 Serial.println(httpResponseCode);  
 }  
 http.end();  
 }  
 delay(10000);  
}