Prog a humidité + temp avec esp

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include "DHT.h"

// Uncomment one of the lines bellow for whatever DHT sensor type you're using!

//#define DHTTYPE DHT11 // DHT 11

//#define DHTTYPE DHT21 // DHT 21 (AM2301)

#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321

// Change the credentials below, so your ESP8266 connects to your router

const char\* ssid = "CCF\_Montgolfiere";

const char\* password = "";

// Change the variable to your Raspberry Pi IP address, so it connects to your MQTT broker

const char\* mqtt\_server = "192.168.107.136";

// Initializes the espClient. You should change the espClient name if you have multiple ESPs running in your home automation system

WiFiClient espClient;

PubSubClient client(espClient);

// DHT Sensor - GPIO 5 = D1 on ESP-12E NodeMCU board

const int DHTPin = 5;

// Initialize DHT sensor.

DHT dht(DHTPin, DHTTYPE);

// Timers auxiliar variables

long now = millis();

long lastMeasure = 0;

// Don't change the function below. This functions connects your ESP8266 to your router

void setup\_wifi() {

delay(10);

// We start by connecting to a WiFi network

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.print("WiFi connected - ESP IP address: ");

Serial.println(WiFi.localIP());

}

// This functions is executed when some device publishes a message to a topic that your ESP8266 is subscribed to

// Change the function below to add logic to your program, so when a device publishes a message to a topic that

// your ESP8266 is subscribed you can actually do something

void callback(String topic, byte\* message, unsigned int length) {

Serial.print("Message arrived on topic: ");

Serial.print(topic);

Serial.print(". Message: ");

String messageTemp;

for (int i = 0; i < length; i++) {

Serial.print((char)message[i]);

messageTemp += (char)message[i];

}

Serial.println();

}

// This functions reconnects your ESP8266 to your MQTT broker

// Change the function below if you want to subscribe to more topics with your ESP8266

void reconnect() {

// Loop until we're reconnected

while (!client.connected()) {

Serial.print("Attempting MQTT connection...");

if (client.connect("ESP8266Client", "usersnir2", "projetmontgolfiere")) {

Serial.println("connected");

// Subscribe or resubscribe to a topic

// You can subscribe to more topics (to control more LEDs in this example)

} else {

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

// Wait 5 seconds before retrying

delay(5000);

}

}

}

// The setup function sets your ESP GPIOs to Outputs, starts the serial communication at a baud rate of 115200

// Sets your mqtt broker and sets the callback function

// The callback function is what receives messages and actually controls the LEDs

void setup() {

dht.begin();

Serial.begin(115200);

setup\_wifi();

client.setServer(mqtt\_server, 1883);

client.setCallback(callback);

}

// For this project, you don't need to change anything in the loop function. Basically it ensures that you ESP is connected to your broker

void loop() {

if (!client.connected()) {

reconnect();

}

if (!client.loop())

client.connect("ESP8266Client");

now = millis();

// Publishes new temperature and humidity every 30 seconds

if (now - lastMeasure > 30000) {

lastMeasure = now;

// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)

float h = dht.readHumidity();

// Read temperature as Celsius (the default)

float t = dht.readTemperature();

// Check if any reads failed and exit early (to try again).

if (isnan(h) || isnan(t) ) {

Serial.println("Failed to read from DHT sensor!");

return;

}

// Computes temperature values in Celsius

float hic = dht.computeHeatIndex(t, h, false);

static char temperatureTemp[7];

dtostrf(hic, 6, 2, temperatureTemp);

// Uncomment to compute temperature values in Fahrenheit

// float hif = dht.computeHeatIndex(f, h);

// static char temperatureTemp[7];

// dtostrf(hif, 6, 2, temperatureTemp);

static char humidityTemp[7];

dtostrf(h, 6, 2, humidityTemp);

// Publishes Temperature and Humidity values

client.publish("room/temperature", temperatureTemp);

client.publish("room/humidity", humidityTemp);

Serial.print("Humidite: ");

Serial.print(h);

Serial.print(" %\t Temperature: ");

Serial.print(t);

Serial.print(" \*C ");

}

}