



INSTRUMENT VIRTUAL LABVIEW PENTRU AFIŞARE ŞI TRIMITIRE DATE DE LA SENZORI



Viorel-Mihai Diaconu

Tabel de figuri:

Figure 1: ESP32	3
Figure 2: DHT11	4
Figure 3: MQ-135	4
Figure 4: Sensibilizate MQ-135.....	5
Figure 5: Diagramă de conectare senzori.....	6
Figure 6: Panou frontal	9
Figure 7: Buclă citire serial.....	9
Figure 8: Buclă prelucrare si afişare date	10
Figure 9: Buclă trimitere către Spreadsheet.....	10
Figure 10: Structură citire date de la Spreadsheet	11
Figure 11: Structură afişare date de la spreadsheet.....	12

Cuprins:

1. Introducere	3
2. Componente utilizate	3
a. ESP32	3
b. DHT11	4
c. MQ-135.....	4
Calibrare sensor:	5
3. Sistem de masurare	6
a. Diagrama de conectare senzori	6
b. Cod si librarii folosite:	6
4. Spreadsheet si appscript.....	8
5. Labview	9
a. Citirea de la portul serial	9
b. Prelucrarea și afișarea datele	10
c. Transmiterea către Google Spreadsheet	10
d. Citirea din Spreadsheet și afișarea pe intervale selectate	11
6. Bibliografie	13

1. Introducere

Calitatea aerului a devenit o preocupare globală majoră, Organizația Mondială a Sănătății estimând că peste 6,7 milioane de decese premature au fost cauzate de poluarea aerului în 2019 [1]. În acest context, monitorizarea parametrilor de mediu devine esențială atât în mediul urban, cât și în interiorul locuințelor. Proiectul de față propune realizarea unui instrument virtual în LabVIEW, conectat la un microcontroler ESP32, care colectează date de la senzori de temperatură, umiditate și calitate a aerului (DHT11 și MQ-135), afișează valorile în timp real și le salvează automat într-un Google Spreadsheet pentru analiză ulterioară.

Acest sistem combină capacitățile hardware accesibile ale ESP32 cu flexibilitatea grafică oferită de LabVIEW, oferind o soluție eficientă pentru prototipare rapidă sau monitorizare ambientală locală. Integrarea cu Google Sheets permite și extinderea funcționalității către platforme de analiză în cloud, fără a fi necesare infrastructuri complexe sau costisitoare.

2. Componente utilizate

a. ESP32

ESP32 este un microcontroler de tip low-cost, disponibil începând de la aproximativ 28-30 lei, ceea ce îl face accesibil pentru publicul larg. Este un dispozitiv cu consum redus de energie, fiind ideal pentru aplicații IoT care necesită autonomie ridicată, datorită modurilor de economisire a energiei precum deep sleep. Integrează capacități Wi-Fi, permițând conectarea la o rețea existentă sau crearea unei rețele proprii, facilitând astfel comunicarea între dispozitive. Majoritatea plăcilor ESP32 sunt echipate cu un procesor dual-core (Xtensa LX6 pe 32 de biți), oferind performanțe ridicate. Dispune de o interfață bogată de intrare/ieșire, ce include suport pentru atingere capacitivă, convertoare ADC și DAC, precum și protocoale de comunicație precum UART, SPI, I²C și PWM. ESP32 este compatibil atât cu limbajul de programare Arduino, cât și cu MicroPython, oferind astfel flexibilitate pentru dezvoltatori, fie că sunt începători sau avansați. [2]



Figure 1: ESP32

b. DHT11

Senzorul de temperatură și umiditate DHT11 are un complex de senzori de temperatură și umiditate cu o ieșire de semnal digital calibrată. Prin utilizarea tehnicii exclusive de achiziție a semnalului digital și a tehnologiei de detectare a temperaturii și umidității, asigură o fiabilitate ridicată și o stabilitate excelentă pe termen lung. Acest senzor include o componentă de măsurare a umidității de tip rezistiv și o componentă de măsurare a temperaturii NTC și se conectează la un microcontroler de înaltă performanță pe 8 biți, oferind o calitate excelentă, răspuns rapid, capacitate anti-interferență și rentabilitate.[3]

