**PROIECT DE DIPLOMĂ**

**Candidat: Florin-Alexandru TĂNASE**

**Coordonator științific: s.l.dr.ing. Raul-Cristian ROMAN**

Sesiunea: Iunie 2025

# STAȚIE METEO DIGITALĂ CU MICROCONTROLER ESP32 ȘI SENZORI MULTIPLI INTEGRAȚI

**Partea tehnologică:** Utilizarea microcontrolerului ESP32 și a unor senzori digitali(BME280,LDR,DS3231,ploaie) pentru monitorizarea locală a condițiilor de mediu.

**Partea experimentală:** Realizarea unui montaj pe breadboard și testarea software-ului scris în Arduino IDE

**REZUMAT**

Lucrarea urmărește realizarea unei stații meteo digitale, capabilă să monitorizeze în timp real mai mulți parametri de mediu relevanți: temperatura, umiditatea aerului, presiunea atmosferică, nivelul de lumină și prezența ploii. Încă din cele mai vechi timpuri oamenii au fost interesați de fenomenele meteorologice care le afectau viața de zi cu zi. Plecând de la această idee am decis sa fac un dispozitiv prototip care să poată prezenta datele meteorologice in timp real.

Dispozitivul este construit în jurul microcontrolerului ESP32, ales pentru performanțele sale ridicate, consumul redus de energie și capacitatea de a integra eficient funcții de control și comunicație într-un sistem electronic compact. Sistemul integrează senzori specializați: BME280 pentru temperatură, umiditate și presiune; un foto-rezistor (LDR) pentru detectarea luminozității ambientale; un senzor de ploaie; și un modul de timp real (RTC DS3231), care asigură afișarea exactă a datei și orei. Toate datele sunt afișate local pe un ecran OLED, astfel creează accesibilitate crescută asupra condițiilor atmosferice. Existența modului de timp cu bateria CR2032 oferă posibilitatea continuarea evidenței chiar și fără alimentare. Montajul hardware a fost realizat pe breadboard, iar programarea sistemului a fost efectuată în Arduino IDE, folosind biblioteci dedicate pentru fiecare componentă. Pe parcursul lucrării, au fost testate și analizate toate funcțiile sistemului, simulând variații reale de temperatură, umiditate, lumină sau ploaie.

Din punct de vedere structural lucrarea debutează prin prezentarea elementelor hardware din dispozitiv, pe fiecare în parte atât din punct de vedere tehnic, cât și operațional, cu scheme și simulări individuale, iar mai apoi pe etapa finală cuprinzând dispozitivul complet și afișarea rezultatelor complexe. În cuprinsul lucrării sunt introduse atât date generale despre necesitatea măsurării datelor meteorologice, cât și evoluția în timp a dispozitivelor care efectuează aceste măsurători. În ceea ce privește partea software ea este prezentată ca un cod sursă modular, cu explicații locale aferente, majoritatea făcând parte din codurile de funcționare a senzorilor, cu adaptările punctuale date de obiectivul lucrării. Funcționarea stației meteorologice este asigurată prin integrarea hardware-ului și software-ului, cu un cod modular și structurat care gestionează achiziția, procesarea și afișarea în timp real a datelor.

**ABSTRACT**

This paper aims to develop a digital weather station capable of monitoring, in real time, several relevant environmental parameters: air temperature, humidity, atmospheric pressure, light level, and the presence of rain. Since ancient times, people have been concerned with meteorological phenomena that influenced their daily lives. Starting from this idea, I decided to build a prototype device that can display weather data in real time.

The device is built around the ESP32 microcontroller, chosen for its high performance, low power consumption, and ability to efficiently integrate control and communication functions in a compact electronic system. The system integrates specialized sensors: the BME280 for measuring temperature, humidity, and pressure; a photoresistor (LDR) for detecting ambient light; a rain sensor; and a real-time clock module (RTC DS3231), which ensures accurate display of date and time. All data is displayed locally on a 0.96” OLED screen, providing quick and easy access to current weather conditions. The presence of the RTC module powered by a CR2032 battery allows the system to keep track of time even in the absence of power supply. The hardware setup was assembled on a breadboard, while the system was programmed using the Arduino IDE, with dedicated libraries for each component. Throughout the project, all system functionalities were tested and analyzed by simulating real-world variations in temperature, humidity, light, and rainfall.

Structurally, the paper begins by presenting the hardware components of the device, each described both technically and functionally, along with individual schematics and simulations. The final stage covers the complete system and the display of complex results. The paper also includes general information about the importance of measuring meteorological data, as well as the evolution of weather monitoring devices over time. The software section is presented as a modular source code, with comments and local explanations, most of it based on sensor-specific operation codes, adapted to the objectives of the project. The weather station operates through the integration of hardware and software, using a structured, modular code that handles the acquisition, processing, and real-time display of data.

## CUPRINS

[STAȚIE METEO DIGITALĂ CU MICROCONTROLER ESP32 ȘI SENZORI MULTIPLI INTEGRAȚI 2](#_Toc201704797)

[CUPRINS 5](#_Toc201704798)

[1. STUDIUL DE LITERATURĂ 6](#_Toc201704799)

[1.1. OBIECTIVELE PROIECTULUI. PREZENTAREA GENERALĂ A PRODUSULUI. 6](#_Toc201704800)

[1.2. PROCESE DE OBȚINERE. JUSTIFICAREA ALEGERII PROCESULUI. 7](#_Toc201704801)

[1.3. PROPRIETĂȚILE PRODUSULUI ȘI METODE DE CARACTERIZARE 8](#_Toc201704802)

[I.4. UTILIZĂRI 10](#_Toc201704803)

[2. PROIECTARE TEHNOLOGICĂ 13](#_Toc201704804)

[2.1. ALEGEREA ȘI CARACTERIZAREA COMPONENTELOR 14](#_Toc201704805)

[2.1.1 Microcontroler ESP32-WROOM-32D 15](#_Toc201704806)

[2.1.2 Placa de test (breadboard),buton tactil și elemente auxiliare 19](#_Toc201704807)

[2.1.3 Ecran OLED 0.96” (I2C) 20](#_Toc201704808)

[2.1.3 Senzor BME280 – temperatură, umiditate și presiune 21](#_Toc201704809)

[2.1.4 Foto-rezistor (LDR) – senzor de luminozitate ambientală 23](#_Toc201704810)

[2.1.5 Senzor de ploaie capacitiv YL-83 24](#_Toc201704811)

[2.1.6 Modul umidificator cu ultrasunete 25](#_Toc201704812)

[2.1.7 Modul RTC DS3231 26](#_Toc201704813)

[2.2. FUNCȚIONAREA INDIVIDUALĂ A COMPONENTELOR 28](#_Toc201704814)

[2.2.1 Primul subsistem-ecranul OLED 0.96” 28](#_Toc201704815)

[2.2.2 Al doilea subsistem-senzorul „BME280” 32](#_Toc201704816)

[2.2.3 Al treilea subsistem-afișare a orei în timp real – „RTC DS3231” 37](#_Toc201704817)

[2.2.4 Al patrulea subsistem-senzorul de ploaie capacitiv YL-83 42](#_Toc201704818)

[2.3. DESCRIEREA MONTAJULUI FINAL A STAȚIEI METEO 46](#_Toc201704819)

[2.3.1 Montajul final pe breadboard și conectarea modulelor la ESP32 46](#_Toc201704820)

[2.3.1 Montajul final pe breadboard și conectarea modulelor la ESP32 50](#_Toc201704821)

[2.4. PROGRAMUL PRINCIPAL-FUNCȚIONALITATEA PROGRAMULUI 52](#_Toc201704822)

[2.5. AFIȘAREA PARAMETRILOR ÎNTR-O INTERFAȚĂ GRAFICĂ 58](#_Toc201704823)

[2.5.1Crearea interfeței grafice 58](#_Toc201704824)

[2.5.2 Afișarea parametrilor în interfață 60](#_Toc201704825)

[3. CONCLUZII ȘI IMPRESII 67](#_Toc201704826)

[3.1 PAȘII DE REALIZARE A PROIECTULUI 67](#_Toc201704827)

[3.2 AVANTAJE ȘI DEZAVANTAJE 68](#_Toc201704828)

[3.3 IMPRESII PERSONALE 68](#_Toc201704829)

[BIBLIOGRAFIE 69](#_Toc201704830)

# STUDIUL DE LITERATURĂ

Meteorologia este o ramură a științelor naturii care se ocupă cu studiul fenomenelor atmosferice, proceselor fizice și chimice din atmosferă și cu previziunea vremii. De-a lungul istoriei, studiul vremii a fost o preocupare centrală pentru oameni, fiind esențial pentru agricultură, transport, securitate și sănătate. Primele forme rudimentare de măsurare a fenomenelor atmosferice apar încă din antichitate, sub forma observațiilor empirice și a interpretărilor astrologice. În perioada Renașterii au fost introduse instrumentele clasice: barometrul (Torricelli, 1643), termometrul (Galilei, apoi Fahrenheit), higrometrul și anemometrul. Aceste dispozitive au pus bazele meteorologiei ca știință exactă.

Secolul XX a adus o dezvoltare accelerată a tehnicilor de măsurare și prognoză, odată cu apariția stațiilor meteo automate, sateliților meteorologici și a modelelor numerice de simulare climatică. În paralel, miniaturizarea componentelor electronice și apariția microcontrolerelor au permis dezvoltarea unor stații meteorologice digitale accesibile, configurabile și interconectate.

În prezent, dezvoltarea stațiilor meteo digitale integrate cu senzori de mediu și microcontrolere precum **ESP32** face posibilă monitorizarea în timp real a parametrilor meteorologici esențiali: temperatura, umiditatea, presiunea atmosferică, luminozitatea și nivelul de precipitații. În plus, utilizarea platformelor de simulare grafică (precum **Wokwi**) permite testarea prototipurilor virtuale înainte de implementarea fizică.

# OBIECTIVELE PROIECTULUI. PREZENTAREA GENERALĂ A PRODUSULUI.

Scopul lucrării este dezvoltarea unei **stații meteo digitale** capabile să colecteze și să afișeze în timp real mai mulți parametri de mediu relevanți: temperatura aerului, umiditatea relativă, presiunea atmosferică, luminozitatea ambientală și detectarea prezenței ploii. Totodată, sistemul include și un **modul de timp real (RTC)** pentru evidențierea corectă a datei și orei.

Proiectul are la bază **ESP32-WROOM-32D**, un microcontroler cu performanțe excelente și consum energetic redus, care oferă suficiente resurse pentru integrarea mai multor senzori simultan. Acesta este programat în **Arduino IDE**, iar proiectarea, testarea și validarea logicii de funcționare au fost realizate în mediul de simulare grafică **Wokwi**, care permite o verificare vizuală și funcțională a întregului sistem.

Stația integrează:

* senzor **BME280** (măsurarea temperaturii, umidității și presiunii atmosferice),
* un **foto-rezistor (LDR)** pentru luminozitate,
* un **senzor de ploaie capacitiv**,
* un modul **RTC DS3231** pentru timp real,
* și un **ecran OLED 0.96"** pentru afișaj local.
* modul **umidificator cu ultrasunete**

Un aspect important este existența unei **interfețe interactive de afișare**, în care utilizatorul poate vizualiza valorile actualizate în timp real direct pe ecranul OLED. Această interfață este gândită să fie intuitivă, lizibilă și să asigure acces rapid la datele colectate, contribuind astfel la utilitatea practică a sistemului.

Lucrarea urmărește nu doar realizarea fizică a unei stații meteo, ci și înțelegerea proceselor hardware și software care stau la baza unui sistem funcțional integrat, cu aplicații în educație, cercetare și medii casnice sau industriale.

# 1.2. PROCESE DE OBȚINERE. JUSTIFICAREA ALEGERII PROCESULUI.

La baza procesului de obținere a stat analiza temei alese. Am decis destul de repede că vreau să am atât o componentă hardware pe lângă cea software integrată. În faza de proiectare a stației meteo, am mers pe idea că elementele hardware trebuie alese în funcție de eficiența dovedită anterior. Selecția s-a bazat atât pe specificațiile tehnice ale componentelor disponibile comerciale. Am început cu alegerea microcontrolerului. După ce am analizat mai multe opțiuni – cum ar fi Arduino Uno, ESP8266 sau Raspberry Pi Pico – am decis să folosesc **ESP32-WROOM-32D**. Acesta oferă multiple avantaje: procesor dual-core, memorie RAM și flash generoasă, conectivitate wireless integrată și, cel mai important, un număr mare de pini digitali și analogici, ceea ce îl face ideal pentru conectarea mai multor senzori simultan. Senzorul principal care stă la baza măsurării principalilor parametrii ai stației meteo este **senzorul BME280**,fiind cel mai recomandat ca precizie și mai ales pentru faptul că întrunește mai mulți alți senzori cum ar fi (DHT22,BMP180).Această alegere reduce complexitatea montajului și minimizează consumul de pini.

Pentru detecția luminozității ambientale, am integrat un **foto-rezistor LDR**. Acesta este o soluție simplă și eficientă pentru identificarea variațiilor de lumină, fără a complica schema electrică. Este ușor de citit printr-un divizor de tensiune conectat la un pin analogic.

În ceea ce privește identificarea precipitațiilor, am ales un **senzor de ploaie capacitiv**, mai rezistent în timp decât versiunile bazate pe plăci cu urme metalice. Acesta permite detectarea picăturilor de apă sau a umezelii, semnalizând astfel prezența ploii. Un element important în configurarea proiectului este includerea unui **modul RTC DS3231**,dotat cu o baterie de tip CR2032 care oferă continuitatea funcționării chiar și atunci când nu există alimentare. Principalul său rol este contorizarea orei și a datei exacte. Pentru a facilita utilizarea sistemului și afișarea valorilor colectate am introdus un **ecran OLED 0.96”**.Acesta are un consum redus de energie și afișează clar caracterele, având și un **buton tactil** pentru schimbarea afișajului.

Pentru realizarea și testarea logicii de funcționare a stației meteo, am optat pentru două unelte software complementare: **Arduino IDE** și platforma de simulare online **Wokwi**. Alegerea acestor două medii a fost determinată de criterii precum accesibilitatea, compatibilitatea cu ESP32, suportul pentru biblioteci specifice și experiența practică pe care o oferă dezvoltatorului.

**Arduino IDE** este una dintre cele mai populare platforme de programare pentru microcontrolere precum ESP32. Are o interfață simplă, este gratuit și oferă suport extins pentru o gamă largă de plăci și senzori. De asemenea, dispune de o bibliotecă vastă de exemple și documentație, ceea ce a făcut mai ușoară integrarea componentelor precum BME280, OLED, RTC DS3231 sau senzorul de ploaie. Un avantaj major este posibilitatea încărcării directe a codului pe dispozitivul fizic și monitorizarea datelor prin serial monitor.

Pe de altă parte, în fazele incipiente ale proiectului, când unele componente fizice nu erau disponibile, am folosit **Wokwi**, o platformă online de simulare a circuitelor cu microcontrolere. Aceasta oferă suport pentru ESP32 și permite adăugarea virtuală a senzorilor și a altor module, folosind aceleași biblioteci și structuri de cod ca în Arduino IDE. Avantajele Wokwi sunt multiple: testarea rapidă a codului, vizualizarea conexiunilor, posibilitatea de debugging în timp real și eliminarea riscurilor legate de greșeli în montajul fizic.

Această combinație – dezvoltare în Arduino IDE și testare/simulare în Wokwi – a contribuit la reducerea semnificativă a timpului de depanare și a asigurat o tranziție lină între prototipul virtual și cel real. În plus, Wokwi oferă un mediu ideal pentru începători, fiind intuitiv și accesibil direct din browser, fără a necesita instalări suplimentare.

Procesul de selecție al componentelor a fost ghidat de dorința de a construi un sistem funcțional, dar și eficient din punct de vedere tehnic și logistic. Alegerea ESP32 a venit firesc, având în vedere experiența anterioară cu acest microcontroler, dar și nevoia unui dispozitiv care să poată susține mai mulți senzori și să rămână stabil în funcționare. Fiecare modul adăugat cum ar fi de BME280, senzorul de lumină sau afișajul OLED, a fost ales în urma unei analize a compatibilității, consumului de resurse și ușurinței de integrare.

Pe partea software, am combinat lucrul în Arduino IDE cu simulările din Wokwi, ceea ce mi-a permis să testez rapid ideile și să corectez eventuale probleme înainte de a trece la montajul fizic. Alegerea acestor platforme mi-a oferit un echilibru între controlul complet asupra codului și siguranța în testare. Privind în urmă, consider că deciziile luate reflectă nu doar cerințele tehnice ale proiectului, ci și experiențele personale acumulate pe parcursul facultății. Fiecare alegere a fost justificată prin utilitate, eficiență și dorința de a învăța aplicat.

# 1.3. PROPRIETĂȚILE PRODUSULUI ȘI METODE DE CARACTERIZARE

Stația meteo realizată în cadrul acestei lucrări urmărește monitorizarea în timp real a unor parametri meteorologici de bază: temperatura ambientală, umiditatea relativă a aerului, presiunea atmosferică, luminozitatea și prezența precipitațiilor. Aceste informații sunt esențiale pentru înțelegerea condițiilor climatice locale și pentru anticiparea variațiilor de mediu pe termen scurt.

Produsul este construit pe o arhitectură modulară, în centrul căreia se află microcontrolerul ESP32, recunoscut pentru consumul redus de energie, conectivitatea avansată și capacitatea de a gestiona simultan mai mulți senzori ***[4]***. Datele preluate de la senzori sunt procesate local și afișate în timp real pe un ecran OLED 0.96", oferind astfel o interfață clară și accesibilă pentru utilizator.

Caracteristicile fiecărui modul utilizat sunt prezentate în tabelul de mai jos, împreună cu metodele de preluare a valorilor. Senzorii comunicați prin interfață I2C sau analogică sunt interpretați de ESP32 și calibrați prin codul sursă. Precizia măsurătorilor este dată de specificațiile tehnice ale componentelor și este suficientă pentru aplicații educaționale sau de uz personal.

Tabelul 1. Caracteristicile tehnice ale componentelor utilizate în stația meteo

|  |  |  |  |
| --- | --- | --- | --- |
| **Componentă** | **Parametru** | **Interval/Precizie** | **Tip comunicare** |
| **BME280** | Temperatură | -40...+85 °C ±0.5 °C | I2C |
|  | Umiditate | 0...100 % RH ±3 % | I2C |
|  | Presiune | 300...1100 hPa ±1 hPa | I2C |
| **LDR (fotorezistor)** | Luminozitate | Rezistență variabilă (kΩ) | Analogic |
| **Senzor de ploaie** | Precipitații (umed/uscat) | Detecție prin conducție | Analogic/Digital |
| **RTC DS3231** | Ceas timp real | ±2 ppm (precizie ridicată) | I2C |
| **OLED 0.96"** | Afișare locală | 128x64 pixeli, monocrom | I2C |
| **ESP32-WROOM-32D** | Procesare & comunicare | 240 MHz, dual-core, Wi-Fi/BLE | - |
| **Umidificator cu ultrasunete (20 mm)** | Generare vapori (simulare ploaie) | Activare manuală, alimentare 5V | Independent (microUSB) |

Metodele de caracterizare implică achiziția periodică a datelor din fiecare senzor și afișarea lor sincronă pe ecran. Afișajul OLED este actualizat automat la fiecare ciclu de citire, iar valorile sunt reîmprospătate în timp real. RTC-ul permite menținerea orei exacte chiar și în lipsa alimentării, iar senzorii analogici sunt calibrați software pentru a oferi interpretări relevante (ex. valoare prag pentru ploaie sau întuneric).

Această abordare asigură o funcționare fiabilă și coerentă a întregului ansamblu, oferind o reprezentare clară a stării atmosferice locale. În plus, fiecare componentă a fost aleasă astfel încât să poată fi ușor înlocuită sau extinsă, ceea ce face din această stație meteo un prototip flexibil și scalabil.

# I.4. UTILIZĂRI

Stațiile meteo nu mai sunt demult rezervate doar instituțiilor de cercetare sau meteorologie. În prezent, datorită dezvoltării tehnologiei și accesibilității componentelor electronice, aceste sisteme pot fi construite și utilizate la scară mică în diverse domenii – de la locuințe private și agricultură, până la educație, smart city și proiecte de cercetare aplicată.

**Utilizare în medii rezidențiale și smart home:**

În mediul rezidențial, stația meteo aduce un plus major de confort și eficiență energetică. Informațiile colectate despre temperatură și umiditate pot fi folosite pentru a regla în mod automat sistemul de climatizare, reducând astfel consumul inutil de energie. În zilele foarte calde, sistemul poate trimite un semnal către un ventilator sau aer condiționat și să pornească doar atunci când este cu adevărat necesar. Senzorul de lumină poate adapta intensitatea becurilor LED din locuință, iar senzorul de ploaie poate fi integrat cu un sistem de acționare a geamurilor, pergolelor sau jaluzelelor exterioare. Astfel de automatizări contribuie la un mediu mai inteligent și mai eficient energetic.



