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)

De 2 schema’s zijn hetzelfde.

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)

7. Toon de werking aan de docent.

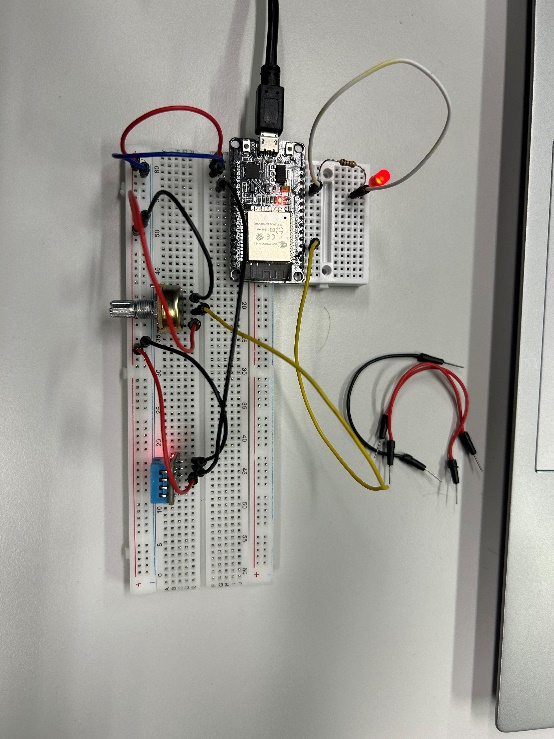
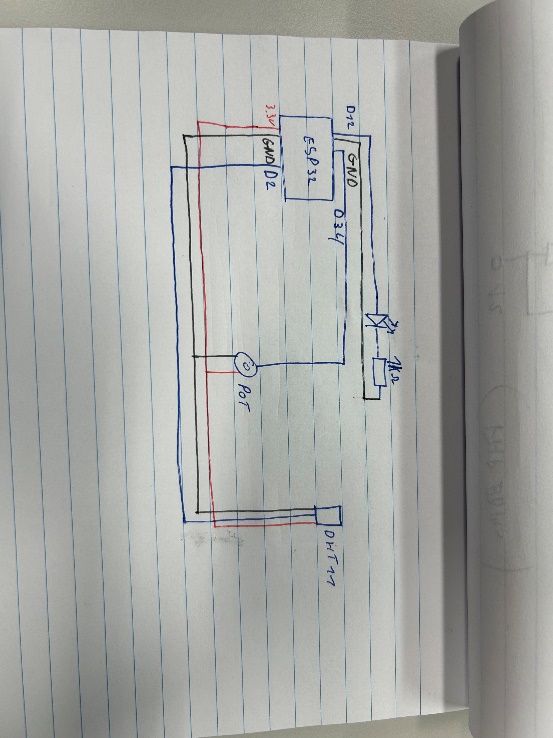
Maak eventueel later met de webcam van je PC (of een andere telefoon) een filmpje waarin je zelf de werking van je IoT thermostaat uitlegt (aan iemand die geen specialist is in de materie) met demo waarop duidelijk de werking ervan te zien is en plak de link ernaar hieronder. Zet het filmpje bij voorkeur 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 !

Schema is hetzelfde voor de 2 delen.

Opdracht zonder blynk:



Met blink: Afbeelding met elektronica, kabel, tekst, Elektrische bedrading

Automatisch gegenereerde beschrijving

Volledige code:

#define BLYNK\_TEMPLATE\_ID   "user9"

#define BLYNK\_TEMPLATE\_NAME "user9@server.wyns.it"

#define BLYNK\_PRINT Serial

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <DHT.h>

// Blynk instellingen

char auth[] = "8Wle5tD5B\_\_VxYCnc7kVadJXSRUZWZEm";

char ssid[] = "embed";

char pass[] = "weareincontrol";

char server[] = "server.wyns.it";

int port = 8081; // Poortnummer voor de Blynk-server

// DHT11 instellingen

#define DHTPIN 2       // Pin waarop de DHT11 is aangesloten

#define DHTTYPE DHT11  // Type DHT-sensor

DHT dht(DHTPIN, DHTTYPE);