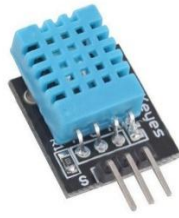


Figure 2: DHT11

c. MQ-135

Senzorul MQ-135 este un senzor de gaze cu o gamă largă de detecție, fiind sensibil la amoniac (NH_3), oxid de azot (NO_x), alcool, benzen, fum, dioxid de carbon (CO_2) și alți compuși care afectează calitatea aerului. Principiul său de funcționare se bazează pe un material sensibil din dioxid de staniu (SnO_2), al cărui nivel de rezistență scade în prezența gazelor țintă, generând astfel un semnal electric ce poate fi măsurat.[4]



Figure 3: MQ-135

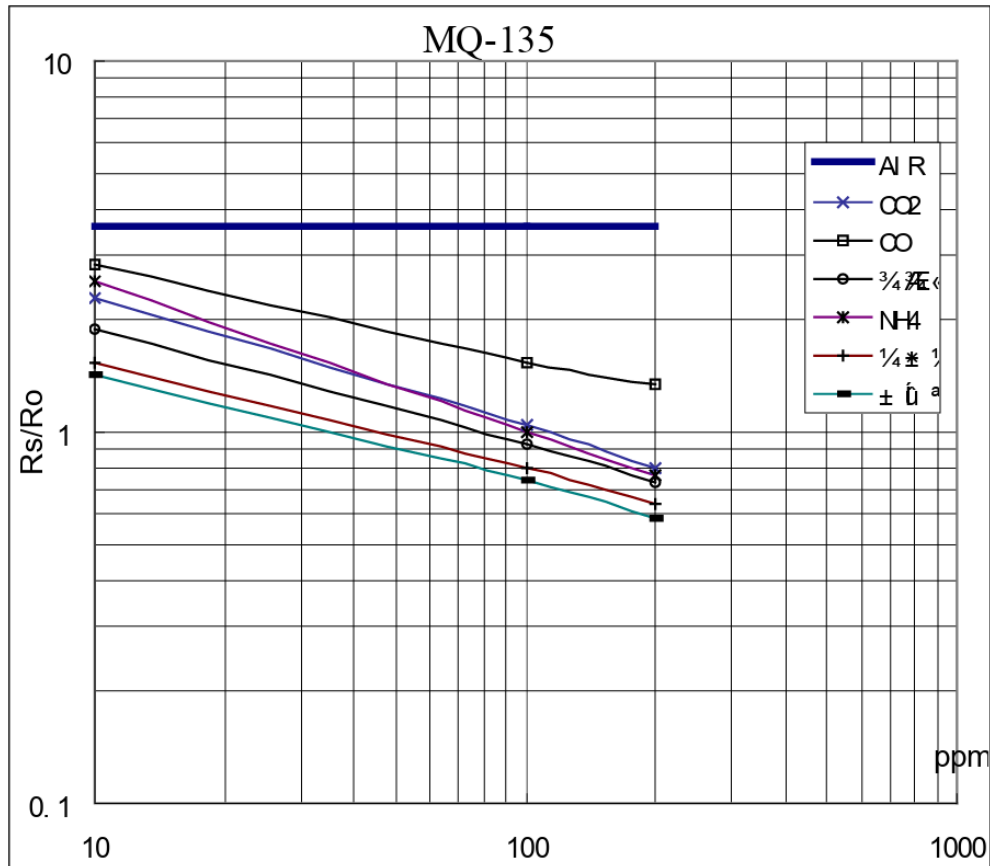


Figure 4: Sensibilitate MQ-135

Calibrare sensor:

Pentru a putea calibra sensorul avem nevoie de valorile R_s si R_0 . Pentru a putea calcula valoarea R_s trebuie știută valoarea rezistenței de sarcină R_l a cărei valoare măsurată este 9.89Kohmi și valoarea tensiunii V_{out} . Aceste valori se introduc în relația de mai jos.

$$R_s = \frac{(V_{in} - V_{out}) * R_l}{V_{out}}$$

unde:

- V_{in} este tensiunea de alimentare a senzorului (3.3 V),
- V_{out} este tensiunea măsurată la ieșirea senzorului,
- R_l este rezistența de sarcină (9.89 k Ω).