**Figura 1 – Stație meteo digitală smart home***- Sursă:* [*www.bresser.de*](http://www.bresser.de) ***[1]***

**Utilizare în agricultură:**

În agricultură, meteorologia reprezintă factorul de bază. Prin măsurarea precisă a umidității aerului și a temperaturii, se pot preveni fenomene precum uscarea plantelor sau apariția bolilor fungice cauzate de umiditate excesivă. Senzorul de ploaie permite oprirea automată a sistemelor de irigare atunci când natura își face treaba, economisind apă și protejând plantele de exces. Presiunea atmosferică este, de asemenea, un indicator bun pentru anticiparea schimbărilor bruște ale vremii – o scădere rapidă putând semnala apropierea unei furtuni. În acest context, stația devine un instrument extrem de valoros pentru micii agricultori, grădinari sau pasionați de sere, care pot gestiona resursele mai eficient și ecologic.



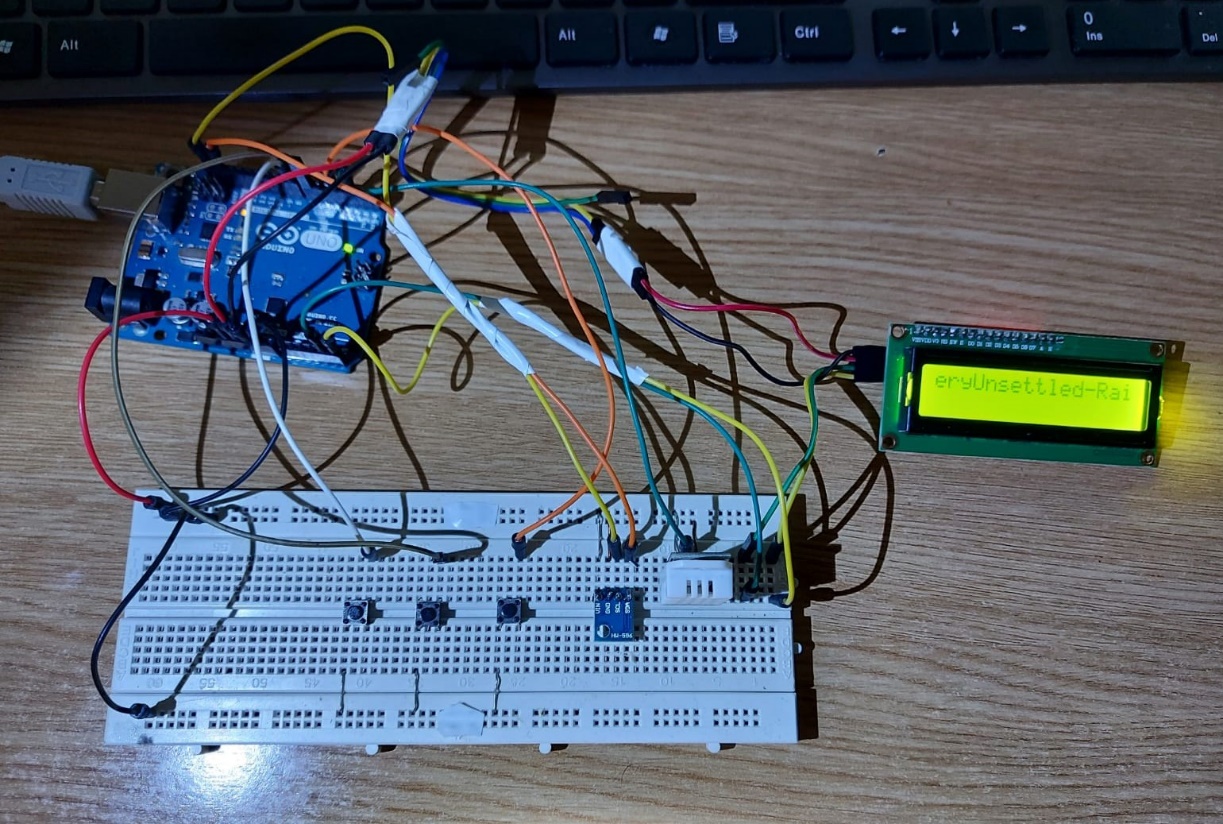
**Figura 2 – Stație meteo agricolă -***Sursă:* [*Agrobiznes.md*](https://agrobiznes.md/moldova-si-japonia-vor-realiza-un-proiect-comun-in-domeniul-meteorologiei.html) ***[2]***

**Utilizări comerciale și industriale:**

Dincolo de aplicațiile personale, sistemul poate fi adaptat și pentru utilizări comerciale sau industriale. În sere automatizate, depozite sensibile la temperatură/umiditate, silozuri, centre logistice sau ferme comerciale, datele meteo în timp real pot fi esențiale pentru controlul calității și prevenirea pierderilor. Sistemul bazat pe ESP32 permite integrarea cu platforme externe prin rețea Wi-Fi sau comunicații seriale, făcând posibilă transmiterea datelor către sisteme SCADA sau aplicații cloud.

**Utilizare educațională și dezvoltare embedded:**

Această stație reprezintă, totodată, o unealtă ideală pentru scopuri educaționale. Oferă un exemplu aplicat al modului în care diverse concepte teoretice – măsurare analogică și digitală, protocoale de comunicație (I2C), afișarea datelor, programarea modulară – pot fi transpuse într-un proiect concret și funcțional. Pentru elevi și studenți din domeniile automatică, electronică sau informatică, sistemul poate deveni o bază de plecare pentru experimente, simulări și dezvoltări ulterioare. Fiind ușor de montat și accesibil financiar, permite explorarea creativă a ideii de sistem embedded, fără a necesita infrastructură costisitoare.



**Figura 3 – Stație meteo educațională**- *sursă****[3]***

**Scalabilitate și integrare în IoT:**

Fiind bazată pe microcontrolerul ESP32, stația meteo este complet scalabilă. Pot fi adăugate cu ușurință module pentru trimiterea datelor către un server web sau pentru salvarea acestora pe un card SD. Astfel, proiectul capătă valențe majore în contextul Internet of Things (IoT), putând fi integrat în rețele mai mari de senzori distribuiți. De exemplu, mai multe stații meteo montate în diferite zone ale unei localități pot contribui la realizarea unei hărți locale de condiții atmosferice, utile pentru administrații locale, ferme mari sau aplicații de cercetare ***[18]***.

# PROIECTARE TEHNOLOGICĂ

Proiectarea Tehnologică reprezintă în mare desfășurarea activității proiectului de licență, care a fost structurat pe mai multe subcapitole. În baza incipientă, pe lângă identificarea temei, alegerea și procurarea componentelor și resurselor necesare au fost foarte importante. În continuare am stabilit ca este necesar sa expun procesul prin împărțirea funcționalității individuale a componentelor apoi pe asamblarea unitara si funcționalitatea completa a stație meteo. Partea de implementare a codului de funcționare este descris într-un capitol separat, acesta finalizându-se cu afișarea primara a unor valori colectate, pentru a prezenta aplicativitatea codului sursă. Un alt aspect tehnic este implementarea unei interfețe grafice simple, dar totodată utile pentru a facilita utilizarea stație meteo de către orice alte persoana. Celelalte capitole se refera la moduri in care se poate dezvolta aplicația. Începând cu subcapitolul 2.7, sunt prezentate elementele legate de automatizarea procesului tehnologic, în special declanșarea unor acțiuni automate pe baza valorilor senzorilor. Se continuă cu analiza limitărilor prototipului și sunt propuse direcții de dezvoltare viitoare a sistemului, precum integrarea în rețele IoT sau extinderea funcționalităților. La final, sunt tratate aspectele ecologice, respectiv normele de siguranță și securitate a muncii, necesare pentru completarea documentației tehnice a proiectului.

# ALEGEREA ȘI CARACTERIZAREA COMPONENTELOR

După ce am ales tema lucrării – realizarea unei stații meteo digitale – primul pas a fost să mă gândesc cum poate fi pusă în practică ideea, ce componente sunt necesare și ce cerințe tehnice trebuie îndeplinite. Am vrut ca sistemul să fie cât mai simplu de construit, dar și suficient de precis și stabil pentru a putea fi folosit real, nu doar demonstrativ.

Fiind vorba de un proiect care implică mai mulți senzori și afișaj, aveam nevoie de un **microcontroler** care să suporte toate aceste conexiuni, să poată citi datele rapid și să permită eventuale extinderi pe viitor (ex. transmiterea datelor prin Wi-Fi sau salvarea lor pe un card). Am luat în calcul variante precum **Arduino Uno**, **ESP8266** și **Raspberry Pi Pico**, dar în final am ales **ESP32-WROOM-32D**. Acesta oferă procesor dual-core, suficienți pini digitali și analogici, memorie mai mare decât Arduino și, cel mai important, are Wi-Fi și Bluetooth încorporat. A fost alegerea potrivită pentru un sistem integrat și flexibil ***[4]***.

După stabilirea controlerului principal, am trecut la alegerea senzorilor. Pentru a măsura **temperatura, umiditatea și presiunea**, am ales **BME280**, un senzor complet și precis, care comunică prin interfață I2C. Față de alternative ca DHT22 sau BMP180, BME280 este mai compact, mai stabil și oferă toate cele trei măsurători dintr-un singur modul, reducând astfel complexitatea conexiunilor ***[10]***.

Pentru **luminozitate ambientală**, am integrat un **foto-rezistor (LDR)**, deoarece este simplu de folosit și suficient pentru a detecta variații de lumină. Valoarea analogică returnată poate fi folosită pentru a reacționa în funcție de lumina zilei (ex. afișaj sau iluminare) ***[11]***.

Pentru **detectarea ploii**, am optat pentru un **senzor capacitiv**, mai fiabil decât cel clasic cu urme metalice, deoarece este mai rezistent la coroziune și uzură. Acesta poate detecta umezeala și semnaliza momentul în care încep precipitațiile ***[12]***.

În plus, pentru a putea testa în mod controlat reacția senzorului de ploaie fără a expune direct circuitul la apă, am adăugat un **modul umidificator cu ultrasunete**, care generează vapori de apă prin vibrații de înaltă frecvență. Acesta este folosit exclusiv pentru simularea artificială a ploii, fiind activat manual și alimentat separat, fără integrare directă în schema logică a stației meteo ***[13]***.

Am inclus și un **modul RTC DS3231**, pentru a avea afișare exactă a orei și datei, chiar și atunci când dispozitivul nu este alimentat. Este util mai ales pentru înregistrarea valorilor pe intervale de timp și are o precizie ridicată, datorită cristalului integrat ***[15]***.

În final, pentru afișarea locală a valorilor am folosit un **ecran OLED de 0.96”**, cu interfață I2C. Este compact, are un consum redus și oferă o vizibilitate bună, fiind potrivit pentru un prototip de dimensiuni mici ***[7]***.

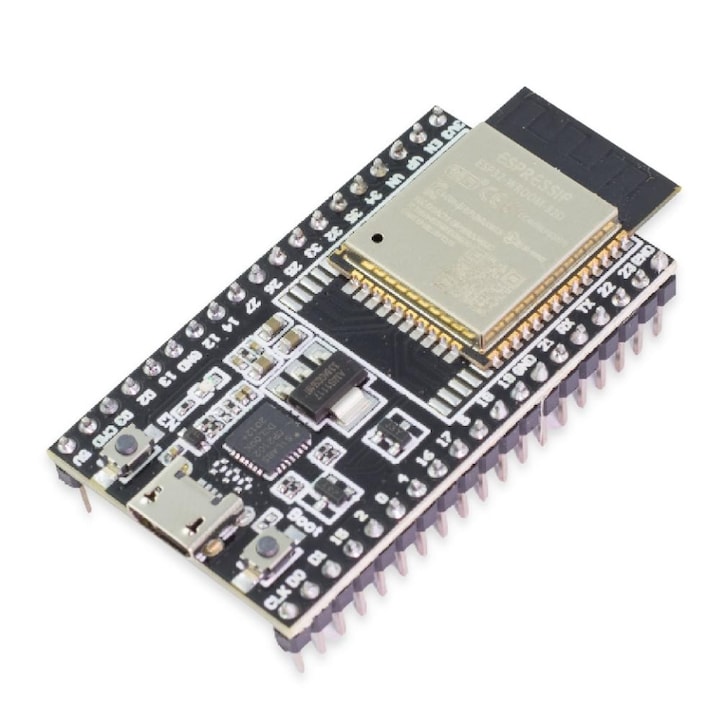
Toate aceste componente au fost alese nu doar pentru că sunt eficiente, ci și pentru că sunt bine documentate, ușor de integrat cu ESP32, compatibile cu Arduino IDE și disponibile comercial la un preț rezonabil. În plus, multe dintre ele pot fi simulate în **Wokwi**, ceea ce a ușurat partea de testare.

## 2.1.1 Microcontroler ESP32-WROOM-32D

La începutul proiectului, unul dintre cele mai importante lucruri la care m-am gândit a fost ce microcontroler să folosesc. Voiam ceva destul de puternic ca să poată citi date de la mai mulți senzori în același timp, dar și suficient de flexibil încât să permită extinderi mai târziu – poate o conexiune Wi-Fi sau stocarea datelor online.

Auzisem de **ESP32** din mai multe surse, atât de la colegi care l-au folosit în proiecte similare, cât și din videoclipuri și forumuri unde era lăudat pentru performanțele sale. Mi-a atras atenția pentru că venea cu Wi-Fi și Bluetooth integrate – ceea ce deschidea mai multe posibilități – și pentru că are suficienți pini digitali și analogici pentru tot ce îmi propusesem să conectez. Față de plăcile clasice Arduino, părea mai modern, mai rapid și mai „pregătit” pentru IoT.

După ce m-am documentat puțin, am decis să merg pe varianta **ESP32-WROOM-32D**, în format DevKitC. Am găsit placa disponibilă pe „**eMAG”**, unde a fost ușor de comandat, și am avut și avantajul că este compatibilă direct cu Arduino IDE, deci nu aveam nevoie să învăț un mediu nou de la zero.



**Figura 4.1 – Microcontrolerul ESP32-WROOM-32D –** *sursă:* [*emag.ro/esp32-wroom-32d*](https://www.emag.ro/placa-esp32-wifi-ble-4-2-compatibila-cu-devkitc-esp-wroom-32d/pd/D85798MBM) ***[4]***

Aspecte prezentate de site-ul „**eMAG”** care comercializează plăcuța de dezvoltare creată de **„Espressif**” împreună cu detalii tehnice ***(sursa: [4])***

„*Placa de dezvoltare ESP-WROOM-32D cu cip ESP32 de la Espressif este o platforma respectata si populara pentru dezvoltarea hardware-ului si a software-ului IoT (Internet of Things). Unul dintre cele mai mari avantaje ale cipurilor ESP32 este Wi-Fi-ul 2,4 GHz integrat in standardul B/G/N. Acesta poate funcționa atât ca un dispozitiv final conectat la o rețea existenta, cat si ca un punct de acces care creează o rețea Wi-Fi proprie. In plus fata de Wi-Fi, ESP32 este echipat cu un modul BLE v4.2 (Bluetooth Low Energy), ceea ce îl face ideal pentru aplicațiile IoT. Popularitatea continua sa crească pentru ESP32, in parte datorita dimensiunilor sale reduse. Acest modul deschide uși nelimitate pentru experimentarea in domeniul IoT. De la case inteligente la dispozitive conectate la cloud, posibilitățile sunt nelimitate. Programarea modulului ESP32 este ușoara datorita compatibilității cu Arduino IDE. Cu bibliotecile Arduino, programarea pentru IoT nu a fost niciodată mai simpla. Modulul vine echipat cu un convertor USB-UART cu cip CP2102 pentru programare, iar un buton suplimentar "Boot" il aduce in modul de programare. Programarea se realizează prin intermediul portului micro USB, care furnizează si 5V pentru alimentarea întregii placi.”*

Tabelul 2. Specificații Tehnice Cheie ESP32 ESP-WROOM-32D-*(sursa: [4])*

|  |  |
| --- | --- |
| **Parametru** | **Specificație** |
| Tip cip | ESP32 |
| Memorie RAM | 520 KB |
| Memorie Flash | 4 MB |
| Număr GPIO-uri | 32 |
| Interfețe UART | 3 |
| Interfețe SPI | 3 |
| Interfețe I2C | 2x I2C (2x I2S) |
| ADC (convertor analog-digital) | 12 canale |
| DAC (convertor digital-analog) | 2 canale |
| Canale PWM | Multiple (număr nelimitat prin software) |
| Slot card SD | Suport pentru conectare |
| Senzor Hall | Integrat |
| Suport interfață tactilă | Da |
| Port USB | microUSB pentru programare și alimentare |
| Convertor USB↔UART | CP2102 |
| Antena | PCB integrată |
| Raster pini (Goldpin) | 2,54 mm |
| Dimensiuni modul | 55 x 28 x 5 mm (fără conectori) |
| Pini de conectare | Conectori și pini de aur lipiți pe modul |

**Configurația pinilor și funcționalități avansate**

ESP32-WROOM-32D integrează un set bogat de pini digitali și analogici, împărțiți între funcții de intrare/ieșire (GPIO), comunicație serială (UART, SPI, I2C), convertori analog-digital (ADC), digital-analog (DAC), interfețe capacitive (touch sensing) și funcții speciale (PWM, SDIO, etc.). Dispunerea pinilor **(figura 4.2)** poate fi consultată și în documentația oficială ***[4]*** și în ghidurile de utilizare ***[5]****.*

Pinii sunt multiplexați, adică pot îndeplini mai multe funcții în funcție de cum sunt configurați în software. ESP32 dispune de:

până la 34 GPIO-uri utilizabile;

12 canale ADC cu rezoluție de 12 biți;

2 canale DAC pentru generare semnal analogic;

suport pentru PWM pe toți pinii digitali;

10 canale de touch sensing capacitiv;

interfețe seriale dedicate: 3 UART, 2 I²C, 3 SPI, 2 I²S, SDIO.

Nivelul logic este de **3.3V**, iar pinii **GPIO6 – GPIO11** nu sunt utilizați, fiind conectați intern la memoria flash SPI.

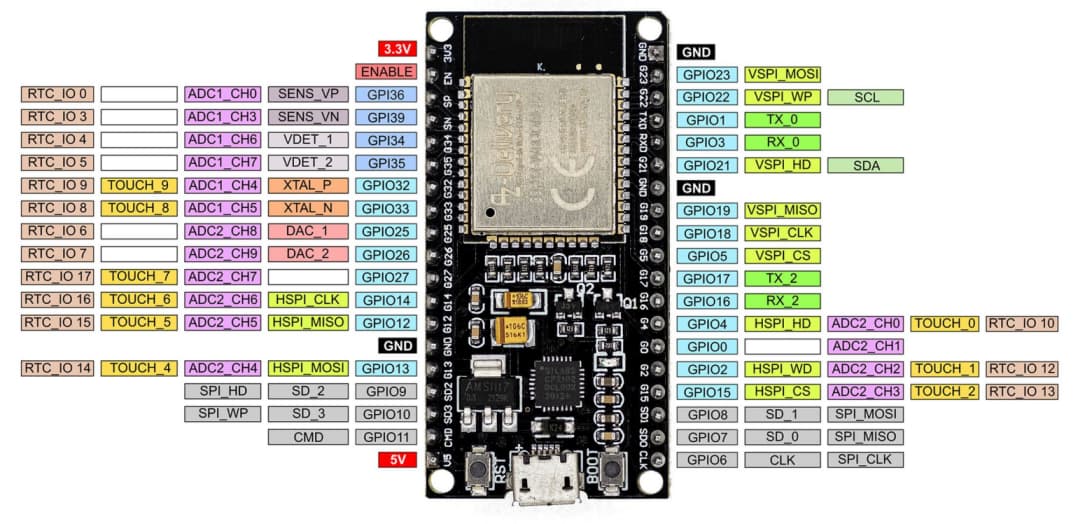
**Figura 4.2 – Diagrama pinout ESP32-WROOM-32D – *(sursa: [5])***

Diagrama pinout evidențiază distribuția clară a celor 38 de pini fizici. Pinii sunt organizați în două coloane laterale, corespunzătoare layout-ului plăcii ESP32 DevKitC. Culorile marchează funcțiile principale:

**Albastru deschis – GPIO uzual**  
Acestea sunt pini care pot fi folosiți în mod flexibil, fie ca **input**, fie ca **output digital**, fără restricții speciale.

