Sensoren en interfacing – Evaluatie week 5

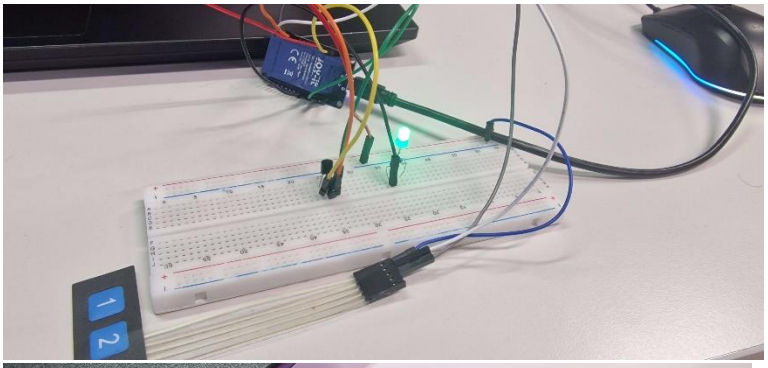
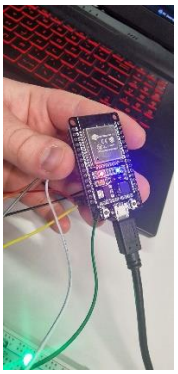
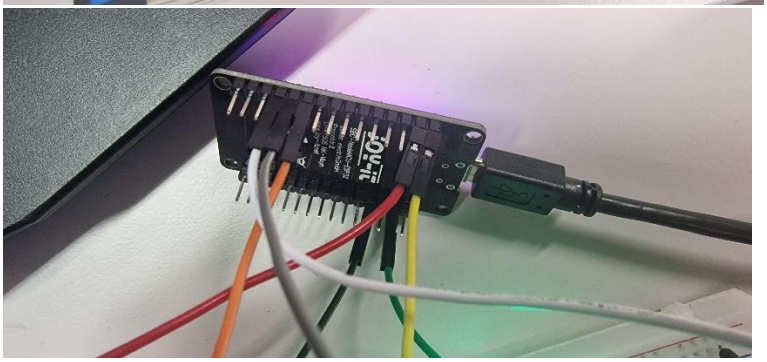
Opdracht: ontwerp (op basis van een microcontroller naar keuze en sensoren naar keuze) een IoT thermostaat voor een centraal verwarmingssysteem:

1. Temperatuur meting (0-30 graden op basis van een digitale sensor naar keuze).
2. Gewenste temperatuur instelling (0-30 graden met knopjes of potentiometer)
3. Lokale weergave van de gemeten en gewenste temperatuur en de toestand van de verwarmingsketel (aan/uit) naar keuze via serial monitor, LED display, LCD, OLED,… Ketel zelf wordt weergegeven door een LED.
4. Via de Blynk applicatie volledige bediening vanaf je telefoon (weergave van actuele temperatuur, ingestelde temperatuur (met instelknop) en status van de verwarmingsketel.

Al je antwoorden komen op dit document. **Je plakt telkens onder de vraag je antwoord**, mag via knipprogramma een schemaatje zijn of foto van je schets op papier.

Als je klaar bent **sla je dit document op als .pdf** en laad je het (tijdig) op in de uploadzone.

1. Teken het schema van je temperatuursensor en hoe hij verbonden is met je controller. (mag eventueel samen met vraag 2)

(3 punten)

2. Maak het **basis programma** dat lokaal de functie van thermostaat uitoefent. Simuleer je verwarmingsketel met een LED (aan/uit). Knip en plak het programma hieronder. Instellen van de temperatuur mag met drukknopjes of potentiometer naar keuze.
   (5 punten)
3. Maak een foto van je schakeling en plak deze hieronder
   (3 punten)
4. Breid je schakeling uit met je afstandsbediening via telefoon. Zorg ervoor dat je
   - De huiskamertemperatuur kan zien op je telefoon
   - De status van de verwarmingsketel kan zien op je telefoon
   - De gewenste temperatuur kan instellen