După determinarea valorii R_s , se poate calcula valoarea de referință R_0 , necesară pentru estimarea concentrației de gaze. Calibrarea se face într-un mediu considerat curat folosind următoarea formulă empirică:

$$R_0 = \frac{R_s}{3.6}$$

3. Sistem de masurare

a. Diagrama de conectare senzori

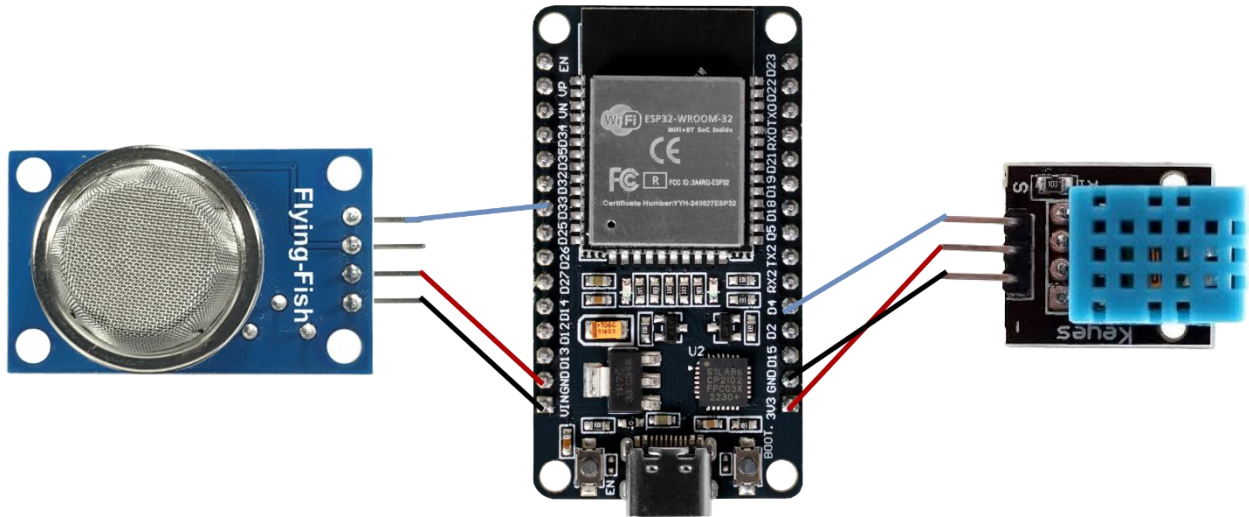


Figure 5: Diagramă de conectare senzori

Pinul de date pentru DHT11 este D4 si pinul pentru MQ-135 D33.

b. Cod si librarii folosite:

```
#include <DHT.h>
#define PPM_PIN 33    // Pin connected to MQ-135
#define DHTPIN 4      // Pin connected to DHT11
#define DHTTYPE DHT11 // DHT 11 sensor
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  analogReadResolution(12);
  dht.begin();
}

void loop() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();
  float RL = 9.89; // 9.89 kohm
  float R0 = 4.98;
  int analogValue = analogRead(PPM_PIN);
  float Vout = analogValue * (3.3 / 4095.0);
  float Rs = ((3.3 - Vout) * RL) / Vout;
  // float R0 = Rs / 3.6
  float ratio = Rs / R0;
  //https://davidegironi.blogspot.com/2014/01/cheap-co2-meter-using-mq135-sensor-with.html
  float ppm = 116.6020682 * pow(ratio, -2.769034857);
```

```

if (isnan(temperature) || isnan(humidity)) {
  Serial.println("Eroare citire DHT11!");
} else {
  Serial.print(ppm);
  Serial.print(",");
  Serial.print(temperature);
  Serial.print(",");
  Serial.println(humidity);
}
delay(500);
}