**Verde – comunicație (UART, SPI, I2C)**  
Pini dedicați pentru protocoale seriale:

**UART** – comunicație serială (ex. RX, TX pentru Serial Monitor)

**SPI** – comunicație rapidă master-slave (ex. senzori, SD card)

**I²C** – comunicație cu mai mulți senzori pe doi pini (SDA, SCL)

**Mov–ADC(Analog-to-Digital Converter)**

Pini care pot citi semnale analogice (ex. variația de tensiune de la un LDR sau senzor de ploaie). Rezoluție de 12 biți.

**Roz – DAC (Digital-to-Analog Converter)**  
Pini care pot genera semnal analogic (ex. pentru difuzoare sau control fin al luminozității LED-urilor). Sunt doar 2 pini: GPIO25 și GPIO26.

**Galben – touch sensing (atingere capacitivă)**  
Pini care pot detecta atingerea unui deget sau a unui obiect condutiv. Sunt folosiți pentru implementarea unor „butoane” capacitive.

**Gri – SPI flash (de evitat)**  
GPIO6 până la GPIO11 sunt conectați **intern la memoria flash** și **nu trebuie utilizați în proiecte externe**. Pot cauza instabilitate sau boot failure dacă sunt forțați electric.

Pentru a compara în mod obiectiv performanțele și capabilitățile plăcii ESP32-WROOM-32D, am luat în calcul câteva dintre cele mai populare plăci de dezvoltare folosite în proiecte embedded: Arduino UNO, ESP8266 NodeMCU și Raspberry Pi Pico. Comparativ cu Arduino Uno, ESP32 oferă resurse hardware superioare ***(sursa: [20])***. Criteriile avute în vedere au fost: puterea de procesare, capacitatea de memorie, numărul de pini disponibili, prezența sau absența conectivității wireless, precum și flexibilitatea la integrarea mai multor senzori. Această comparație se evidențiază in **Tabelul 3**.

Tabelul 3. ESP32 vs alte plăci populare

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Microcontroler** | **CPU** | **RAM** | **Wi-Fi / BT** | **GPIO-uri** | **Interfețe** | **Observații principale** |
| ESP32-WROOM-32D | Dual-core 240 MHz | 520 KB | Da / Da | 34 | UART, I2C, SPI, PWM, ADC, DAC | Performant, versatil, ideal pentru IoT |
| Arduino UNO | 8-bit, 16 MHz | 2 KB | Nu / Nu | ~20 | UART, I2C, SPI | Simplu, stabil, dar limitat |
| ESP8266 (NodeMCU) | Single-core 80 MHz | 160 KB | Da / Nu | ~17 | UART, I2C, SPI | Bun pentru proiecte simple cu Wi-Fi |
| Raspberry Pi Pico | Dual-core 133 MHz (ARM) | 264 KB | Nu / Nu | ~26 | UART, SPI, I2C, ADC | Puternic, dar necesită accesorii pentru Wi-Fi |

## 2.1.2 Placa de test (breadboard),buton tactil și elemente auxiliare

Breadboard-ul este un instrument esențial în faza de dezvoltare, deoarece permite realizarea rapidă a conexiunilor fără a fi necesară lipirea componentelor, oferind flexibilitate și ușurință în testare și modificare. Placa utilizată este un model clasic, cu 830 de puncte de contact, având două magistrale de alimentare pe fiecare parte și o zonă centrală extinsă pentru montarea circuitelor ***[6]***, cum arată **figura[5]**. Toate conexiunile s-au realizat cu **fire jumper tip male-male**, care asigură un contact ferm și o vizibilitate clară asupra traseelor. Pentru a oferi un minim de interacțiune cu utilizatorul, am integrat un **buton tactil capacitiv**, compatibil cu ESP32. Acesta nu are piese mecanice în mișcare și reacționează la atingerea ușoară a suprafeței, fiind mai fiabil decât un buton clasic. Alimentarea microcontrolerului ESP32 a fost realizată printr-un **cablaj USB – microUSB**, conectat la un port USB standard de pe laptop. ESP32 are un regulator intern care permite alimentarea directă la 5V prin USB, convertind tensiunea pentru a funcționa la 3.3V la nivel logic. În plus, portul USB este utilizat și pentru programarea plăcii din Arduino IDE.

**Descrierea produsului:**

Panou fără lipit (MB-102)

Dimensiune sarma: potrivit pentru 29-20 fire AWG

Dimensiune: 16,5 x 5,5 x 0,85 cm

Banda dublu fata in partea de jos. Poate fi fixat pe o suprafața plana

Va produce 5V si 3,5V atunci când se aplica 12 V CC

Curent de ieșire maxim: <700ma

Poate fi comutat la 0V, 3.3V, 5”

|  |  |
| --- | --- |
| Kit breadboard 830 gauri + 65 fire + modul tensiune alimentare MB102 | Buton cu o singura atingere, TTP223, Cu pini, 3 V, Multicolor |

**Figura 5 – Breadboard mare buton tactil și set de fire jumper** *– sursă:*[*emag.ro/breadboard-alimentare*](https://www.emag.ro/kit-breadboard-830-gauri-65-fire-modul-tensiune-alimentare-mb102-tri273/pd/DY1YP6BBM/?ref=embedding_similar_model_1_10&provider=rec&recid=rec_102_f683d208fe000b4e94c8bc5786b98ee8a1f9fd7d637ff20a165374acfc974cef_1749675941&scenario_ID=102) ***[6]***

## 2.1.3 Ecran OLED 0.96” (I2C)

Pentru afișarea locală a parametrilor măsurați de stația meteo, am optat pentru un ecran OLED de 0.96”, monocrom, cu interfață I2C,(**figura [6]**). Alegerea a fost motivată de dimensiunile compacte, consumul redus de energie și lizibilitatea foarte bună a caracterelor, chiar și în condiții de iluminare slabă. Am considerat important ca sistemul să ofere o metodă de verificare vizuală rapidă a valorilor, fără a fi necesară conectarea la un calculator.

Dispozitivul a fost achiziționat de pe „**eMAG**”, fiind parte dintr-un set de afișaje OLED compatibile cu platformele Arduino. Descrierea de pe site-ul vânzătorului **(sursa:[8])** conține următoarele caracteristici:

„*Ecran tehnologie OLED 0.96”, compatibil cu Arduino din seria 51 Series, MSP430, STM32 / 2, CSR IC, etc. Are un consum redus de energie, de doar 0.08W. Luminozitatea si contrastul foarte înalt sunt reglabile. Are controler incorporat, iar tipul interfeței este IIC”*

**Caracteristici tehnice:**

„4 PINI: GND, VCC, SCL, SDA

Tensiune: 3V ~ 5V DC

Temperatura de lucru: -30 ℃ ~ 70 ℃

Rezoluție înalta: 128 \* 64

Dimensiunile panoului: 26,70 \* 19,26 \* 1,85mm / 1,03 \* 0,76 \* 0,07 inch (aprox)

Zona activa: 21.74 \* 11.2mm /0.86 /0.44 inch (aprox.)

Driver IC: SSD1306”



***Figura 6 – Ecran OLED 0.96” I2C pentru afișaj local*** *– sursă:*[*emag.ro/ecran-oled*](https://www.emag.ro/ecran-oled-0-96-ai409-s322-323-324/pd/D69S02MBM/) ***[8]***

## 2.1.3 Senzor BME280 – temperatură, umiditate și presiune

Dintre toate componentele din proiect, senzorul **BME280** este cel mai important pentru partea de măsurători de mediu. Am ales acest modul deoarece oferă, într-un singur chip, trei parametri esențiali pentru o stație meteo: temperatura, umiditatea relativă și presiunea atmosferică. Inițial, am analizat alternative precum DHT22 (pentru temperatură și umiditate) și BMP180 (pentru presiune), însă BME280 le înlocuiește pe ambele, oferind precizie mai bună, calibrare internă, și o interfață I2C simplă, care economisește pini pe microcontroler. În plus, este suportat nativ în Arduino IDE prin biblioteci precum *Adafruit\_BME280*, ceea ce a făcut integrarea foarte rapidă ***(sursa: [10])***. Senzorul care a fost achiziționat de pe același site „**eMAG”** este dezvoltat de firma internațională „**Bosch**” și are recenzii bune la nivelul cumpărătorilor ***(sursa: [9])***, fiind descris astfel:

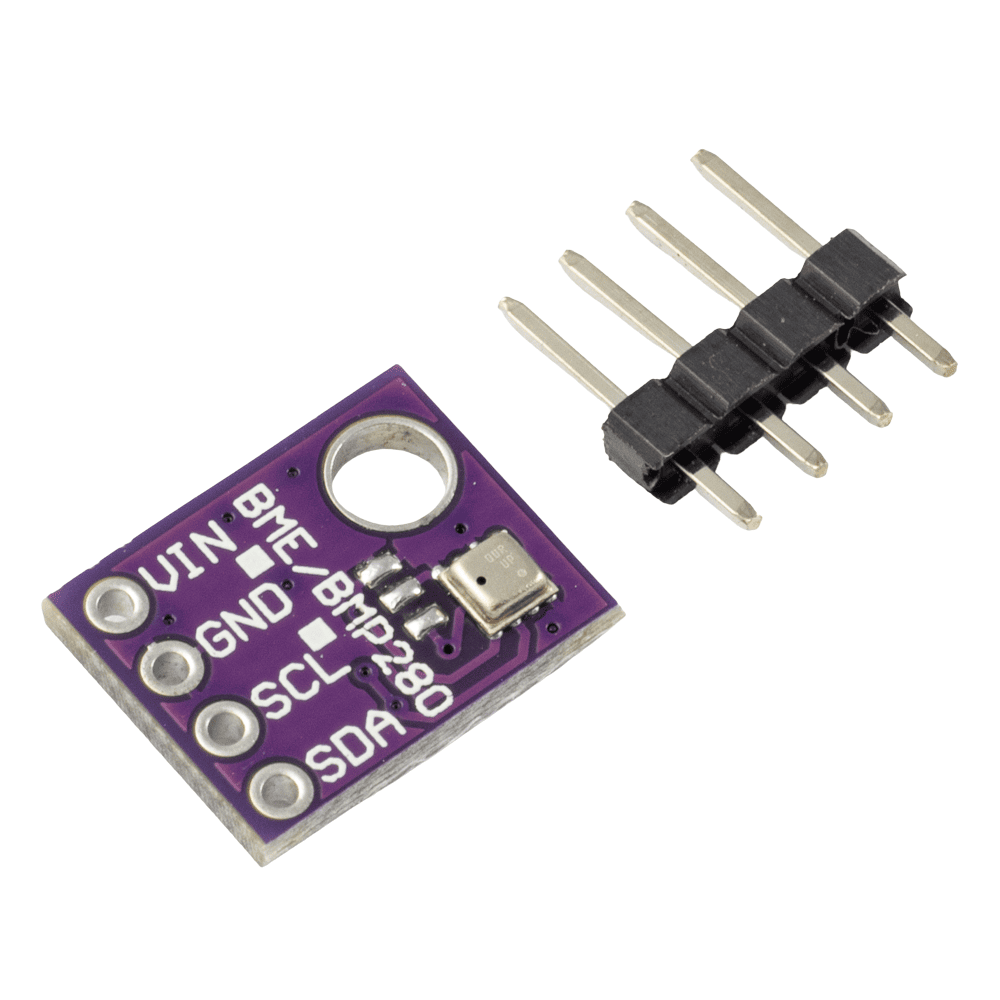
„*MODUL DE TEMPERATURA PRESIUNE UMIDITATE BME280 5V*

*Un modul mic cu senzorul****BME280 de la Bosch****. Este o dezvoltare a BMP280, fiind echipat suplimentar cu măsurarea umidității. Modulul este foarte mic. Este ideal pentru proiecte precum stații meteorologice, drone sau alte dispozitive alimentate cu baterii. Bosch BME280 permite măsurarea: Presiunea atmosferica, Temperatura, Umiditate*”

Tabelul 4. Caracteristici tehnice BME280

|  |  |
| --- | --- |
| **Parametru** | **Specificație** |
| Tip senzor | Digital, integrat (T + RH + Presiune) |
| Măsurare temperatură | -40°C ... +85°C, precizie ±0.5°C |
| Măsurare umiditate | 0 ... 100% RH, precizie ±3% |
| Măsurare presiune | 300 ... 1100 hPa, precizie ±1 hPa |
| Interfață comunicație | I2C sau SPI (în funcție de versiunea modulului) |
| Tensiune de operare | 1.8V – 3.6V (modul compatibil și cu 5V logic) |
| Dimensiuni modul | ~12 x 10 mm |
| Curent consumat (mediu) | ~1.8 µA în standby, ~1.8 mA activ |
| Alte funcții | Calibrare internă, filtrare digitală |

**Caracteristici generale:**

* Tensiune alimentare 5 V
* Lungime 15 mm
* Lățime 10 mm
* Culoare Violet

**Figura 7 –Modul BME280 pentru măsurarea temperaturii, umidității și presiunii** – *sursă:*[*emag.ro/bme280*](https://www.emag.ro/modul-bme280-5v-pentru-senzorul-de-temperatura-presiune-si-umiditate-arduino-5904501666077/pd/D13YFGYBM/)

## 2.1.4 Foto-rezistor (LDR) – senzor de luminozitate ambientală

Pentru detectarea nivelului de lumină din mediul înconjurător, am ales să folosesc un **LDR (Light Dependent Resistor)** – o soluție simplă, fiabilă și ușor de integrat în orice montaj electronic, prezentat in **figura[8]** . Am preferat această variantă în locul unui senzor digital de lumină, deoarece LDR-ul este accesibil, ușor de utilizat și oferă un semnal analogic proporțional cu intensitatea luminoasă. Din punct de vedere funcțional, un LDR își modifică rezistența electrică în funcție de cantitatea de lumină primită: cu cât lumina este mai intensă, cu atât rezistența scade. Conectat într-un **divizor de tensiune**, acest semnal poate fi citit de ESP32 printr-un **pin analogic**, interpretând astfel condițiile de lumină din mediul exterior ***(sursa: [11])***.

Tabelul 5. Caracteristici tehnice ale senzorului de lumină LDR

|  |  |
| --- | --- |
| **Parametru** | **Specificație aproximativă** |
| Tip componentă | Rezistor dependent de lumină (fotorezistor) |
| Tensiune de operare | 3.3V – 5V (prin divizor de tensiune) |
| Interval de rezistență | 1 kΩ (lumină puternică) – 10–20 kΩ (întuneric) |
| Tip semnal | Analogic (tensiune proporțională cu lumina) |
| Interfață | Pin analog ESP32 (ex: GPIO 36 / A0) |
| Conectare | Serie cu rezistor fix, în divizor de tensiune |



**Figura 8 – Foto-rezistor (LDR) – senzor de luminozitate ambientală**– *sursă:*[*emag.ro/ldr-5mm*](https://www.emag.ro/rezistor-sensibil-la-lumina-ldr-model-fotorezistor-5-mm-l141-ldr5/pd/D4SYM8YBM/)

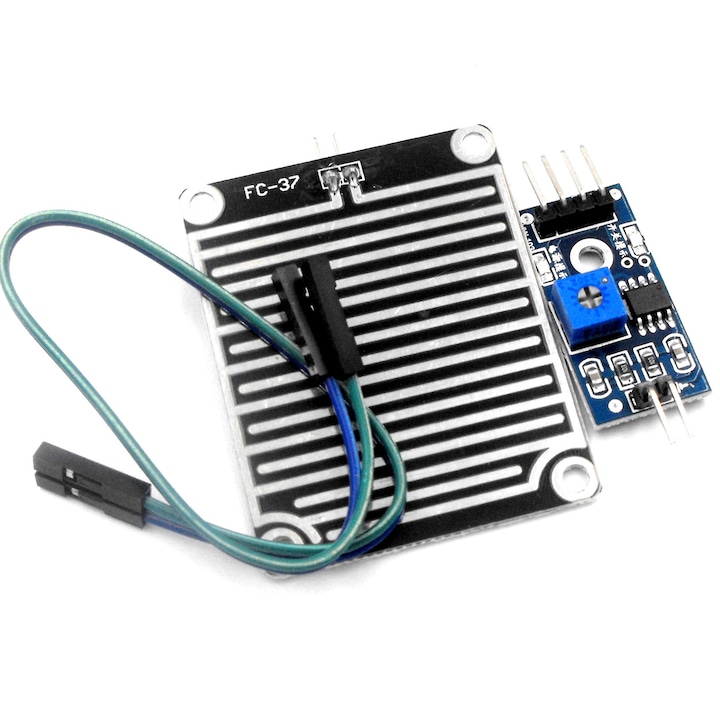
## 2.1.5 Senzor de ploaie capacitiv YL-83

Pentru detectarea prezenței ploii, am ales un **senzor de ploaie capacitiv**,(**figura 10**) care oferă o alternativă mai fiabilă față de modelele clasice cu urme metalice. Această decizie a fost luată pentru a evita problemele comune de oxidare sau scurtcircuit care pot apărea la senzorii cu trasee conductoare expuse, mai ales în utilizări de durată sau condiții în aer liber.

Senzorul capacitiv funcționează prin modificarea valorii capacitive în funcție de umiditatea acumulată pe suprafața sa. El oferă un **semnal analogic** proporțional cu nivelul de umezeală, care poate fi interpretat în software pentru a determina dacă a început să plouă. În unele modele, există și un **semnal digital** (HIGH/LOW) generat automat la atingerea unui prag, utilizabil ca întrerupător sau declanșator rapid ***(sursa: [12])***.

Tabelul 6. Caracteristici tehnice – senzor de ploaie capacitiv(sursa: [12])

|  |  |
| --- | --- |
| **Parametru** | **Specificație** |
| Tip senzor | Capacitiv (detecție fără contact metalic direct) |
| Semnal de ieșire | Analogic (0–3.3V) + opțional digital (HIGH/LOW) |
| Alimentare | 3.3V – 5V |
| Dimensiuni aproximative | 60 mm x 20 mm |
| Tensiune logică | 3.3V compatibilă cu ESP32 |
| Conectare | 3 pini (VCC, GND, A0/D0) |
| Avantaje | Rezistență la coroziune, fiabilitate ridicată |
| Limitări | Necesită protecție fizică la expunere îndelungată |



**Figura 10 – Senzor de ploaie capacitiv Yl -83cu ieșiri analogică și digitală** – *sursă:*[*emag.ro/senzor-ploaie*](https://www.emag.ro/senzor-de-ploaie-ajmaker-digital-analog-5v-30x1-6-mm-z000035/pd/DHH1LHYBM/)

## 2.1.6 Modul umidificator cu ultrasunete

După ce am integrat în proiect un senzor capacitiv pentru detecția ploii, am realizat că, pentru a-l testa eficient, am nevoie de o metodă sigură și controlabilă de a simula prezența picăturilor de apă. Nu voiam să folosesc apă direct turnată pe senzor, deoarece există riscul de scurtcircuitare a montajului sau de deteriorare a altor componente de pe breadboard. Astfel, am căutat o alternativă care să reproducă efectul umezelii, dar fără contact direct cu lichid.

Așa am ajuns să folosesc un **modul umidificator cu ultrasunete**, de mici dimensiuni, care generează vapori reci vizibili. Acesta funcționează pe principiul vibrației piezoelectrice la frecvență înaltă, transformând apa în ceață fină. ***(sursa: [12])***

Modulul are la bază un disc piezoelectric de 20 mm, cu o placă activă care vibrează în contact cu o cantitate mică de apă, transformând-o în vapori la temperatură joasă. Se alimentează la 5V printr-un port microUSB și este activat manual prin apăsarea unui buton capacitiv. Nu are piese în mișcare și este complet silențios în funcționare.

Este conceput pentru uz experimental/decorativ, dar se potrivește foarte bine în contextul testării componentelor electronice sensibile la umiditate.

Modulul a fost achiziționat de pe platforma **Bitmi.ro**, la prețul de **13,99 RON** (15 RON, inclusiv TVA), în luna mai 2025. În oferta magazinului, acesta este descris astfel:

„Modulul umidificator cu ultrasunete și port micro USB, 20 mm, oferă o soluție compactă și eficientă pentru adăugarea umidității în diverse medii, utilizând tehnologia ultrasunetelor pentru transformarea apei în particule fine de ceață.” ***(sursa: [13])***

Acest modul este recomandat pentru aplicații precum sisteme de climatizare, terarii, birouri sau uz casnic, fiind portabil și ușor de alimentat prin orice sursă USB (calculatoare, încărcătoare, power‑bank).