5. Publiceer alle files op je github pagina in een nieuwe repository “IoT thermostaat”.
   (4 punten)
6. Knip en plak het **volledige programma** hieronder.
   (5 punten)

```
#define BLYNK_TEMPLATE_ID "user12"
#define BLYNK_TEMPLATE_NAME "user12@server.wyns.it"
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = "4ZxxWy2LjJtcwxGcdLOgwiAGvxkyYAwW";
char ssid[] = "embed";
char pass[] = "weareincontrol";

int lm35Pin = 32;      // Analog pin connected to LM35 sensor
int lm35Value;         // Variable to store LM35 sensor value
int gezetteWaarde = 0; // Initialize the gezetteWaarde

const int increaseButtonPin = 34; // Pin for the increase button
const int decreaseButtonPin = 35; // Pin for the decrease button

void setup() {
  pinMode(lm35Pin, INPUT);      // Set LM35 pin as input
  pinMode(LED_BUILTIN, OUTPUT); // Set built-in LED pin as output
  pinMode(increaseButtonPin, INPUT_PULLUP); // Set increase button pin
  as input with internal pull-up resistor
```

```
30.   pinMode(decreaseButtonPin, INPUT_PULLUP); // Set decrease button pin
    as input with internal pull-up resistor
31.
32.   Serial.begin(115200);
33.   delay(10);
34.   Serial.print("Connecting to ");
35.   Serial.println(ssid);
36.
37.   WiFi.begin(ssid, pass);
38.   while (WiFi.status() != WL_CONNECTED) {
39.     delay(500);
40.     Serial.print(".");
41.   }
42.
43.   Serial.println("WiFi connected");
44.
45.   Blynk.begin(auth, ssid, pass, "server.wyns.it", 8081);
46. }
47.
48. void loop() {
49.   Blynk.run();
50.
51.   // Read LM35 sensor value
52.   lm35Value = analogRead(lm35Pin);
53.
54.   // Send LM35 value to Blynk app
55.   Blynk.virtualWrite(V1, lm35Value);
56.
57.   Blynk.virtualWrite(V3, gezetteWaarde);
58.
59.   // Print LM35 value to Serial monitor
60.   Serial.print("LM35: ");
61.   Serial.println(lm35Value);
62.
63.   // Check if LM35 value is above 27 degrees
64.   if (lm35Value > 27) {
65.     digitalWrite(LED_BUILTIN, HIGH); // Turn on LED
66.   } else {
67.     digitalWrite(LED_BUILTIN, LOW); // Turn off LED
68.   }
69.
70.   // Check if increase button is pressed
71.   if (digitalRead(increaseButtonPin) == 1) {
72.     gezetteWaarde++;
73.     delay(250); // Add a small delay to debounce the button
74.   }
75.
76.   // Check if decrease button is pressed
```

```
77.   if (digitalRead(decreaseButtonPin) == 1) {
78.     gezetteWaarde--;
79.     delay(250); // Add a small delay to debounce the button
80.   }
81.
82.   delay(200);  // Adjust delay as needed
83. }
```

84. Toon de werking aan de docent.
    Maak eventueel later met de webcam van je PC (of een andere telefoon) een filmpje waarin je zelf de werking van je IoT thermostaat uitlegt (aan iemand die geen specialist is in de materie) met demo waarop duidelijk de werking ervan te zien is en plak de link ernaar hieronder. Zet het filmpje bij voorkeur op je GitHub pagina. Noteer hieronder ook ***de link naar je Github project****. Als om de een of de andere reden iets in je programma niet naar behoren werkt leg je dit in dit filmpje uit waar je tegenaan gelopen bent.*
    (10 punten)

Zoals altijd is dit een individuele opdracht. Overleg over oplossingsmethodes kan en mag zolang je maar een **eigen originele realisatie** oplevert, geen kopieerwerk toegelaten, niet naar hardware noch software. Deadline voor indienen van je .pdf document is 17:45.

Succes !