```

Senzorul MQ-135 oferă un semnal analogic proporțional cu concentrația de gaze poluante. Este conectat la unul dintre pinii cu convertor analog-digital (ADC) ai ESP32, iar valoarea este apoi citită cu funcția `analogRead()`. Astfel, tensiunea de ieșire de pe pinul senzorului (în intervalul 0–3.3 V) este convertită într-o valoare digitală pe 12 biți (0–4095). Această valoare este proporțională cu concentrația de gaz detectată. Pentru determinarea PPM este folosită următoarea relație de calcul. [5]

$$PPM = 116.6020682 * \frac{R_S^{-2.769034857}}{R_0}$$

Senzorul DHT11 furnizează date digitale despre temperatură și umiditate printr-un protocol specific pe un singur fir. Pentru a comunica cu senzorul, programul ESP32 utilizează biblioteca `DHT.h`, care gestionează automat trimiterea semnalului de inițializare, așteptarea răspunsului și decodarea pachetului de 40 biți trimis de senzor.

După citire, valorile de temperatură (în °C), umiditate relativă (în %) și PPM sunt transmise serial delimitate prin virgulă și preluate în Labview.

4. Spreadsheet si appscript

După ce datele sunt preluate de la senzori prin interfața serială și procesate în LabVIEW, acestea sunt trimise automat către un Google Spreadsheet, unde sunt stocate pentru analiză ulterioară. Acest lucru permite păstrarea unui istoric al valorilor măsurate (temperatură, umiditate, ppm), precum și vizualizarea și analiza acestora într-un mod ușor accesibil din cloud. Pentru a permite scrierea automată a datelor într-un spreadsheet, este necesară configurarea unui Apps Script adică un mic program JavaScript atașat documentului Google Sheets, care definește cum și unde se inserează datele.

Procesul include următoarele etape:

- Crearea unui fișier Google Sheets nou;
- Deschiderea editorului Apps Script (Extensions > Apps Script);
- Adăugarea unui cod care definește o funcție doPost(e) pentru a transmite datele prin metoda HTTP POST și o funcție doGet(e) pentru a prelua datele sub formă JSON și a le afișa pe un grafic.
- Publicarea scriptului ca web app (Deploy > Manage deployments), cu permisiunea „Anyone with the link”.

```
function doGet(e) {
  return handleRequest(e);
}
function doPost(e) {
  return handleRequest(e);
}
function handleRequest(e) {
  var sheet = SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();

  if (e.parameter.action == "read") {
    var data = sheet.getDataRange().getValues();
    return ContentService
      .createTextOutput(JSON.stringify(data))
      .setMimeType(ContentService.MimeType.JSON);
  } else {
    var ppm = parseFloat(e.parameter.ppm);
    var temp = parseFloat(e.parameter.temp);
    var hum = parseFloat(e.parameter.hum);
    sheet.appendRow([new Date(), ppm, temp, hum]);
    return ContentService.createTextOutput("Success");
  }
}
```