Datele tehnice complete furnizate de vânzător includ:

Tensiune de alimentare: **5 V DC** prin microUSB

Putere: **2 W**

Capacitate de dispersie a lichidului: **35 ml/h**

Nivel de zgomot: **< 36 dB**

Tabelul 6. Caracteristici tehnice – modulul umidificator cu ultrasunete

|  |  |
| --- | --- |
| **Parametru** | **Specificație** |
| Tip funcționare | Piezoelectric (ultrasunete) |
| Diametru disc | 20 mm |
| Alimentare | 5V DC prin port microUSB |
| Curent nominal | 300 mA (maxim) |
| Activare | Manuală (buton fizic integrat) |
| Tip abur generat | Rece, vizibil |
| Conectivitate logică | Nu este necesară |
| Utilizare tipică | Simulare de umiditate, testare senzori |



**Figura 11 – Modul umidificator cu ultrasunete-** *sursă:* [*bitmi.ro*](https://www.bitmi.ro/module-electronice/modul-umidificator-cu-ultrasunete-si-micro-usb-20mm-11246.html)

## 2.1.7 Modul RTC DS3231

Pentru a adăuga funcționalitatea de afișare a datei și orei în stația meteo, am utilizat un modul **RTC DS3231** (Real Time Clock), cunoscut pentru precizia sa ridicată și consumul redus de energie. Modulul a fost ales pentru că are ceas integrat cu cristal compensat termic, ceea ce îl face mult mai stabil decât variantele mai vechi (ex: DS1307), și include o **baterie CR2032**, care menține ora și data chiar și atunci când dispozitivul este deconectat de la alimentare.DS3231 comunică prin interfață I2C, fiind foarte ușor de conectat la ESP32. În cadrul proiectului, acesta a fost folosit pentru a afișa ora curentă pe ecranul OLED, dar și pentru a sincroniza citirea senzorilor în funcție de momentul zilei.

Site-ul oficial al motorului de achiziție „**eMAG”** oferă detalii legate de produs:

„*Modul de ceas in timp real foarte precis, cu memorie incorporată și cuarț. Are un slot pentru baterii CR2032.Modulul permite citirea orelor, minutelor, secundelor și a datelor: lună, zi, an. Comunicarea are loc prin intermediul magistralei I2C.Perfect pentru proiectele Arduino sau Raspberry Pi. Nota: atunci când se aplică o sursă de alimentare externă de 5V, tensiunea este transferată la baterie, in acest caz trebuie folosită doar o baterie reîncărcabilă.*” ***(sursa: [15])***

DS3231 este un modul robust și precis, ideal pentru orice sistem care necesită temporizare pe termen lung fără dependență de alimentare constantă (vezi **figura 11**).

Tabelul 7. Caracteristici tehnice ale modulului RTC DS3231

|  |  |
| --- | --- |
| **Parametru** | **Specificație** |
| Tip modul | Ceas în timp real (RTC) cu cristal compensat |
| Precizie | ±2 ppm (până la ~1 minut pe an) |
| Comunicație | I2C (SDA/SCL) |
| Tensiune de operare | 3.3V – 5V |
| Baterie backup | CR2032 (3V) – permite menținerea orei fără alimentare |
| Interval oră/dată | 00:00:00 – 23:59:59 / calendar complet |
| Alte funcții | Alarmă programabilă, ieșire SQW (semnal pătrat) |
| Compatibilitate | Arduino IDE, biblioteci: RTClib, DS3231.h |
| Dimensiuni modul | ~38 x 22 mm |

**Bateria CR2032** (**figura 11**).este de tip litiu, 3V, foarte comună și fiabilă pentru aplicații de joasă putere, cum este un RTC. Având o capacitate medie de aproximativ 210 mAh, această baterie poate asigura alimentarea modului DS3231 pentru câțiva ani, în regim de consum redus.

|  |  |
| --- | --- |
| Modul RTC DS3231/EEPROM 24C32, Multicolor | Baterie tip buton CR 2032, 230 mAh , 3V |

**Figura 11 – Modul RTC DS3231 cu baterie CR2032 pentru menținerea timpului real** – *sursă:*[*emag.ro/ds3231*](https://www.emag.ro/rtc-cu-batetrie-elektroweb-ds3231-2-e-058/pd/DLZMTMYBM/)

# FUNCȚIONAREA INDIVIDUALĂ A COMPONENTELOR

În acest capitol sunt prezentate pașii prin care am verificat **funcționarea corectă a fiecărei componente** hardware înainte de integrarea într-un sistem complet. Testarea individuală este esențială pentru identificarea timpurie a erorilor de conectare, a conflictelor de pini sau a incompatibilităților software, permițând o dezvoltare modulară și controlată a proiectului. Primul mod de testare este acela prin intermediul platformei online „**Wokwi – Arduino Simulator**”. Întregul capitol se împarte în mai multe subcapitole care reprezintă subansambluri de montaje hardware menite sa arate funcționalitatea diferitelor componente si moduri de implicare in proiectul final. Primul subansamblu este și cel mai necesar deoarece este format din componente necesare testării și validării celorlalte componente. Acesta cuprinde în primul rând elementele auxiliare de testare cum ar fi: **plăcuța breadboard** cu 830 de puncte de conectare dar și **fire jumper** pentru a facilita conexiunile. Pe lângă acestea nu poate lipsi „creierul” întregului sistem care face ca celelalte să funcționeze acesta fiind **Microcontrolerul ESP32-WROOM-32D** care asigură și alimentarea cu energie a întregului subsistem. La fel de necesar pentru validarea funcționalității este si modul de afișare care pentru această bază incipientă este reprezentat de **Ecran OLED 0.96” (I2C)** care este foarte facil de folosit. Pe lângă acest subsistem se adaugă si celelalte elemente descrise în capitolul anterior, aici punându-se accentul pe funcționalitatea lor și pe afișarea parametrilor pe care îi colectează.

## 2.2.1 Primul subsistem-ecranul OLED 0.96”

**Simularea subsistemului în „Wokwi”**

Înainte de realizarea montajului fizic, am testat funcționarea subsistemului format din **ESP32 și ecranul OLED 0.96"** folosind platforma online **Wokwi – Arduino Simulator**. Această etapă de simulare a fost utilă pentru a verifica codul sursă, adresele de comunicație I2C, bibliotecile utilizate și logica generală a afișajului, fără a risca deteriorarea componentelor reale. Simularea în Wokwi oferă multiple avantaje: permite testarea rapidă fără componente fizice, are afișaj OLED virtual integrat, suportă biblioteci Arduino direct în browser, oferă vizualizare a rezultatelor și este ideală pentru verificarea codului și a conexiunilor fără riscuri hardware.

**Pași realizați în simulare:**

Am accesat [wokwi.com](https://wokwi.com) și am selectat „New Project ,apoi ESP32”.

Am adăugat un modul OLED din bara laterală.

Am conectat componentele astfel:

**SDA** OLED către **GPIO 21**

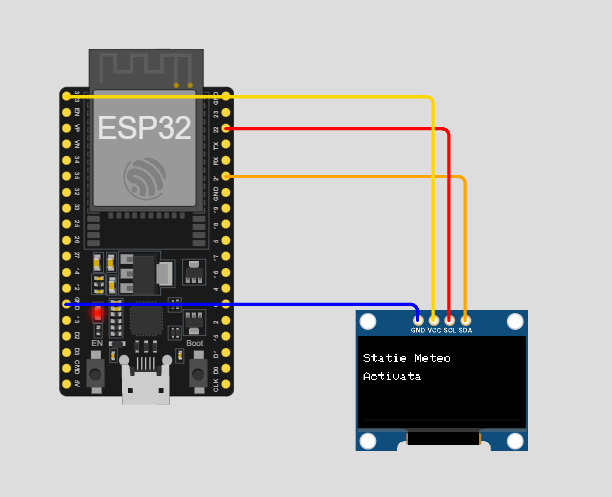
**SCL** OLED către **GPIO 22**

VCC la VCC (3.3V virtual)

GND la GND

Am încărcat codul de test, folosind biblioteca Adafruit\_SSD1306.h.

Am rulat simularea și am verificat afișajul textului „Stație Meteo Activat



**Figura 12 – Simularea subansamblului de bază**- *sursă: proiect personal Wokwi*

**Construirea fizică a subsistemului:**

Pentru toate testele realizate în cadrul acestui proiect, am construit un **subsistem de bază** care stă la fundamentul fiecărui montaj parțial. Acesta este compus din:

placa de test (breadboard) – suport fizic pentru toate componentele,

firele de conexiune (jumper) – pentru alimentare, semnal și împământare,

microcontrolerul ESP32-WROOM-32D – nucleul de procesare și comunicație,

ecranul OLED 0.96” – elementul de afișare locală a valorilor sau reacțiilor din test.

**Alimentarea sistemului:**

Alimentarea microcontrolerului ESP32 s-a realizat prin cablul microUSB conectat la un port USB al laptopului. Acesta oferă tensiunea de 5V, iar modulul ESP32 are un regulator intern de tensiune care transformă 5V în 3.3V, necesar funcționării interne și pentru alimentarea senzorilor și a ecranului OLED.

Pe breadboard, alimentarea a fost distribuită astfel:

Linia galbenă de alimentare (VCC) conectată la pinul 3.3V al ESP32,

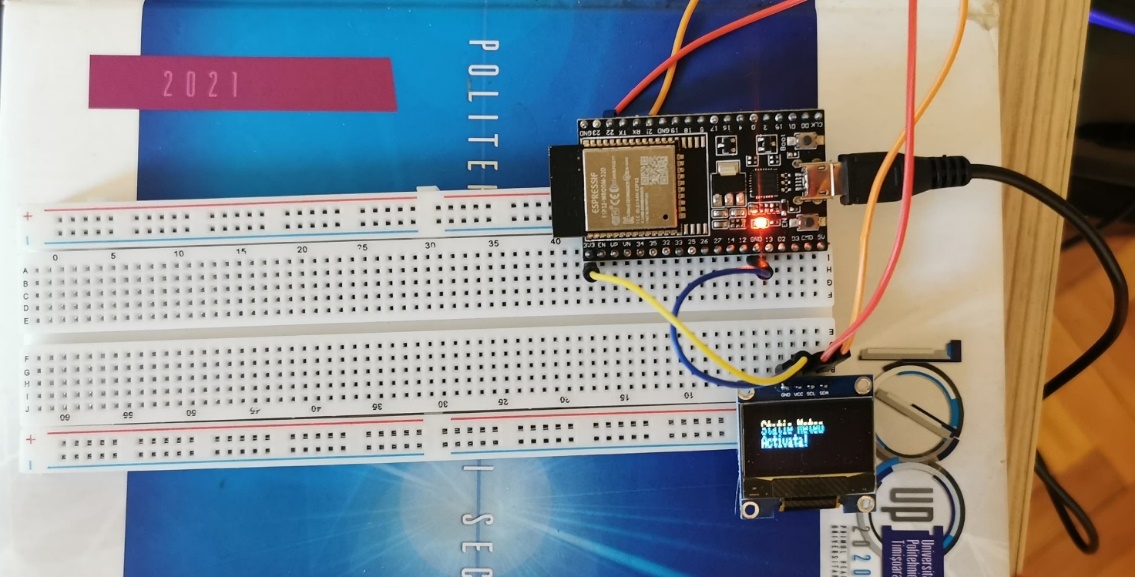
Linia albastră de masă (GND) conectată la pinul GND de pe ESP32.

Această configurație, prezentată în **figura 13**, a permis alimentarea simultană a ecranului OLED și, mai târziu, a celorlalte componente adăugate treptat.

Tabelul 8.Conexiuni pentru afișajul OLED *[8]*

| **Pin OLED** | **Funcție** | **Conectare la ESP32** |
| --- | --- | --- |
| VCC | Alimentare | 3.3V |
| GND | Împământare | GND |
| SDA | Date | GPIO 21 (SDA – standard ESP32) |
| SCL | Ceas (clock) | GPIO 22 (SCL – standard ESP32) |

Schema prezentată mai jos în **figura 13** este ușor de implementat și compatibilă cu biblioteca Adafruit\_SSD1306 din Arduino IDE ***[16]***.



**Figura 13 –montajul fizic OLED 96’**- *sursă: fotografie proprie*

**Pașii de realizare a montajului:**

Am așezat **ESP32** pe partea dreaptă a breadboard-ului, astfel încât toți pinii să fie accesibili pentru conectare rapidă. "Conectarea a fost realizată pe baza diagramei oficiale de pinout ***[5]***."Am conectat alimentarea și comunicația OLED-ului folosind **fire jumper** în culorile următoare:

**Galben** de la pinul **3V3** al ESP32 la pinul **VCC** al OLED.

**Albastru** de la pinul **GND** al ESP32 la pinul **GND** al OLED.

**Roșu** de la pinul **GPIO 22 (SCL)** al ESP32 către pinul **SCL** al OLED.

**Portocaliu** de la pinul **GPIO 21 (SDA)** al ESP32 către pinul **SDA** al OLED.

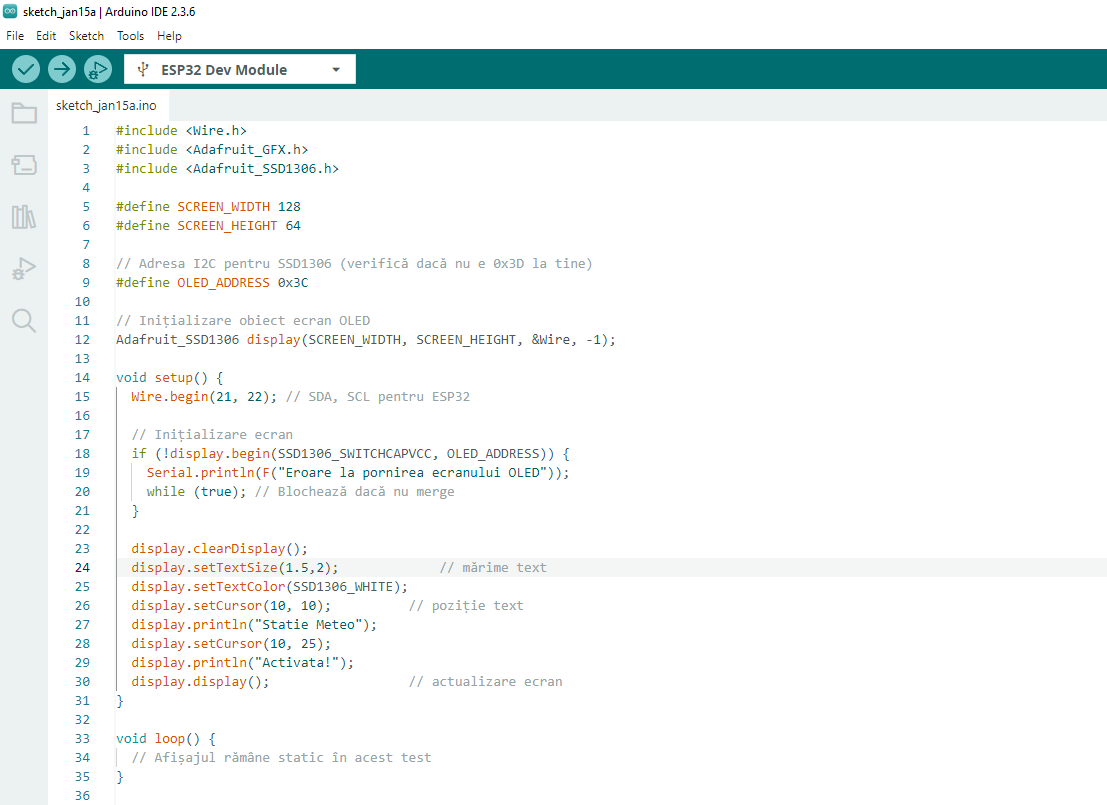
Am conectat ESP32 la laptop prin **cablul microUSB** (negru) pentru alimentare și programare.

Ecranul OLED a fost montat în partea inferioară a breadboard-ului, oferind vizibilitate clară în timpul testelor.

Conexiunile au fost verificate conform schemei și au fost stabilite fără lipire, pentru testare rapidă.

**Codul de test și bibliotecile utilizate pentru montajul fizic:**

Pentru a verifica funcționarea corectă a ecranului OLED și compatibilitatea acestuia cu microcontrolerul ESP32, am realizat un test software simplu în mediul de dezvoltare **Arduino IDE**. Obiectivul acestui test a fost să afișez un mesaj personalizat, static, pe două linii, pentru a mă asigura că afișajul primește date corect prin interfața I2C.Am folosit biblioteca grafică **Adafruit\_SSD1306**, împreună cu **Adafruit\_GFX și Wire**, care sunt necesare pentru controlul afișajului și gestionarea comunicației I2C ,conform documentației producătorului ***[8]****.*Aceste biblioteci sunt compatibile cu ESP32 și sunt ușor de instalat din Library Manager-ul Arduino.



**Secvența de cod 1** – **Codul pentru funcționarea Ecranului OLED 0.96**

**Explicații:**

Liniile 1–3 – Includerea bibliotecilor necesare pentru afișajul OLED și comunicația I2C.

Liniile 5–7 – Definirea dimensiunilor ecranului și a adresei I2C standard (0x3C).

Linia 10 – Inițializarea obiectului display care controlează OLED-ul.

Linia 13 – Inițializarea magistralei I2C pe pinii ESP32: SDA (21), SCL (22).

Liniile 16–20 – Verificarea conexiunii cu OLED-ul. Dacă afișajul nu răspunde, programul se oprește.

Linia 23 – Curățarea ecranului pentru a evita afișajul anterior.

Linia 24 – Setarea dimensiunii textului (simulată, nu este standard Arduino).

Linia 25 – Setarea culorii textului (alb).

Liniile 26–28 – Poziționarea textului pe ecran și scrierea celor două linii („Statie Meteo” și „Activata”).

Linia 29 – Comanda finală de afișare a textului pe ecran.

Liniile 32–33 – Funcția loop() este goală, deoarece afișajul nu se actualizează dinamic în acest test.

După verificarea comunicării corecte între placă și afișaj, mesajul **„Stație Meteo”** și **„Activata”** este redat pe ecran pentru confirmarea vizuală a funcționării. Codul este executat o singură dată, fără actualizare dinamică, fiind utilizat exclusiv ca test de validare hardware și software.

## 2.2.2 Al doilea subsistem-senzorul „BME280”

**Simularea subsistemului în „Wokwi”:**

La fel ca la montajul anterior am ales asamblarea si testarea inițială a senzorului digital **BME280** prin intermediul platformei online **Wokwi – Arduino Simulator.** Am ales să utilizez această platformă datorită ușurinței de utilizare, accesului rapid la componente virtuale și posibilității de a testa codul direct în browser, fără echipamente fizice. Deoarece platforma Wokwi, utilizată pentru simulare, **nu include implicit senzorul BME280 ca dispozitiv prestabilit**, a fost necesară o soluție alternativă. Astfel, senzorul l-am adăugat **manual în** fișierul **diagram.json** al proiectului Wokwi, folosind următorul identificator: "***type": "wokwi-bme280"***

**Pași realizați în simulare:**

Am păstrat configurația existentă a subansamblului cu OLED, conectat la placa ESP32 prin pinii GPIO 21 (SDA) și GPIO 22 (SCL), folosind protocolul I2C.

Am adăugat senzorul BME280 în cadrul proiectului Wokwi, folosind identificatorul specific *"wokwi-bme280"* în fișierul diagram.json, deoarece acesta nu este disponibil implicit în interfața grafică a simulatorului.

Am realizat conexiunile fizice ale senzorului BME280 la placa ESP32 folosind fire jumper în următoarele culori:

**Galben** de la pinul **3V3** al ESP32 către pinul **VCC** al senzorului BME280;

**Albastru** de la pinul **GND** al ESP32 către pinul **GND** al senzorului;

**Roșu** de la pinul **GPIO 22 (SCL)** al ESP32 către pinul **SCL** al senzorului;

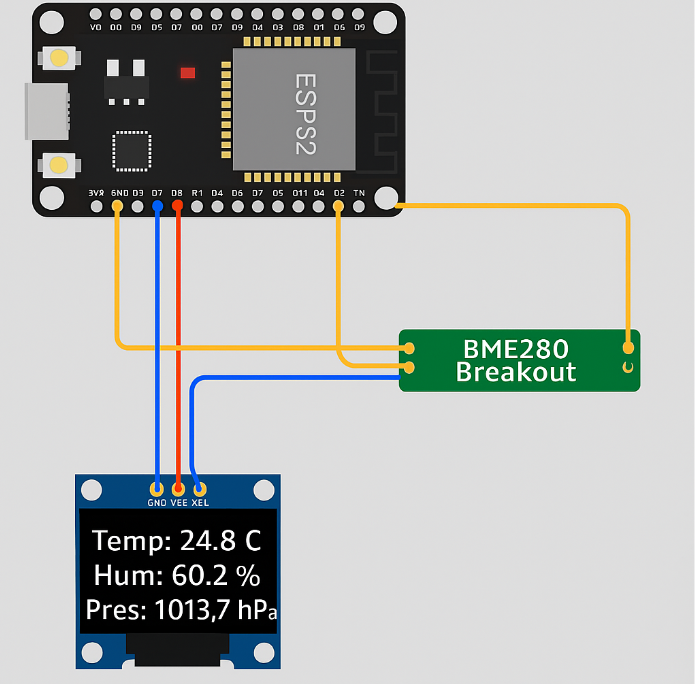
**Portocaliu** de la pinul **GPIO 21 (SDA)** al ESP32 către pinul **SDA** al senzorului.