// Potentiometer instellingen

#define POT\_PIN 34     // Pin waarop de potentiometer is aangesloten

#define POT\_MAX\_VALUE 4095.0  // Maximale waarde van de potentiometer (12-bit resolutie)

#define TEMPERATURE\_RANGE 30.0  // Gewenste temperatuur range

#define POT\_TOLERANCE\_PERCENT 3.0 // Tolerantie van de potentiometer in procenten

// Virtuele pinnen voor Blynk

#define V1 1 // Virtuele pin voor gemeten temperatuur

#define V2 2 // Virtuele pin voor gewenste temperatuur

#define V3 3 // Virtuele pin voor slider (gewenste temperatuur instellen)

#define V4 4 // Virtuele pin voor LED-waarde

#define LAMP\_PIN 12 // Pin voor de lamp

float gemetenTemperatuur = 0.0;

float gewensteTemperatuur = 0.0; // Start bij 0 graden

float laatstAangepasteWaarde = gewensteTemperatuur; // Nieuwe variabele toevoegen

int laatstGelezenPotValue = 0; // Variabele om de laatst gelezen potentiometerwaarde op te slaan

int LedValueOn;

void setup() {

  Serial.begin(115200);

  Blynk.begin(auth, ssid, pass, server, port);

  dht.begin();

  pinMode(POT\_PIN, INPUT);

  pinMode(LAMP\_PIN, OUTPUT);

  // Configuratie van virtuele pinnen

  Blynk.virtualWrite(V1, gemetenTemperatuur);

  Blynk.virtualWrite(V2, gewensteTemperatuur);

  // Blynk Slider-widget

  Blynk.virtualWrite(V3, gewensteTemperatuur);

  Blynk.virtualWrite(V3, 0, TEMPERATURE\_RANGE); // Stel de minimale en maximale waarde van de slider in

}

void loop() {

  Blynk.run();

  meetTemperatuur();

}

void meetTemperatuur() {

  gemetenTemperatuur = dht.readTemperature();

  int potValue = analogRead(POT\_PIN);

  // Lees potentiometer bij elke verandering

  if (abs(potValue - laatstGelezenPotValue) > (POT\_MAX\_VALUE \* POT\_TOLERANCE\_PERCENT / 100.0)) {

    float potGewensteTemperatuur = map(potValue, 0, POT\_MAX\_VALUE, 0, TEMPERATURE\_RANGE);

    gewensteTemperatuur = max(potGewensteTemperatuur, laatstAangepasteWaarde);

    laatstGelezenPotValue = potValue; // Update de laatst gelezen waarde

  }

  Serial.print("Gemeten temperatuur: ");

  Serial.print(gemetenTemperatuur);

  Serial.print(" °C, Gewenste temperatuur: ");

  Serial.print(gewensteTemperatuur);

  Serial.println(" °C");

  Blynk.virtualWrite(V1, gemetenTemperatuur);

  Blynk.virtualWrite(V2, gewensteTemperatuur);

  // Zet de lamp aan als de gewenste temperatuur hoger is dan de gemeten temperatuur

  if (gewensteTemperatuur > gemetenTemperatuur) {

    LedValueOn = 1023;

    digitalWrite(LAMP\_PIN, HIGH);

  } else {

    LedValueOn = 0;

    digitalWrite(LAMP\_PIN, LOW);

  }

  Blynk.virtualWrite(V4, LedValueOn);

  delay(1000);

}

BLYNK\_WRITE(V3) {

  gewensteTemperatuur = param.asFloat();

  Serial.print("Nieuwe gewenste temperatuur: ");

  Serial.println(gewensteTemperatuur);

  // Update de variabele met de laatst aangepaste waarde

  laatstAangepasteWaarde = gewensteTemperatuur;

}