5. Labview

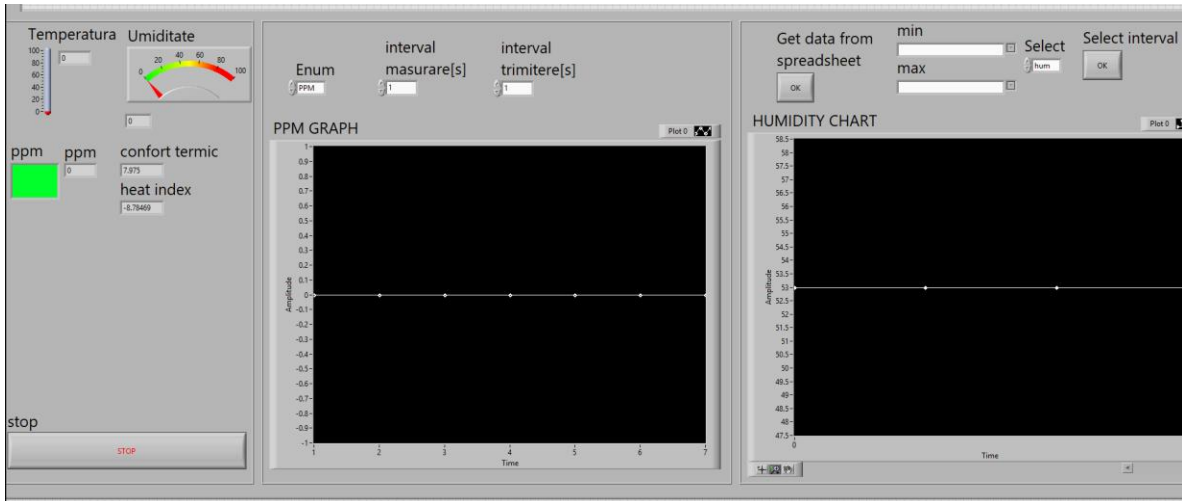


Figure 6: Panou frontal

Structura aplicației realizate în LabVIEW este împărțită în patru bucle while care rulează în paralel, fiecare având un rol bine definit în procesul de achiziție, prelucrare și stocare a datelor. Bucle comunică între ele prin queue-uri, asigurând un flux continuu și ordonat al informațiilor.