Am conectat BME280 în paralel cu ecranul OLED, pe aceiași pini I2C (SDA și SCL), așa cum permite protocolul I2C.

Am adăugat în codul sursă inițializarea senzorului BME280, utilizând biblioteca ***Adafruit\_BME280.h*** și adresa sa I2C standard (0x76), astfel: ***bme.begin(0x76);***

Am completat funcția **loop()** pentru a prelua valorile măsurate de senzor (temperatură, umiditate, presiune) și a le afișa pe ecranul OLED deja existent.

Am testat funcționarea întregului subsistem în Wokwi, confirmând faptul că toate componentele I2C pot coexista și comunica simultan fără interferențe



**Figura 14 – Simularea subansamblului cu senzorul BME280** *sursă: proiect personal Wokwi*

**Construirea fizică a subsistemului cu BME280**

După validarea funcționării senzorului BME280 în simularea Wokwi, am trecut la implementarea practică a subsistemului pentru a testa comportamentul acestuia în condiții reale. Montajul a fost realizat pe placă de test (breadboard), având în vedere integrarea senzorului cu placa ESP32 deja utilizată pentru afișajul OLED.

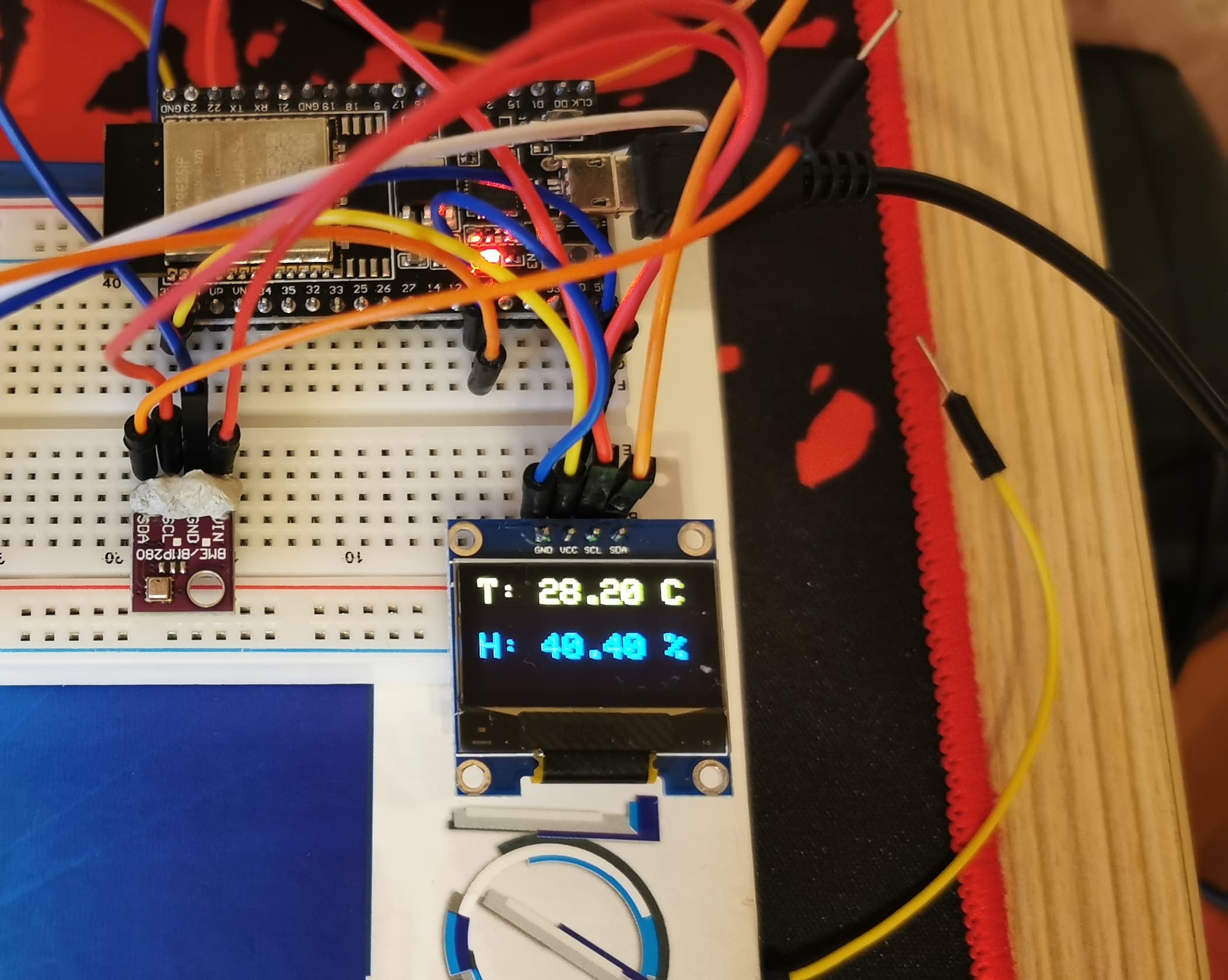
Am respectat aceleași principii de conectare ca în simulare, folosind interfața I2C și păstrând compatibilitatea între componente. Testarea fizică a avut ca scop observarea stabilității datelor măsurate și afișarea lor în timp real pe ecran și este prezentată în proiect prin **figura 15**.

**Alimentarea subsistemului**

Întregul ansamblu (ESP32, OLED și BME280) a fost alimentat prin cablul microUSB conectat la placa ESP32. Aceasta oferă o ieșire de 3.3V stabilă, care a fost utilizată pentru alimentarea senzorului BME280. Deoarece senzorul funcționează corect în intervalul 3.3–5V, alimentarea directă de la pinul 3V3 al ESP32 este suficientă și sigură. Totodată, GND-ul comun asigură referința electrică pentru toate componentele.

Tabelul 9.Conexiuni pentru senzorul BME280

| Pin BME280 |  | Funcție | Conectare la ESP32 |
| --- | --- | --- | --- |
| VCC |  | Alimentare | 3.3V |
| GND |  | Împământare | GND |
| SDA |  | Date | GPIO 21 (SDA – standard ESP32) |
| SCL |  | Ceas (clock) | GPIO 22 (SCL – standard ESP32) |

****

**Figura 15 –montajul fizic senzorul BME280**- *sursă: fotografie proprie*

**Pașii de realizare a montajului:**

Am așezat ESP32 pe partea superioară a breadboard-ului, astfel încât toți pinii să fie ușor accesibili pentru conectare.

Am completat montajul anterior cu afișajul OLED prin integrarea senzorului BME280, care permite măsurarea temperaturii și umidității atmosferice. Conectarea acestuia s-a realizat prin intermediul unor fire jumper în următoarea configurație:

Din pinul **3.3V** al ESP32 către pinul **VIN** al senzorului BME280 – alimentare.

Din pinul **GND** al ESP32 către pinul **GND** al senzorului – împământare.

Din pinul **GPIO 22 (SCL)** al ESP32 către pinul **SCL** al senzorului – semnal de ceas.

Din pinul **GPIO 21 (SDA)** al ESP32 către pinul **SDA** al senzorului – semnal de date.

Senzorul BME280 și afișajul OLED partajează aceiași pini SDA și SCL pe magistrala I2C, conform standardului de comunicare. Montajul a fost realizat astfel încât legăturile să fie clare și accesibile în timpul testelor.

Senzorul BME280 a fost programat să furnizeze în timp real valorile de temperatură și umiditate, care sunt afișate direct pe ecranul OLED conectat. Afișajul este organizat pe două linii, cu text de dimensiune medie pentru o vizibilitate crescută:

pe prima linie se afișează temperatura ambientală în grade Celsius, în formatul

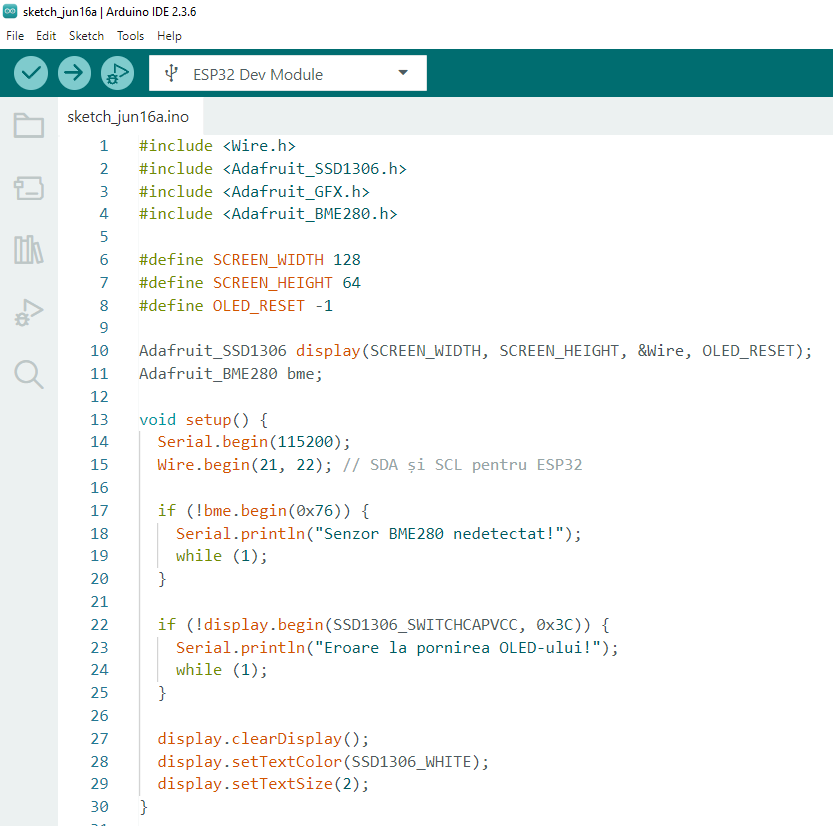
T: xx.xx C

pe a doua linie se afișează umiditatea relativă în procente, în formatul H: xx.xx %

Actualizarea valorilor se face periodic, la un interval de 2 secunde, pentru a permite o lectură clară și stabilă a parametrilor de mediu măsurați de senzor.

**Codul de test și bibliotecile utilizate pentru montajul fizic:**

Pentru a verifica funcționarea corectă a senzorului BME280 și compatibilitatea acestuia cu microcontrolerul ESP32 și afișajul OLED, am realizat un test software simplu, utilizând mediul de dezvoltare **Arduino IDE**. Obiectivul acestui test a fost să afișăm în timp real valorile măsurate pentru temperatură și umiditate, obținute de la senzorul BME280, pe un afișaj OLED I2C, pentru a confirma comunicarea corectă pe magistrala I2C. Senzorul BME280 a fost testat pe baza exemplului de cod oferit de producător ***[10]***.



**Secvența de cod 2.1** – **Codul pentru funcționarea senzorului BME280**

Pentru realizarea acestui test, au fost utilizate următoarele biblioteci:

**Wire.h** – pentru inițializarea și gestionarea comunicației I2C între module;

**Adafruit\_SSD1306.h** – pentru controlul afișajului OLED 0.96 inch;

**Adafruit\_GFX.h** – biblioteca grafică necesară afișajelor Adafruit;

**Adafruit\_BME280.h** – pentru interfațarea senzorului BME280 și citirea valorilor sale.

Aceste biblioteci sunt compatibile cu ESP32 și pot fi instalate cu ușurință direct din Library Manager-ul Arduino IDE.



**Secvența de cod 2.2** – **Codul pentru funcționarea senzorului BME280**

**Explicații:**

**Liniile 1–4** – Includerea bibliotecilor necesare pentru comunicația I2C (Wire), afișajul OLED (Adafruit\_SSD1306, Adafruit\_GFX) și senzorul BME280 (Adafruit\_BME280).

**Liniile 6–8** – Definirea dimensiunilor afișajului OLED (128x64) și a pinului de reset (nesetat, marcat cu -1).

**Liniile 10–11** – Inițializarea obiectelor pentru afișajul OLED (display) și senzorul BME280 necesare pentru interacțiunea cu componentele hardware.

**Linia 13** – Începutul funcției setup(), care rulează o singură dată la pornirea programului.

**Linia 14** – Inițializarea comunicației seriale pentru mesaje de debug.

**Linia 15** – Inițializarea magistralei I2C pe pinii standard ai ESP32: SDA (GPIO 21), SCL (GPIO 22).

**Liniile 17–19** – Verificarea conectării senzorului BME280. Dacă nu este detectat, se oprește execuția programului.

**Liniile 21–23** – Verificarea conectării afișajului OLED. Dacă nu este detectat, se oprește execuția programului.

**Liniile 25–27** – Ștergerea ecranului, setarea culorii textului și a dimensiunii caracterelor pentru afișajul OLED.

**Linia 30** – Începutul funcției loop(), care rulează repetitiv cât timp dispozitivul este alimentat.

**Liniile 31–32** – Citirea temperaturii și umidității de la senzorul BME280, cu stocarea rezultatelor în variabilele temp și hum.

**Liniile 34–35** – Curățarea ecranului înainte de afișarea de noi informații.

**Liniile 36–40** – Setarea poziției cursorului pe ecran și afișarea temperaturii în formatul T: xx.x C.

**Liniile 42–46** – Setarea poziției pentru linia a doua și afișarea umidității în formatul H: xx.x %.

**Linia 48** – Comanda de afișare efectivă a conținutului pe ecranul OLED.

**Linia 49** – Introducerea unei pauze de 2 secunde între afișări pentru o citire stabilă a valorilor.

## 2.2.3 Al treilea subsistem-afișare a orei în timp real – „RTC DS3231”

Subansamblul prezentat în această secțiune are rolul de a integra funcționalitatea de ceas în timp real în cadrul sistemului, folosind modulul RTC DS3231. Acesta permite determinarea exactă a datei și orei, chiar și în absența alimentării, datorită bateriei de backup integrate. Informația de timp este utilă în special pentru marcarea datelor senzorilor cu un „timestamp” precis sau pentru afișarea constantă a orei pe ecranul OLED.

Scopul acestui subansamblu este, așadar, să adauge o componentă de sincronizare temporală în proiect, contribuind la utilitatea practică și caracterul „inteligent” al sistemului final.

**Simularea subsistemului în „Wokwi”**

Pentru testarea funcționării acestui subsistem, am folosit platforma de simulare Wokwi, unde am integrat componentele necesare: placa ESP32, modulul RTC și ecranul OLED.

Menționez că, deși modulul vizat în lucrare este **DS3231**, acesta **nu este disponibil nativ în Wokwi**. Din acest motiv, am utilizat în simulare **modulul compatibil DS1307**, care folosește aceeași interfață I2C și aceeași bibliotecă software, RTClib. Astfel, simularea rămâne relevantă și validă pentru scopul demonstrativ al proiectului.

**Pași realizați în simulare:**

Am creat un nou proiect pe platforma Wokwi, selectând placa de dezvoltare ESP32.

Am adăugat modulul RTC DS1307 din panoul „Add Part” și l-am poziționat lângă ESP32.

Am realizat conexiunile necesare pentru interfața I2C:

**Roșu**: de la pinul **5V** al ESP32 către la pinul **VCC** al modulului RTC DS1307;

**Galben**: de la pinul **3.3V** al ESP32 către la pinul **VCC** al OLED-ului;

**Albastru**: de la pinul **GND** al ESP32 către la pinii **GND** ai ambelor module;

**Portocaliu**: de la pinul **GPIO 21 (SDA)** al ESP32 către la pinul **SDA** al ambelor module;

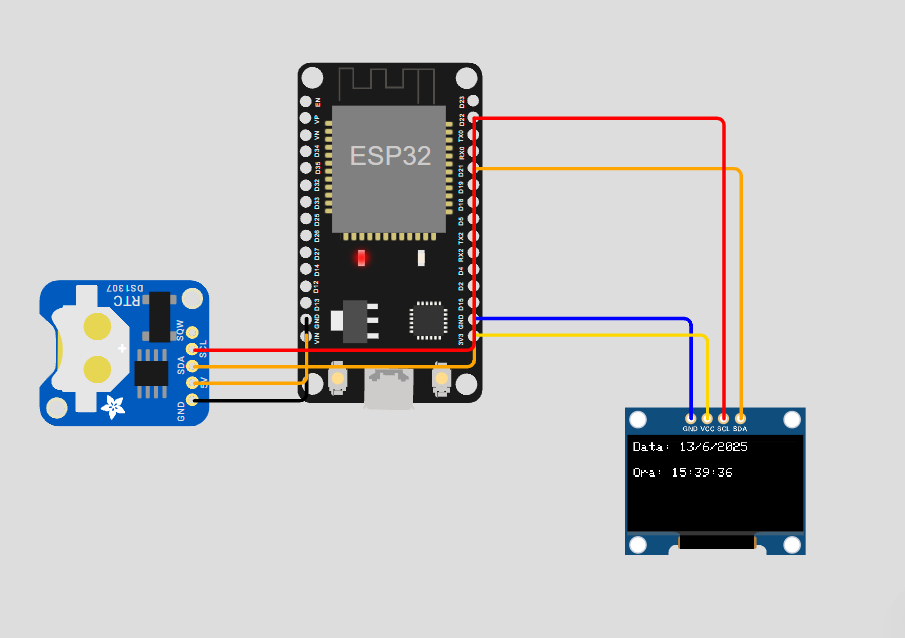
**Roșu deschis**: de la pinul **GPIO 22 (SCL)** al ESP32 către la pinul **SCL** al ambelor module.

Am adăugat ecranul OLED, folosind aceleași linii SDA și SCL (comune pe magistrala I2C).Am folosit biblioteca RTClib în codul Arduino, care permite compatibilitate atât cu DS1307, cât și cu DS3231 fără modificări majore.

Am încărcat un cod de test care afișează ora și data curentă pe OLED, confirmând astfel funcționarea completă a subsistemului.

Această simulare are rol demonstrativ și confirmă logica de integrare a modulului RTC într-un sistem cu mai multe dispozitive I2C conectate simultan.

Simularea subsistemului în platforma Wokwi a demonstrat cu succes integrarea modulului RTC DS1307 în rețeaua I2C comună cu OLED-ul, confirmând afișarea corectă a datei și orei în timp real. Montajul a fost realizat logic, utilizând alimentări corecte (5V pentru RTC, 3.3V pentru OLED), iar afișajul constant al informației temporale validează funcționalitatea completă a subsistemului, conform **figurii16** .



**Figura 16 – Simularea subansamblului cu „RTC DS3231”**-*sursă: proiect personal Wokwi*

**Construirea fizică a subsistemului cu modulul RTC DS3231:**

După simularea funcționării modulului RTC în Wokwi, am trecut la realizarea subsistemului în variantă fizică, utilizând modulul **DS3231**, un ceas de precizie superioară față de DS1307. Acesta păstrează ora și data chiar și în lipsa alimentării, datorită bateriei de backup integrate, și comunică cu placa ESP32 prin interfața I2C.