a. Citirea de la portul serial

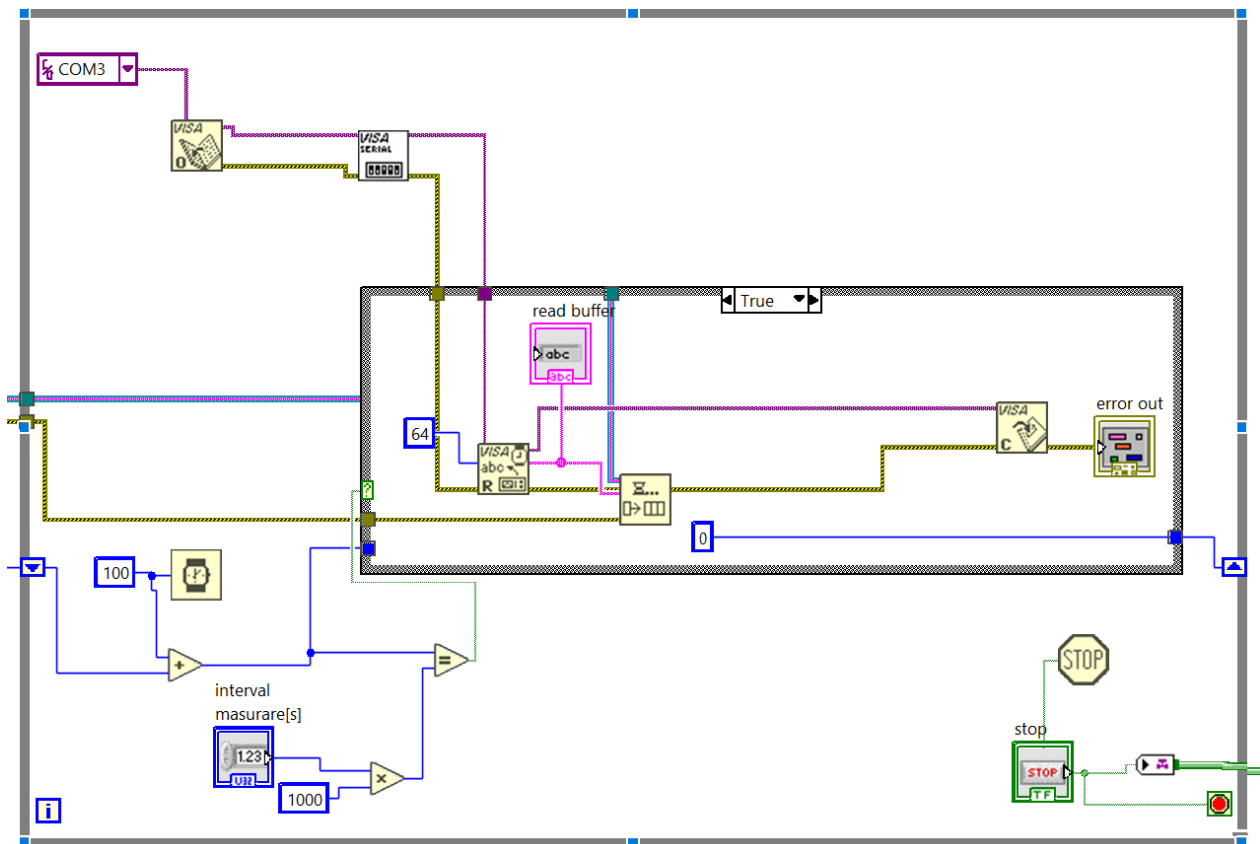


Figure 7: Buclă citire serial

b. Prelucrarea și afișarea datele

A treia buclă are rolul de a prelua datele de la al doilea queue și de a le trimite periodic către un Google Spreadsheet, folosind metoda HTTP POST. Acest lucru permite stocarea datelor într-un format accesibil și ușor de gestionat, precum și o eventuală analiză ulterioară în Google Sheets.

d. Citirea din Spreadsheet și afișarea pe intervale selectate

Ultima buclă este responsabilă cu citirea valorilor stocate în Spreadsheet. Utilizatorul poate selecta o perioadă de timp (dată de început și sfârșit), iar aplicația va afișa valorile corespunzătoare doar pentru acel interval, permițând astfel o vizualizare istorică și comparativă a datelor. Această buclă este împărțită în două case-structure.

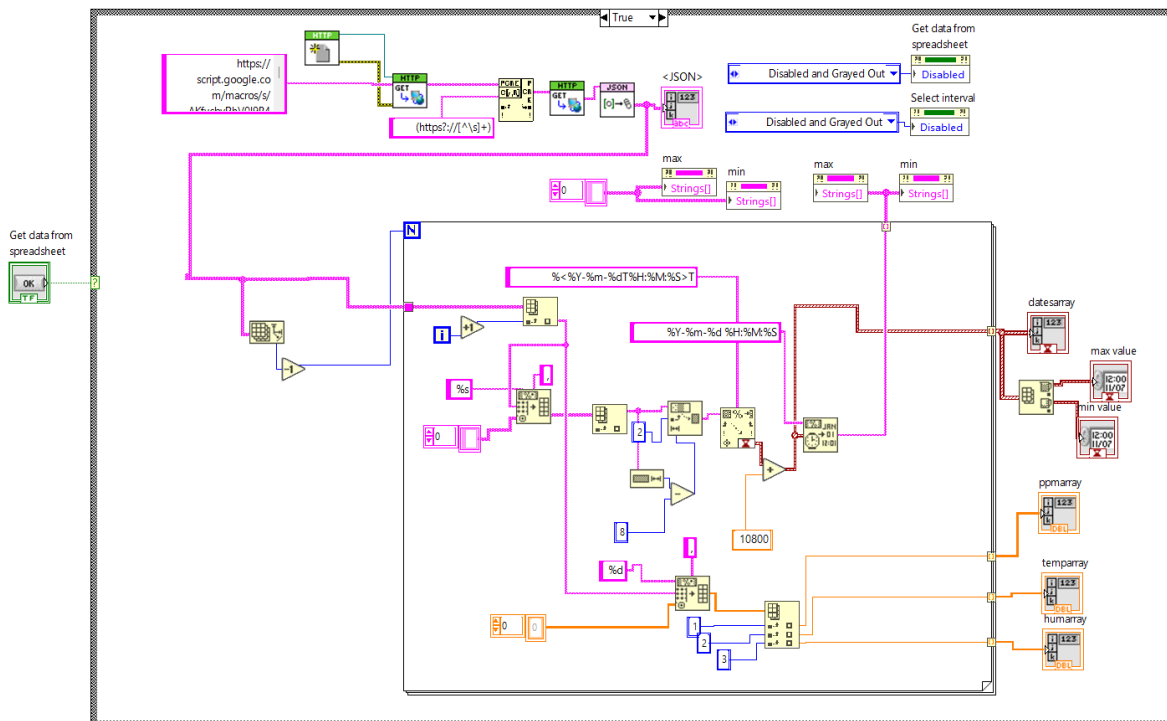


Figure 10: Structură citire date de la Spreadsheet

Primul case-structure se ocupă cu preluarea datelor de la spreadsheet și stocarea lor în array-uri pentru a le face disponibile pentru cel de-al doilea case-structure.