Pentru toate testele realizate în cadrul acestui proiect, am construit un **subsistem de bază** care stă la fundamentul fiecărui montaj parțial. Acesta este compus din:

Placă de dezvoltare **ESP32 DevKit v1**

Modul **RTC DS3231** (cu baterie CR2032)

Ecran OLED 0.96” I2C (deja montat)

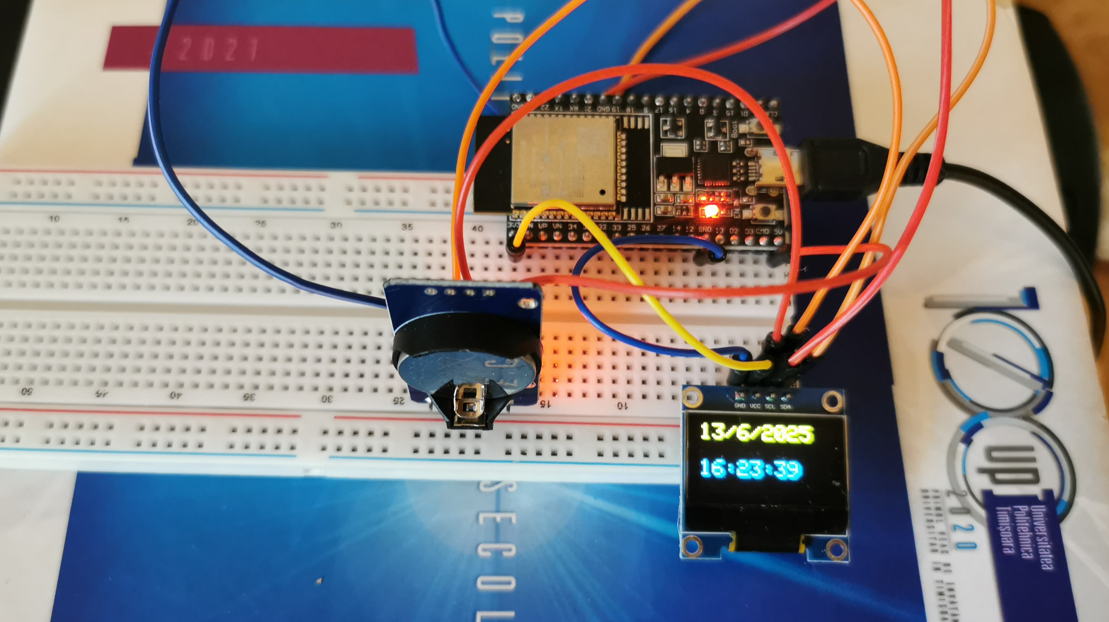
Fire jumper tata-tata

Breadboard

Modulul RTC DS3231 a fost testat pentru menținerea corectă a timpului real, în conformitate cu specificațiile producătorului **[15]**.Montajul fizic și afișarea este prezentată în **figura17**.

Tabelul 10.Conexiuni pentru modulul RTC DS3231

| Pin RTC DS3231 | Funcție | Conectare la ESP32 |
| --- | --- | --- |
| VCC | Alimentare | 5V |
| GND | Împământare | GND |
| SDA | Date | GPIO 21 (SDA – standard ESP32) |
| SCL | Ceas (clock) | GPIO 22 (SCL – standard ESP32) |



**Figura 17 – montajul fizic cu „RTC DS3231”**-*sursă: fotografie personală*

**Pașii de realizare a montajului:**

Montajul s-a realizat pornind de la configurația anterioară a subsistemului ce conținea deja afișajul OLED conectat la ESP32. Am completat ansamblul prin adăugarea modulului RTC DS3231, păstrând liniile comune de comunicație I2C (SDA și SCL) și adăugând alimentarea necesară ***[15]***.

Am așezat ESP32 pe partea superioară a breadboard-ului, astfel încât toți pinii să fie accesibili pentru conectare rapidă.

Am conectat alimentarea și comunicația modulului RTC DS3231 folosind **fire jumper** în culorile următoare:

**Portocaliu** de la pinul **5V** al ESP32 către pinul **VCC** al RTC-ului.

**Albastru** de la pinul **GND** al ESP32 către pinul **GND** al RTC-ului.

**Roșu** de la pinul **GPIO 22 (SCL)** al ESP32 către pinul **SCL** al RTC-ului (nod comun cu OLED-ul).

**Roșu** de la pinul **GPIO 21 (SDA)** al ESP32 către pinul **SDA** al RTC-ului (nod comun cu OLED-ul).

Am montat o baterie **CR2032** în suportul dedicat al modulului RTC pentru a păstra ora și data chiar și în lipsa alimentării externe.

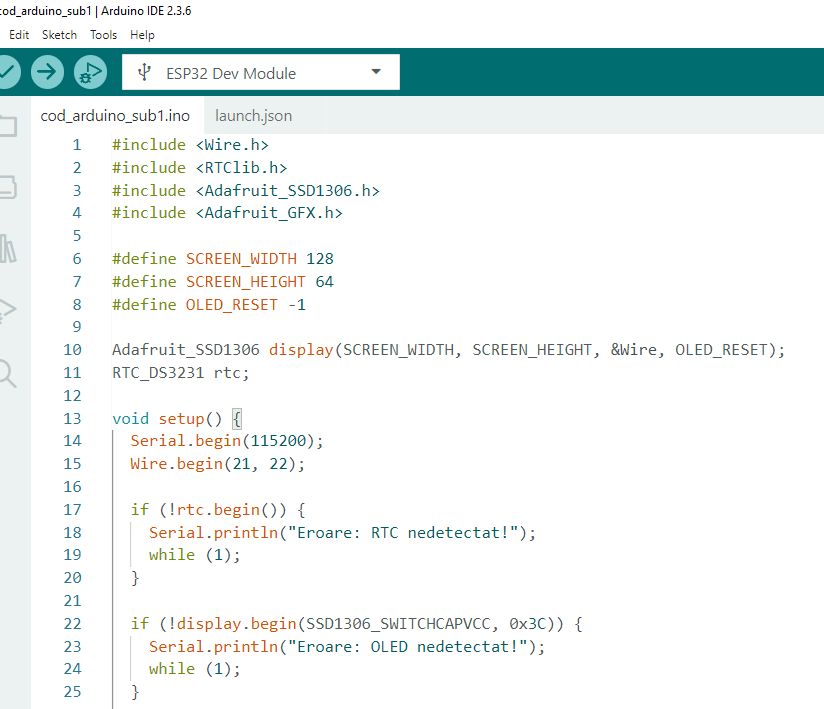
Am conectat ESP32 la laptop prin cablul microUSB (negru) pentru alimentare și programare. Conexiunile I2C au fost realizate în paralel cu afișajul OLED, respectând standardul de comunicație I2C, și au fost testate cu succes prin afișarea datei și orei curente pe ecranul OLED.

**Codul de test și bibliotecile utilizate pentru montajul fizic:**

Pentru a verifica funcționarea modulului RTC DS3231 în montajul realizat, am implementat un cod care inițializează ceasul în timp real și afișează **data** și **ora** curentă pe afișajul OLED, utilizând comunicația I2C.

La începutul codului sunt incluse bibliotecile necesare pentru utilizarea afișajului OLED (**Adafruit\_SSD1306, Adafruit\_GFX**) și pentru interfațarea cu modulul RTC (**RTClib.h**) ***[16]***.

În funcția **setup()**, este inițializat modulul RTC și se stabilește conexiunea cu afișajul OLED. Dacă aceste componente sunt detectate corect, ecranul este șters și apoi este pregătit pentru afișarea datelor. În bucla **loop()**, este obținută data și ora curentă de la RTC. Aceste informații sunt formatate și afișate pe ecran, unde data este afișată pe prima linie, iar ora curentă pe linia a doua. Textul este afișat într-o dimensiune lizibilă, folosind funcțiile de cursor și text ale bibliotecii **Adafruit\_GFX**.



**Secvența de cod 3.1**– **Codul pentru funcționarea „RTC DS3231”**-*sursă:proiect Arduino IDE*

**Secvența de cod 3.2**– **Codul pentru funcționarea „RTC DS3231”**-*sursă:proiect Arduino IDE*

**Explicații:**

**Liniile 1–4** – Includerea bibliotecilor necesare pentru comunicația I2C, afișajul OLED și modulul RTC DS3231.

**Liniile 5–7** – Definirea dimensiunilor afișajului OLED (128x64) și a pinului de reset, care nu este utilizat (setat la -1).

**Liniile 9–10** – Inițializarea obiectelor pentru controlul ecranului OLED (display) și a modulului RTC (rtc).

**Liniile 13–14** – Inițializarea comunicației seriale pentru monitorizare și a magistralei I2C pe pinii standard ai ESP32: SDA (GPIO 21), SCL (GPIO 22).

**Liniile 16–20** – Verificarea conectării corecte a modulului RTC. Dacă nu este detectat, afișează eroare în Serial Monitor și blochează execuția.

**Liniile 22–26** – Verificarea conectării afișajului OLED. În caz de eroare, afișajul nu pornește, iar execuția se oprește.

**Liniile 28–32** – Curățarea ecranului, setarea culorii și dimensiunii textului, poziționarea cursorului și afișarea mesajului „Incarcare...”.

**Linia 34** – Întârziere de 1.5 secunde pentru a permite vizualizarea mesajului de inițializare.

**Linia 37** – Începutul buclei loop(), care rulează continuu și actualizează ora și data.

**Linia 38** – Citirea orei curente de la modulul RTC și salvarea acesteia în variabila now.

**Linia 40** – Curățarea completă a afișajului înainte de actualizarea informației.

**Liniile 43–48** – Afișarea **datei** curente pe prima linie a ecranului, în formatul zi/lună/an.

**Liniile 50–56** – Afișarea **orei** curente pe a doua linie, în formatul HH:MM:SS, cu zero în față pentru valori sub 10.

**Linia 58** – Comandă pentru trimiterea efectivă a conținutului în bufferul grafic către ecranul OLED.

**Linia 60** – Întârziere de 1 secundă între afișări, pentru actualizare ritmică și lizibilă a ceasului.

## 2.2.4 Al patrulea subsistem-senzorul de ploaie capacitiv YL-83

Subansamblul prezentat în această secțiune are rolul de a detecta prezența ploii, folosind un semnal analogic preluat de la un senzor capacitiv de tip YL-83. Acesta este important pentru completarea datelor meteorologice locale, permițând sistemului să reacționeze la variațiile de umiditate de pe suprafață, semnalizând momentul în care încep precipitațiile.

**Simularea subsistemului în „Wokwi”**

Pentru testarea funcționalității senzorului de ploaie, am optat pentru o soluție alternativă, deoarece platforma **Wokwi** nu include nativ senzorul capacitiv YL-83. În locul acestuia, am folosit un **potențiometru**, care simulează perfect comportamentul analogic al senzorului de ploaie, în funcție de umiditatea detectată.

Valoarea analogică returnată de senzorul de ploaie în realitate (0 – 4095 pentru ESP32) este replicată de potențiometru prin rotirea cursorului acestuia, ceea ce permite simularea stării de **umed/uscat**.

**Pași realizați în simulare:**

Am creat un nou proiect pe platforma Wokwi, selectând placa de dezvoltare ESP32. Am adăugat un **potențiometru**, utilizat în rolul de „Senzor de ploaie”, precum și un ecran **OLED 0.96”** (I2C), ambele componente fiind poziționate în proximitatea plăcii ESP32,ca în **figura 18**

Am realizat următoarele conexiuni:

**Roșu:** de la pinul **3.3V** al ESP32 către la pinul **VCC** al OLED-ului și la pinul **VCC** al potențiometrului;

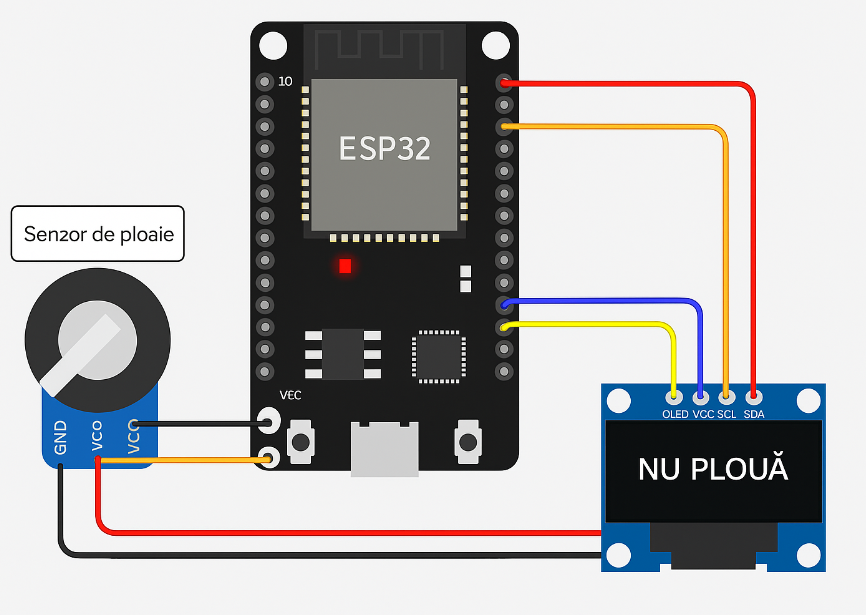
**Negru:** de la pinul **GND** al ESP32 către la pinul **GND** al OLED-ului și la pinul **GND** al potențiometrului;

**Galben:** de la pinul **GPIO21 (SDA)** al ESP32 către la pinul **SDA** al OLED-ului;

**Albastru:** de la pinul **GPIO22 (SCL)** al ESP32 către la pinul **SCL** al OLED-ului;

**Portocaliu:** de la pinul **GPIO36** al ESP32 către la pinul **central** al potențiometrului (ieșire analogică);

Am folosit același canal I2C pentru interconectarea ecranului OLED, iar semnalul de la potențiometru (considerat „senzor de ploaie”) a fost citit prin pin analogic ADC1\_CH0 (GPIO36).



**Figura 18 – Simularea subansamblului cu senzor de ploaie** *sursă: proiect personal Wokwi*

**Construirea fizică a subsistemului cu senzorul de ploaie**

După simularea subsistemului în mediul virtual Wokwi, am trecut la implementarea practică folosind un **senzor de ploaie capacitiv YL-83**, conectat pe un pin analogic al plăcii ESP32. Modulul a fost alimentat la 3.3V, iar semnalul său analogic a fost citit pe pinul GPIO36 (ADC1\_CH0), același utilizat și în simulare. Valoarea returnată de senzor scade pe măsură ce umezeala de pe suprafața sa crește. Această valoare este comparată cu un prag stabilit în cod, pentru a decide dacă **plouă** sau **nu plouă*, (sursa: [12]).*** Afișajul OLED este utilizat pentru a oferi utilizatorului feedback în timp real, prin mesaje clare. Pentru testarea funcționării senzorului de ploaie, am utilizat un subsistem de bază, similar celorlalte testări realizate în cadrul proiectului. Acesta este format din următoarele componente:

Placă de dezvoltare **ESP32 DevKit v1**

**Senzor de ploaie capacitiv YL-83**

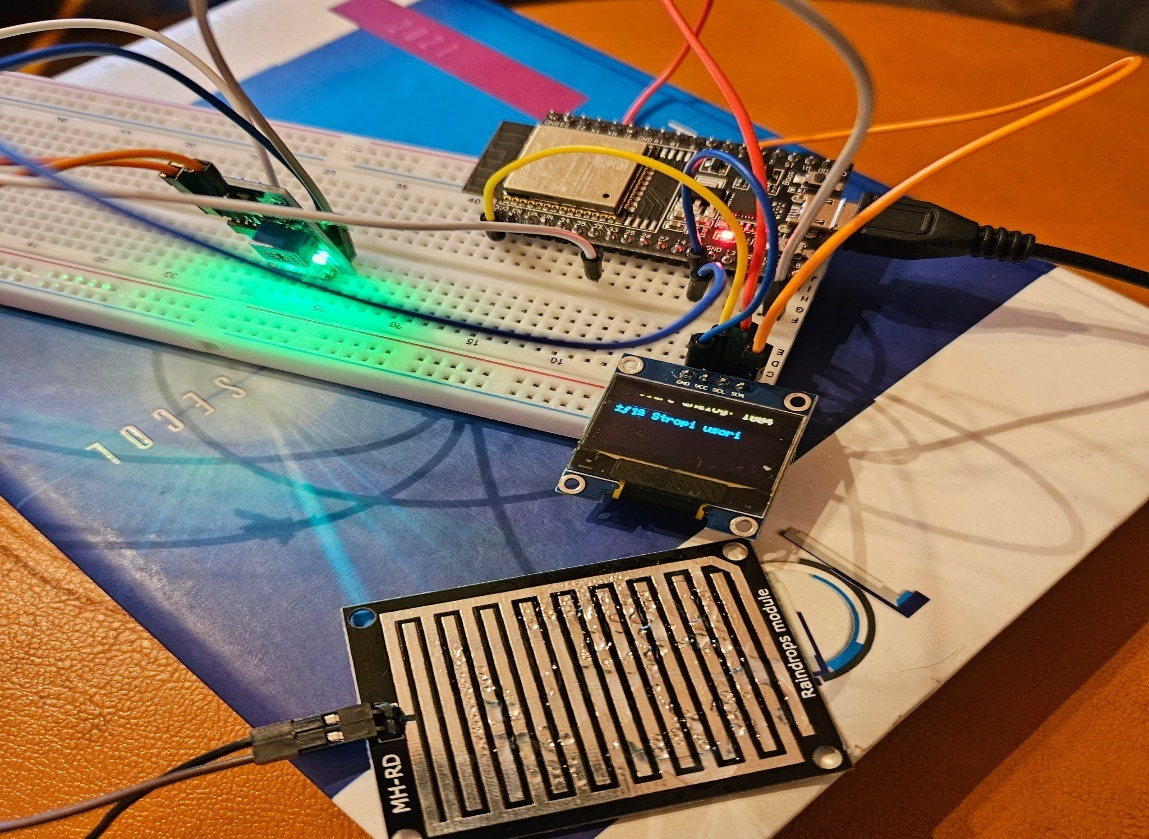
**Ecran OLED 0.96” I2C** (deja montat în subsistemele anterioare)

**Fire jumper tata-tata**

**Breadboard** pentru conexiuni rapide

**Alimentarea sistemului**

Alimentarea s-a realizat prin portul microUSB al plăcii ESP32, la fel ca în celelalte subsisteme. Toate componentele au funcționat la tensiunea de **3.3V**, inclusiv senzorul de ploaie capacitiv și afișajul OLED. Deoarece senzorul YL-83 este compatibil atât cu 3.3V, cât și cu 5V, s-a optat pentru alimentarea de 3.3V pentru a menține coerența sistemului și a evita posibile diferențe de nivel logic. Procesul este prezentat în **figura 19**.



**Figura 19 –montajul fizic cu senzorul de ploaie YL-83”**-*sursă: fotografie personală*

**Pașii de realizare a montajului fizic**

Montajul s-a realizat pornind de la configurația anterioară care includea afișajul OLED conectat la ESP32. Am completat subsistemul prin adăugarea senzorului de ploaie capacitiv YL-83, păstrând conexiunile I2C existente pentru ecran și adăugând un pin analogic dedicat pentru senzor ***[12]***.

Am așezat ESP32 pe partea superioară a breadboard-ului, astfel încât toți pinii să fie ușor accesibili pentru conectare. Conexiunile s-au realizat cu fire jumper de tip tata-tata, după cum urmează:

**Portocaliu:** de la pinul **3.3V** al ESP32 către la pinul **VCC** al senzorului de ploaie;

**Albastru:** de la pinul **GND** al ESP32 către la pinul **GND** al senzorului;

**Roșu:** de la pinul **GPIO36 (ADC1\_CH0)** către la pinul **A0** (ieșire analogică) al senzorului;

Afișajul OLED rămâne conectat pe linia I2C comună: **GPIO21 (SDA)** și **GPIO22 (SCL)**;

ESP32 a fost conectat la laptop prin cablul microUSB (negru), oferind atât alimentare, cât și programare. Montajul a fost testat prin aplicarea de apă pe senzorul capacitiv YL-83. Afișajul OLED a reacționat corect, afișând mesajul **„PLOUA”** în caz de umiditate și **„NU PLOUA”** în stare normală.



**Secvența de cod 3.2**– **Codul pentru funcționarea „RTC DS3231”**-*sursă:proiect Arduino IDE*

# 2.3. DESCRIEREA MONTAJULUI FINAL A STAȚIEI METEO

În acest capitol este prezentată forma finală a stației meteo realizate, atât din punct de vedere al conectării tuturor modulelor electronice, cât și din perspectiva integrării într-un montaj fizic testabil. Toate componentele utilizate în subcapitolele anterioare (modul de temperatură și presiune BME280, afișajul OLED 0.96”, modulul RTC DS3231, senzorul de lumină LDR și senzorul de ploaie capacitiv YL-83) sunt conectate împreună într-un ansamblu coerent, centralizat de microcontrolerul ESP32.

Capitolul este structurat în două subsecțiuni:

**Montajul final pe breadboard și conectarea modulelor la ESP32** –evidențiază conexiunile între componente și modul de funcționare al ansamblului pe placă de test.

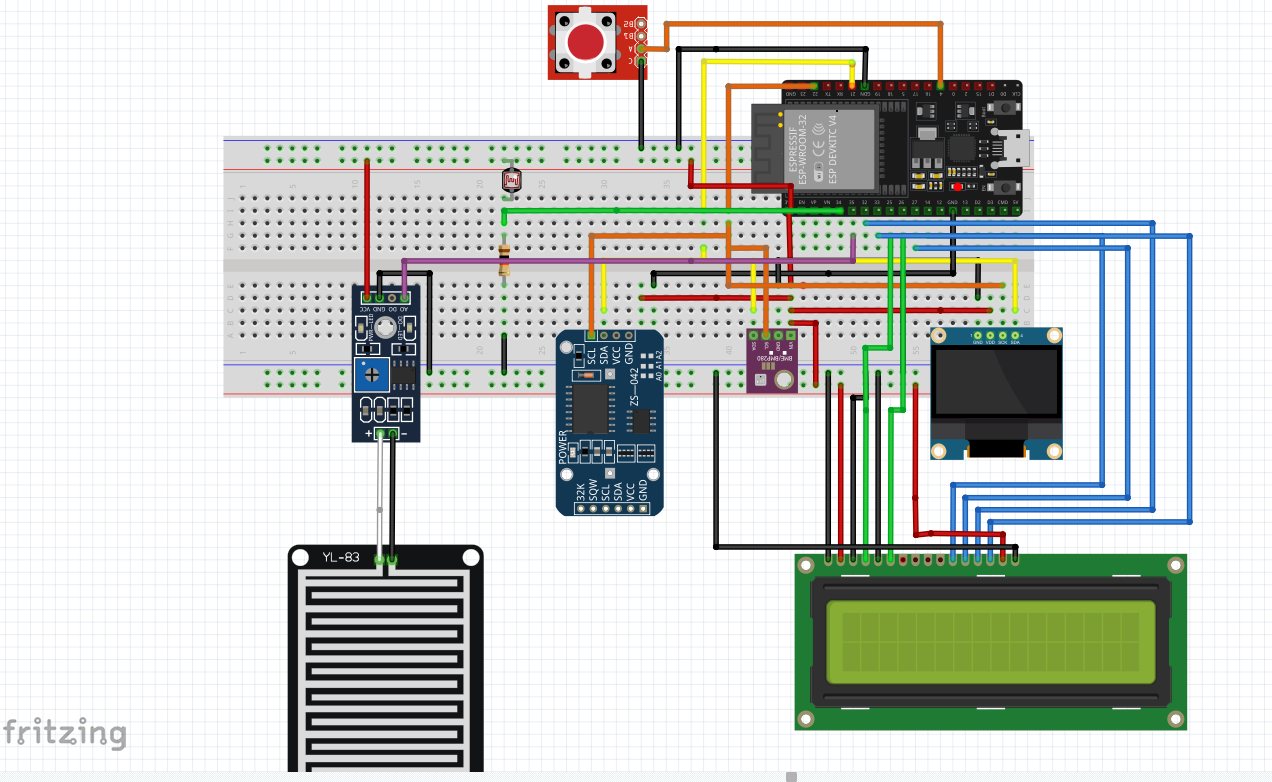
**Integrarea într-un mediu de testare cu simulare de ploaie** – prezintă montajul amplasat într-o carcasă și testat în condiții controlate.

## 2.3.1 Montajul final pe breadboard și conectarea modulelor la ESP32

Asemănător capitolelor anterioare, și pentru montajul final al stației meteo este realizată o schemă de conectare care reflectă modul în care toate modulele electronice sunt integrate într-un sistem unitar. Acest subcapitol are rolul de a prezenta atât organizarea fizică a componentelor pe o placă de test (breadboard), cât și modul în care fiecare dintre acestea este conectat la placa de bază ESP32.Se va include o schemă logică de conectivitate realizată în mediul **Fritzing** sau echivalent, un tabel cu toate conexiunile dintre senzori și microcontroler, precum și pașii urmați în realizarea montajului fizic. De asemenea, va fi explicată și funcționarea sistemului în ansamblu, cu accent pe modul în care datele colectate de senzori sunt afișate centralizat pe ecranul OLED.

**Simularea montajului într-un mediu grafic:**

Pentru reprezentarea grafică a montajului final s-a optat pentru utilizarea platformei Fritzing în locul simulatorului Wokwi, deoarece Wokwi nu oferă suport complet pentru toate componentele utilizate în proiect, precum senzorul BME280 sau RTC DS3231. În schimb, Fritzing permite integrarea vizuală a tuturor modulelor necesare, inclusiv ESP32, și oferă o imagine detaliată a conexiunilor realizate pe o placă de test (breadboard).Tot procesul este prezentat vizual în **figura 20** și în descrierile de mai jos.

Scopul acestei scheme este de a evidenția modul de conectare a fiecărui senzor și modul la placa ESP32, așezarea fizică a componentelor și traseele de alimentare și comunicație. Chiar dacă Fritzing nu permite simularea codului, schema rezultată este ****esențială pentru documentarea arhitecturii fizice a montajului final.

**Figura 20– Schema de conectare finală a stației meteo în Fritzing-** *sursă: proiect personal*

**Pașii realizați în simulare:**

Am început un nou proiect în platforma **Fritzing**, selectând placa de dezvoltare **ESP32 DevKit v1**, pe care am poziționat-o central pe o placă de test (breadboard). Ulterior, am adăugat toate modulele compatibile cu arhitectura finală, conform structurii fizice a stației meteo. Pentru a asigura o afișare permanentă și clară a informațiilor esențiale, precum data și ora curentă, a fost introdus și un afișaj LCD 1602, utilizat în paralel cu ecranul OLED, acesta având rolul de a menține vizibilă în mod constant sincronizarea temporală a sistemului, indiferent de ciclurile de afișare variabilă ale OLED-ului.

Conectarea modulelor s-a realizat respectând atât interfața I2C, cât și intrările analogice disponibile pe ESP32. Comunicația I2C a fost configurată în paralel, astfel încât toate dispozitivele compatibile să împartă aceeași linie SDA și SCL, optimizând astfel utilizarea pinilor digitali.

Am realizat următoarele conexiuni:

**Roșu:** de la pinul **3.3V** al ESP32 la pinii **VCC** ai OLED-ului, RTC-ului, BME280-ului, LCD-ului și modulului de ploaie;

**Negru:** de la pinul **GND** al ESP32 la pinii **GND** ai tuturor modulelor;

**Galben:** de la pinul **GPIO21 (SDA)** al ESP32 la pinii **SDA** ai OLED-ului, RTC-ului și BME280-ului;

**Albastru:** de la pinul **GPIO22 (SCL)** al ESP32 la pinii **SCL** ai acelorași module I2C;

**Portocaliu:** de la pinul **GPIO34** al ESP32 la divizorul de tensiune al LDR-ului;

**Mov:** de la pinul **GPIO35** al ESP32 la pinul A0 al modulului YL-83;

**Verde închis:** de la pinul **GPIO25** al ESP32 la pinul **RS** al LCD-ului;

**Verde deschis:** de la pinul **GPIO26** al ESP32 la pinul **E** al LCD-ului;

**Albastru închis:** de la pinii **GPIO14, GPIO27, GPIO32, GPIO33** la pinii **D4–D7** ai LCD-ului;

**Gri:** de la pinul **GPIO4** al ESP32 către pinul **A** al butonului; pinul **C** este conectat la GND.

Conexiunile au fost realizate după regulile din **Tabelul 11**, iar schema finală permite testarea completă a funcționalității stației meteo într-un mediu de simulare statică, înaintea montajului fizic.

Tabelul 11 – Conexiuni între ESP32 și modulele din montaj

|  |  |  |
| --- | --- | --- |
| Pin ESP32 | Conectat la | Componentă |
| 3.3V | VCC | OLED, RTC DS3231, BME280,  LCD 1602, YL-83 |
| GND | GND | Toate modulele (masă comună) |
| GPIO21 (SDA) | SDA | OLED, RTC DS3231, BME280 |
| GPIO22 (SCL) | SCL | OLED, RTC DS3231, BME280 |
| GPIO34 | Ieșire divizor LDR | LDR |
| GPIO35 | A0 | YL-83 (semnal analogic) |
| GPIO25 | RS | LCD 1602 |
| GPIO26 | E | LCD 1602 |
| GPIO14 | D4 | LCD 1602 |
| GPIO27 | D5 | LCD 1602 |
| GPIO32 | D6 | LCD 1602 |
| GPIO33 | D7 | LCD 1602 |
| GPIO4 | A | Buton digital |

**Pașii de realizare a montajului fizic:**

Placa ESP32 a fost amplasată în colțul din dreapta al breadboard-ului, cu pinii orientați pe partea interioară a plăcii. Alimentarea modulelor s-a realizat prin conectarea pinului **3.3V** al ESP32 la linia de alimentare (+) a breadboard-ului (cu fir roșu) și a unui pin **GND** la linia de masă (–) (fir negru).

Pentru comunicația I2C, am realizat un **nod fizic comun** din care pleacă firele spre toate modulele compatibile: din **GPIO21 (SDA)** al ESP32, un fir **portocaliu** merge către o zonă de pe breadboard unde am creat un punct comun, de unde pleacă alte fire **portocalii** către OLED, BME280 și RTC; în mod similar, din **GPIO22 (SCL)** al ESP32, un fir **galben** merge într-un nod de distribuție, de unde sunt trase fire **galbene** către pinii SCL ai acelorași trei module.

Modulul **OLED** a fost poziționat central, în partea inferioară a breadboard-ului, alimentat direct din liniile + și –, iar liniile sale SDA și SCL sunt conectate în paralel cu celelalte componente I2C, conform nodului anterior.

Senzorul **BME280** a fost amplasat în dreapta OLED-ului și conectat la aceleași fire de date I2C. Modulul **RTC DS3231** a fost poziționat în partea stângă a OLED-ului, legat și el la magistrala I2C. Toate cele trei module folosesc alimentarea comună de 3.3V (roșu) și GND (negru).

Modulul **senzorului de ploaie YL-83** a fost amplasat în colțul stânga-jos al breadboard-ului, iar:

pinul **A0** al acestuia este conectat la **GPIO35** al ESP32 cu fir **albastru** (pentru citire analogică);

VCC și GND sunt conectate la liniile de alimentare.

Senzorul de lumină (**LDR**) a fost integrat într-un **divizor de tensiune**, conectat pe breadboard astfel:

un capăt al LDR-ului este conectat la **linia de 3.3V (roșie)**;

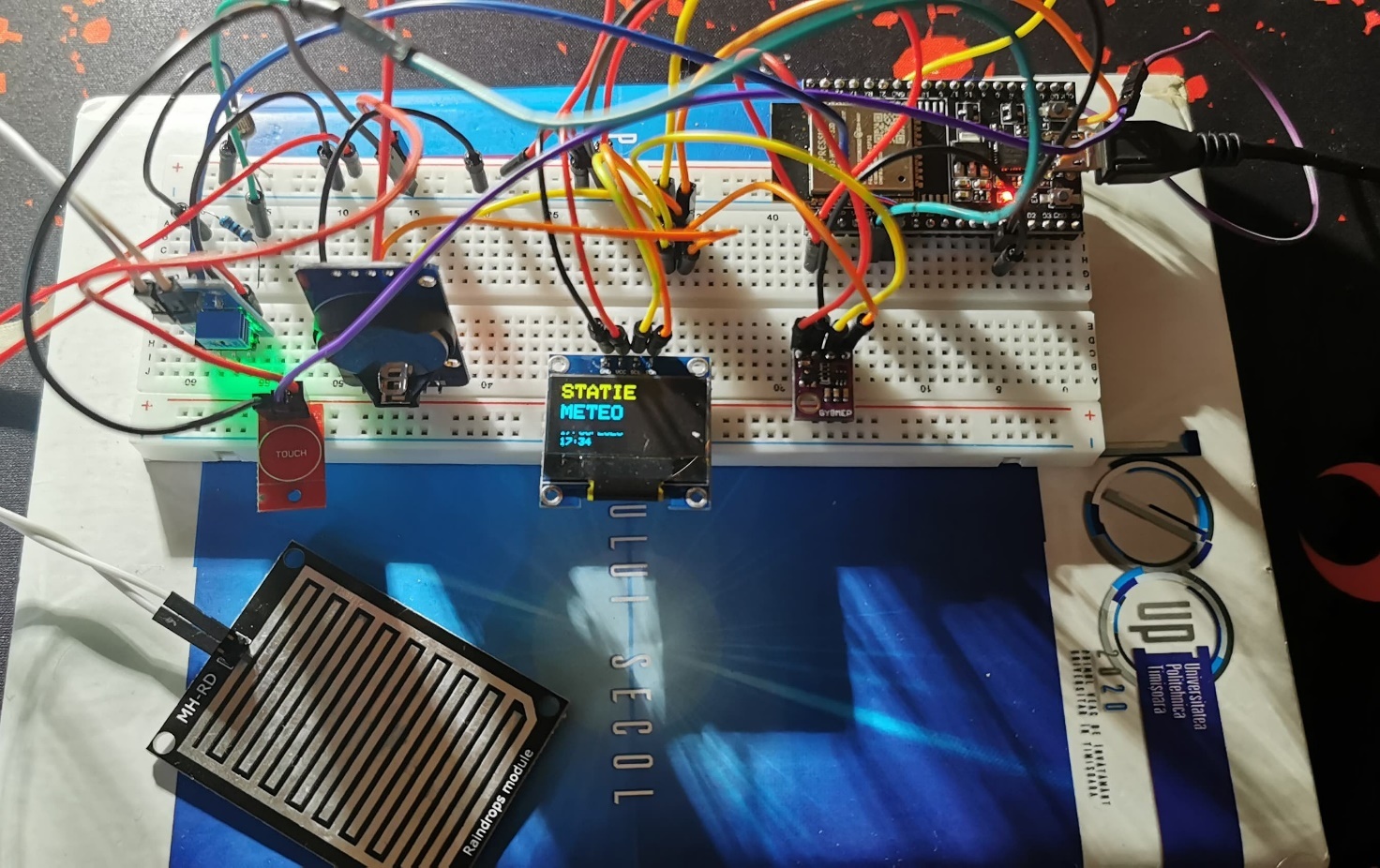
celălalt capăt este conectat la un rezistor de 10kΩ spre GND (linie albastră);

din punctul comun dintre LDR și rezistor pleacă un fir **verde închis** către **GPIO34**, folosit pentru citirea analogică a nivelului de lumină.

Butonul tactil (**modul TTP223**) este montat în partea frontală a breadboard-ului, iar:

pinul său **OUT** este conectat la **GPIO4** prin fir **mov**;

pinii VCC și GND sunt conectați la liniile laterale de alimentare.



**Figura 21– Conexiunea fizică finală a stației meteo-***sursă: fotografie personală*

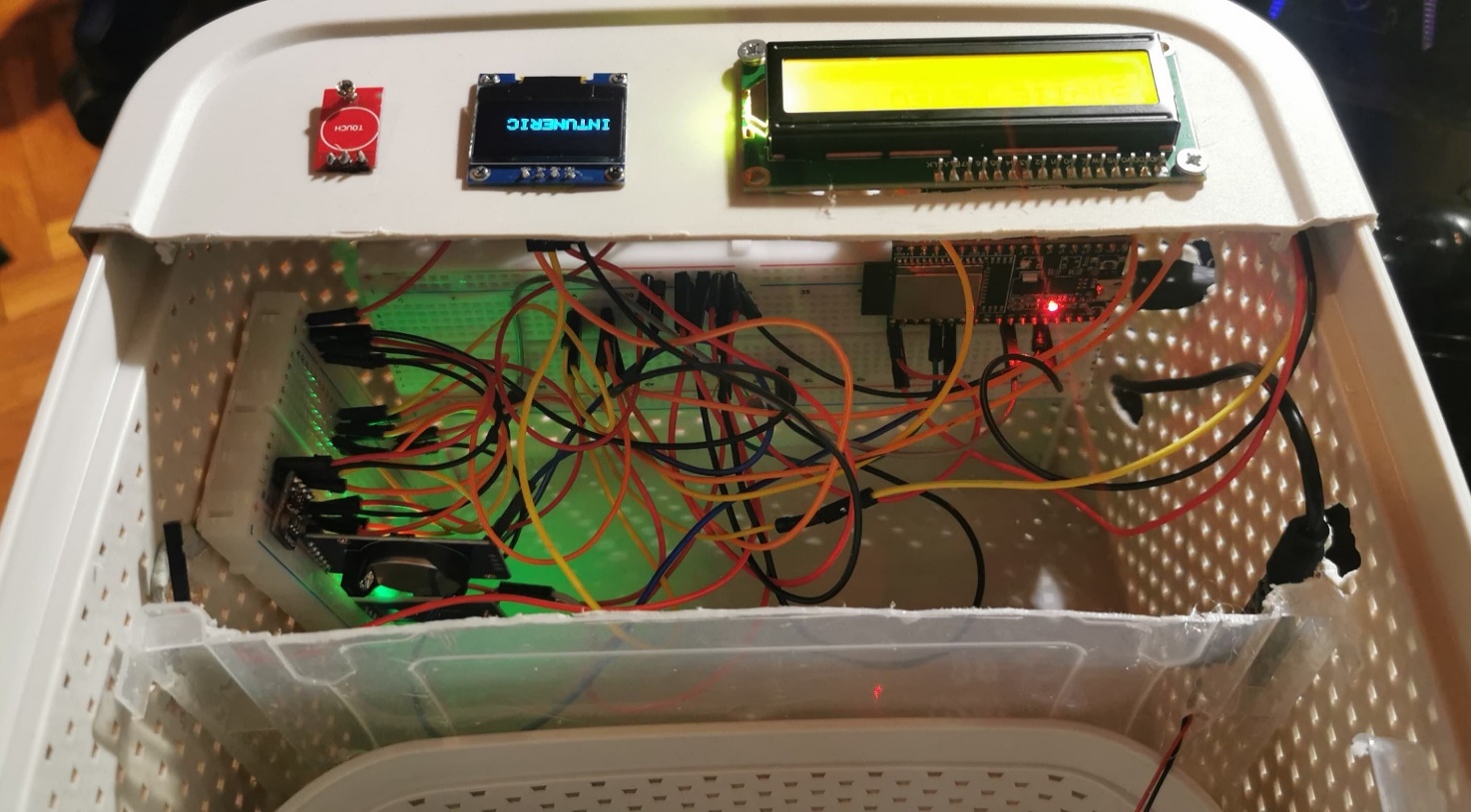
## 2.3.1 Integrarea într-un mediu de testare cu simulare de ploaie

În această etapă a proiectului s-a urmărit integrarea completă a componentelor hardware într-un ansamblu funcțional, protejat și adaptat pentru testarea în condiții cât mai apropiate de cele reale. S-a avut în vedere atât partea de montaj electronic, cât și amenajarea unui spațiu dedicat pentru simularea fenomenului de ploaie, astfel încât întregul sistem să poată fi evaluat într-un mediu controlat.

După realizarea testelor individuale, toate componentele au fost montate într-o carcasă, împărțite pe două plăcuțe breadboard distincte, pentru a facilita organizarea firelor și a reduce riscul de interferențe sau erori de conexiune.

Prima plăcuță breadboard conține placa de dezvoltare ESP32, împreună cu toate conexiunile către afișajul OLED, afișajul LCD și butonul tactil. Aceste elemente, vizibile în partea superioară a carcasei, reprezintă interfața locală prin care utilizatorul poate vizualiza valorile înregistrate și poate comuta între paginile afișate sau simula starea de ploaie. În figura 25 este prezentat montajul fizic al acestei părți a sistemului, cu evidențierea separării clare între plăcuța cu ESP32 și afișaje, respectiv cea dedicată senzorilor.

A doua plăcuță breadboard, amplasată în partea opusă, conține senzorii de mediu senzorul BME280, senzorul de lumină (LDR) și senzorul de ploaie, dar și RTC-ul. Această împărțire permite o conectare ușoară și o testare mai eficientă. În **figura 22** este ilustrat montajul fizic complet al componentelor, cu evidențierea separării clare pe cele două plăcuțe breadboard și a poziționării afișajelor și a butonului tactil.



**Figura 22–Asamblarea într-un mediu de simulare-***sursă: fotografie personală*

Pentru a putea testa în mod controlat reacția senzorului de ploaie și pentru a proteja componentele electronice de umezeală, interiorul carcasei a fost împărțit în două zone distincte. O zonă este dedicată montajului electronic, iar cealaltă a fost amenajată special pentru simularea condițiilor de ploaie.

În această a doua zonă a fost montat un mini-rezervor de apă, suspendat de peretele carcasei, în care a fost fixat umidificatorul cu ultrasunete. Prin activarea controlată a acestuia, se generează vapori de apă sau picături care ajung direct pe plăcuța de detecție a senzorului de ploaie, simulând astfel apariția fenomenului de precipitații. Plăcuța senzorului de ploaie a fost poziționată oblic, astfel încât apa să se poată scurge ușor de pe suprafața acesteia după oprirea umidificatorului.

Pentru a împiedica umezeala să ajungă la partea electronică a sistemului, cele două zone din interiorul carcasei sunt separate printr-un panou de protecție transparent, care delimitează clar spațiul de testare al senzorului de ploaie. În **figura 23** este ilustrată această zonă a carcasei, cu evidențierea rezervorului de apă, a poziționării umidificatorului și a senzorului de ploaie.