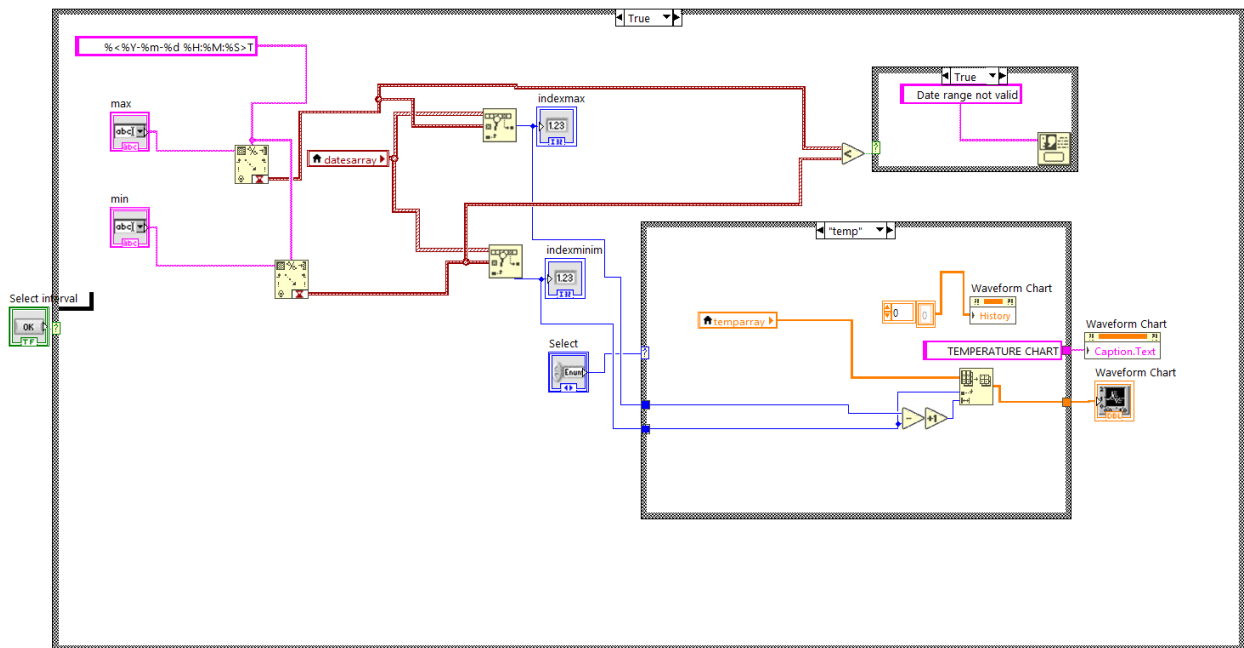


Figure 11: Structură afișare date de la spreadsheet

Cel de-al doilea case-strucure se ocupă cu afișarea datelor pe grafic. În momentul apăsării butonului de selectare a intervalului din array-urile unde au fost stocate datele, se preia un subarray ce conține doar valorile corespunzătoare din intervalul de timp selectat.

6. Bibliografie

- [1] World Health Organization, *Air pollution*, 2023. [https://www.who.int/news-room/fact-sheets/detail/ambient-\(outdoor\)-air-quality-and-health/](https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health/)
- [2] <https://randomnerdtutorials.com/getting-started-with-esp32/>
- [3] <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>
- [4] [https://www.winsensor.com/d/files/PDF/Semiconductor%20Gas%20Sensor/MQ135%20\(Ver1.4\)%20-%20Manual.pdf](https://www.winsensor.com/d/files/PDF/Semiconductor%20Gas%20Sensor/MQ135%20(Ver1.4)%20-%20Manual.pdf)
- [5] <https://davidegironi.blogspot.com/2014/01/cheap-co2-meter-using-mq135-sensor-with.html>