**Figura 23– Asamblarea într-un mediu de simulare -***sursă: fotografie personală*

# 2.4. PROGRAMUL PRINCIPAL-FUNCȚIONALITATEA PROGRAMULUI

Programarea componentelor din cadrul stației meteo a fost realizată în limbajul **C/C++**, limbajul de bază utilizat în platforma **Arduino IDE 2.3.6 *[16]***. Alegerea acestui limbaj a venit în mod natural, întrucât oferă un echilibru foarte bun între controlul detaliat al hardware-ului și ușurința de scriere a codului pentru microcontrolere.

Am avut ocazia să mă familiarizez cu structura programelor în C++ încă de la începutul proiectului, iar experiența dobândită anterior în lucrul cu structuri logice, funcții, variabile globale și biblioteci externe s-a dovedit extrem de utilă. Programarea în acest limbaj m-a ajutat să înțeleg mai bine modul în care funcționează componentele electronice la nivel de registru, semnal și ciclu de execuție.

Din punct de vedere personal, utilizarea C++-ului în acest proiect a reprezentat un progres practic semnificativ. Am învățat cum să gestionez condiții multiple, să structurez codul pentru afișaje secvențiale și să citesc în mod eficient valorile analogice și digitale de la senzori. De asemenea, am înțeles importanța organizării codului în secțiuni clare (setup, loop), dar și avantajul reutilizării funcțiilor pentru un cod curat și eficient.

Acest proiect mi-a oferit o ocazie excelentă de a exersa scrierea de cod în context hardware real, iar limbajul C++ s-a dovedit a fi o alegere potrivită și accesibilă pentru integrarea cu ESP32 și bibliotecile externe necesare ***[4]***. **Tabelul 12** prezintă avantajele și dezavantajele acestei platforme.

Tabelul 12 – Avantaje și dezavantaje ale platformei Arduino IDE [16]

|  |  |
| --- | --- |
| Avantaje | Dezavantaje |
| Interfață simplă și intuitivă, potrivită pentru începători | Nu oferă funcții avansate de depanare (debug) |
| Suport extins pentru plăci precum ESP32, Arduino UNO etc. | Compilarea poate fi lentă la proiecte mari |
| Comunitate mare, cu numeroase exemple și biblioteci disponibile | Organizarea fișierelor nu este modulară sau profesională, ca în alte IDE-uri |
| Integrare rapidă cu plăci prin USB și încărcare directă | Editorul de cod este simplu, fără completare automată inteligentă avansată |
| Ușor de instalat și configurat pe orice sistem de operare | Lipsa integrării native cu Git sau gestionare avansată a versiunilor |
| Se pot adăuga rapid biblioteci pentru senzori și module | Probleme ocazionale de compatibilitate cu plăci noi, fără suport oficial complet |

Limbajul de programare folosit în cadrul acestui proiect a fost **C++**, limbaj cu care am avut o expunere constantă și aprofundată pe parcursul facultății, în cadrul unor materii precum *Programarea Calculatoarelor*, *Tehnici de Programare*, *Metode și Tehnici de Programare (MTP)* și altele. Aceste cursuri mi-au oferit o bază solidă în înțelegerea structurilor de control, a variabilelor, funcțiilor, pointerilor și a modului în care pot fi gestionate resursele la nivel hardware.

C++ este un limbaj care oferă un echilibru foarte bun între nivelul de control asupra componentelor hardware și claritatea logicii de programare. Structura sa clară, comenzile intuitive și sintaxa relativ prietenoasă îl fac potrivit nu doar pentru proiecte embedded, ci și pentru aplicații complexe dezvoltate la nivel industrial.

Limbajul este larg utilizat în companii din domenii precum automotive, robotică, automatizări industriale, aplicații cu microcontrolere și dezvoltare software de sistem. De asemenea, platforme populare precum **Arduino IDE** sau **PlatformIO** sunt bazate pe compilatoare C/C++, ceea ce îl face limbajul standard pentru programarea plăcilor de dezvoltare precum **ESP32**.

**Realizarea Codului Sursă**

Pentru început, am stabilit și instalat bibliotecile necesare funcționării componentelor hardware utilizate în cadrul proiectului. Aceste biblioteci au fost alese în urma etapelor anterioare de testare individuală a fiecărui modul: ecranul OLED, modulul RTC, senzorul de temperatură/umiditate/presiune BME280, ecranul LCD, senzorul de ploaie și cel de lumină. De asemenea, am definit obiectele asociate fiecărei componente și am realizat maparea pinilor de pe placa ESP32 în variabile constante, pentru a permite o structurare mai clară și un control facil asupra întregului montaj ***[16]***.

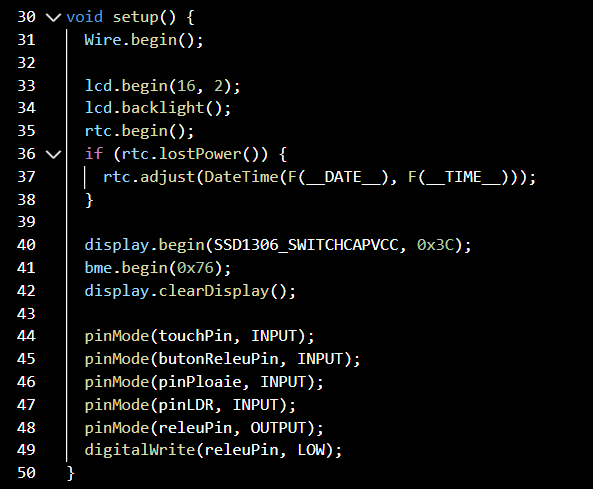
După cum se poate vedea în secvența de cod **4.1** de mai jos, sunt inițializate și câteva variabile auxiliare folosite pentru logica afișajului și controlul interacțiunii.



**Secvența de cod 4.1**– **Codul final stație meteo-***sursă: proiect Arduino IDE*

După ce am definit bibliotecile și variabilele, în funcția *setup()* se realizează inițializarea efectivă a componentelor: pornirea comunicației seriale, activarea afișajelor și a senzorilor, precum și stabilirea modului de lucru al pinilor digitali (INPUT/OUTPUT). Această etapă este esențială pentru pregătirea sistemului înainte de rularea buclei principale *loop().*

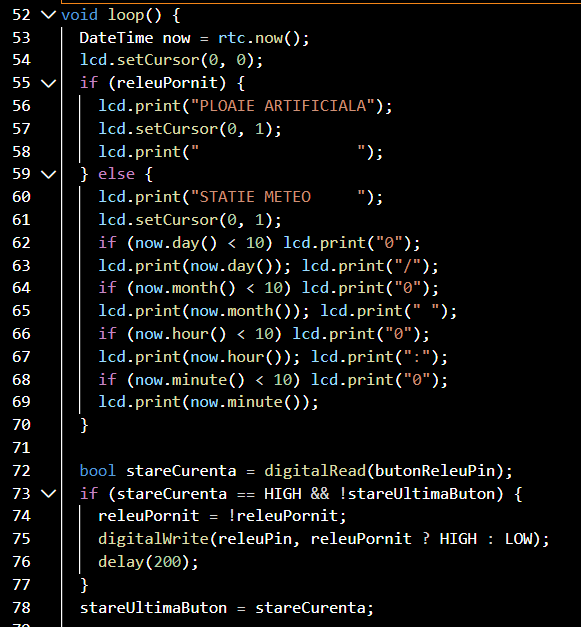
Astfel, sunt inițializate interfețele de comunicare I2C (folosite pentru OLED, RTC și BME280), se pornesc afișajele, se verifică starea modulului RTC (în caz că a pierdut alimentarea), se inițializează senzorul de mediu BME280 și se configurează toți pinii de intrare/ieșire necesari pentru funcționarea butoanelor, senzorilor și releului. După cum se poate observa în **secvența 4.2**, toate aceste operațiuni sunt grupate și clar structurate:



**Secvența de cod 4.2**– **Codul final stație meteo, funcția setup-***sursă: proiect Arduino IDE*

În cadrul funcției loop(), este tratată inițial actualizarea afișajului LCD cu informații esențiale pentru utilizator. Se citește ora curentă de la modulul RTC și se plasează cursorul LCD pe prima linie. Dacă releul care comandă umidificatorul este activ, pe ecran se afișează mesajul „PLOAIE ARTIFICIALĂ”, oferind un indiciu vizual clar că ploaia artificială este în desfășurare. În cazul în care releul este inactiv, pe primul rând apare textul fix „STATIE METEO”, urmat apoi, pe linia a doua, de data și ora actualizate în timp real. Afișajul este formatat astfel încât toate valorile să fie afișate în format complet, cu zerouri pentru valori mai mici de 10, pentru o lizibilitate crescută.

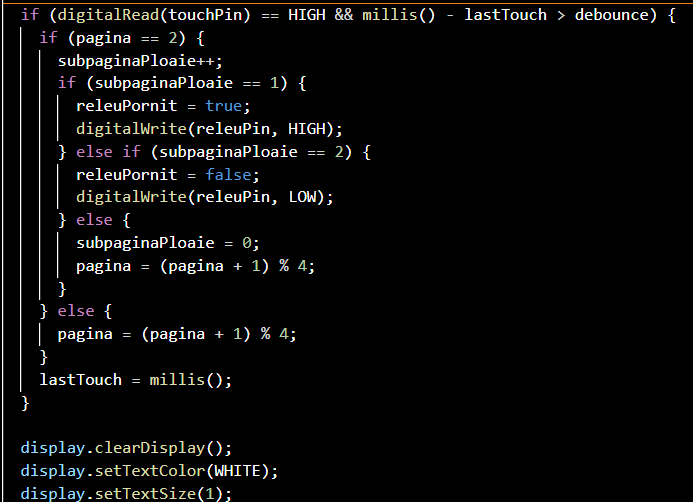
După această secțiune de afișare, codul tratează citirea stării unui buton fizic conectat la pinul digital 15. Atunci când acesta este apăsat, se inversează starea logică a variabilei releu Pornit, iar pinul care comandă releul se setează pe HIGH sau LOW, în funcție de această stare. Tot aici este implementat un sistem de verificare simplu, pentru a evita activări multiple cauzate de zgomotul electric generat la apăsare. Liniile de cod corespunzătoare acestei logici sunt prezentate în **secvența** **4.3** de mai jos.



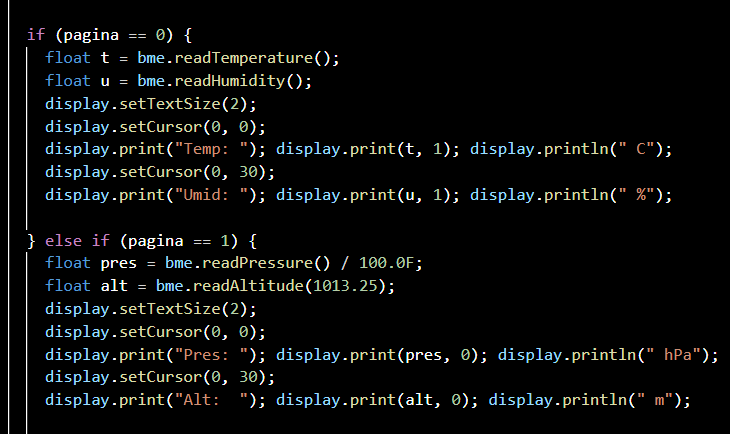
**Secvența de cod 4.3**– **Codul final stație meteo, funcția loop-***sursă: proiect Arduino IDE*

Mai departe, se trece la secțiunea care gestionează afișajul OLED. După ce s-au procesat și afișat datele pe LCD și s-a verificat starea releului, codul se ocupă de logica de comutare a paginilor afișate pe ecranul OLED, în funcție de apăsările detectate pe butonul tactil. La fiecare apăsare, variabila pagina este incrementată, iar dacă se ajunge la finalul setului de pagini definite, aceasta este resetată la zero pentru a relua ciclul. În funcție de valoarea actuală a variabilei pagina, sunt afișate diferite informații: pe prima pagină sunt afișate temperatura, umiditatea și presiunea atmosferică; pe a doua pagină, starea vremii conform senzorului de ploaie și a stării releului; iar pe pagina a treia, intensitatea luminii detectate de senzorul LDR este interpretată și afișată sub forma unor mesaje clare precum „LUMINĂ” sau „ÎNTUNERIC”.

Tot aici, se șterge ecranul OLED înainte de fiecare actualizare pentru a preveni suprapunerile vizuale, iar textele sunt poziționate cu *setCursor* pentru a asigura o afișare structurată și lizibilă. Liniile de cod care reflectă această logică sunt prezentate în **secvențele** **4.4 ,4.5 și 4.6** de mai jos.



**Secvența de cod 4.4**– **Codul final stație meteo-***sursă: proiect Arduino IDE*



**Secvența de cod 4.5**– **Codul final stație meteo-***sursă: proiect Arduino IDE*



**Secvența de cod 4.6**– **Codul final stație meteo-***sursă: proiect Arduino IDE*

**Concluzie asupra funcționării codului:**

Prin implementarea acestui cod, sistemul stației meteo devine complet funcțional și interactiv. Afișajul **LCD** redă în mod constant numele proiectului și data/ora curente, extrase de la modulul RTC, iar în timpul activării umidificatorului, mesajul se schimbă automat pentru a indica ploaia artificială.

Afișajul **OLED** permite utilizatorului să navigheze prin mai multe pagini, folosind un buton tactil: temperatura și umiditatea, presiunea atmosferică și altitudinea, starea vremii bazată pe senzorul de ploaie și starea luminii ambientale citită printr-un LDR. În pagina dedicată ploii, se poate porni și opri artificial ploaia printr-un al doilea buton fizic, care comandă releul ce alimentează umidificatorul.

Fiecare secțiune a codului a fost construită modular, pentru a asigura claritate, funcționalitate stabilă și o interfață intuitivă pentru utilizator. Valorile de la senzori sunt afișate în timp real, iar stările componente sunt reflectate vizual, într-un mod simplu și eficient.

# 2.5. AFIȘAREA PARAMETRILOR ÎNTR-O INTERFAȚĂ GRAFICĂ

Unul dintre cele mai importante elemente ale proiectului este interfața grafică, creată special pentru a facilita afișarea și interpretarea valorilor colectate de senzorii stației meteo. Aceasta are rolul de a oferi utilizatorului o imagine de ansamblu asupra condițiilor de mediu, într-un mod vizual, clar și ușor de urmărit chiar și de la distanță.

Prin intermediul interfeței, valorile pot fi comparate, analizate și interpretate mai rapid, fiind astfel posibilă luarea unor decizii corecte, cum ar fi activarea udării artificiale în caz de secetă sau evitarea unor intervenții în condiții de umiditate ridicată.

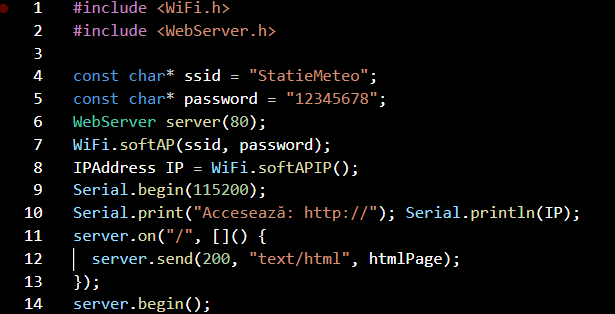
Pentru o prezentare cât mai clară a modului în care a fost realizată această interfață și a funcționalităților incluse, acest capitol este împărțit în două părți:

**Crearea interfeței grafice** – unde este prezentat procesul de realizare a paginii web și modul în care aceasta este accesată;

**Afișarea parametrilor în interfață** – unde sunt descrise componentele vizuale, interacțiunile posibile și modul în care datele sunt prezentate în timp real.

## 2.5.1Crearea interfeței grafice

Pentru a putea vizualiza în timp real valorile senzorilor, s-a dorit dezvoltarea unei interfețe accesibile prin rețea, fără a fi nevoie de conectarea la internet. Astfel, una dintre primele modificări aduse codului sursă a fost activarea modulului Wi-Fi integrat în placa ESP32 și configurarea acestuia în mod **Access Point**. Această metodă permite crearea unei rețele Wi-Fi locale cu numele ales de utilizator, la care orice dispozitiv se poate conecta direct.

Pentru început, în codul sursă existent au fost adăugate bibliotecile necesare pentru funcționarea serverului web, iar ESP32 a fost setat să creeze o rețea proprie cu un nume (SSID) și o parolă definită. Modificările și inițializarea rețelei pot fi observate în **secvența** de cod **5.1** de mai jos.

**Secvența de cod 5.1**–**cod interfață-***sursă: proiect Arduino IDE*

Pentru a transmite în mod dinamic valorile senzorilor către interfața grafică, se definesc mai multe rute distincte în serverul web al ESP32. Fiecare rută este interogată periodic de pagina web folosind cod JavaScript, iar ESP32 răspunde cu datele preluate direct de la senzori în momentul cererii.

Astfel, pagina web rămâne mereu sincronizată cu valorile live măsurate, fără a necesita reîncărcare completă. Această abordare permite afișarea temperaturii, umidității, presiunii atmosferice, luminozității, stării ploii, datei și orei în timp real. **Secvența de cod 5.2** de mai jos evidențiază definirea acestor rute de tip API:

**Secvența de cod 5.2**–**cod interfață-conexiuni-***sursă: proiect Arduino IDE*

Rețeaua creată de placă are un nume predefinit, specificat în codul sursă, respectiv **„StatieMeteo”**. Pentru a permite accesul securizat, a fost definită o parolă implicită, **„12345678”**, tot în interiorul codului sursă.

Utilizatorul se poate conecta la această rețea folosind orice dispozitiv compatibil cu conexiuni wireless (telefon, tabletă, laptop). Odată conectat, accesarea interfeței grafice se face prin introducerea adresei IP **192.168.4.1** în bara de adrese a unui browser web.

După accesarea acestei adrese, serverul web implementat pe placa ESP32 va răspunde prin trimiterea paginii HTML stocate în fișierul ***htmlpage.h***, afișând astfel interfața grafică a stației meteo. Această interfață permite vizualizarea datelor transmise de senzori, salvarea acestora într-un tabel, afișarea grafică în timp real a temperaturii și umidității, precum și generarea de recomandări bazate pe valorile înregistrate.

## 2.5.2 Afișarea parametrilor în interfață

Codul HTML care compune interfața grafică este stocat într-o variabilă de tip ***const char[]*** denumită ***htmlPage***, aflată în fișierul htmlpage.h. Acest cod este transmis browserului utilizatorului în momentul accesării adresei IP locale generate de ESP32.

Scopul principal al acestei interfețe este de a furniza o prezentare clară, în timp real, a datelor furnizate de senzori. Prin intermediul elementelor vizuale precum afișaje dinamice, tabele de înregistrări, grafice și pictograme sugestive, utilizatorul poate interpreta rapid starea mediului și poate lua decizii (ex: activarea ploii artificiale, analiza umidității etc.). Interfața aduce beneficii importante precum:

Accesibilitate rapidă de pe orice dispozitiv cu Wi-Fi

Vizualizarea în timp real a valorilor măsurate

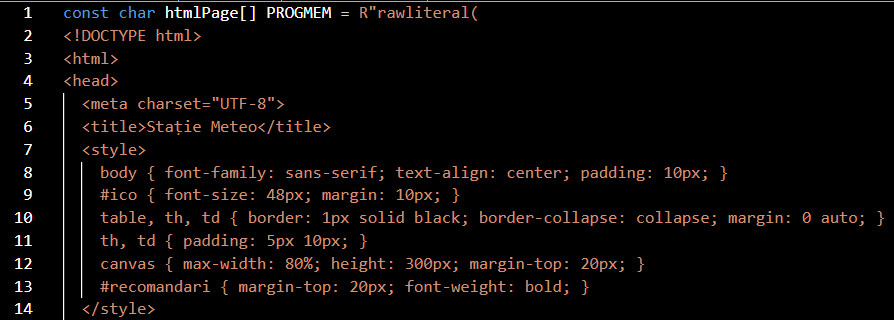
Înregistrarea datelor la cerere într-un tabel

Generarea de recomandări bazate pe valori anormale

În prima parte a codului HTML sunt definite structura de bază ***(<!DOCTYPE html>, <html>, <head>)*** și stilul paginii printr-un bloc de **CSS**. Aceste elemente stabilesc aspectul vizual general al aplicației: fontul, culorile, aspectul butoanelor și al graficelor, poziționarea componentelor pe ecran ***[21]***.

CSS-ul personalizat este integrat direct în interiorul etichetei **<style>**, eliminând nevoia de fișiere externe și simplificând astfel distribuirea interfeței prin ESP32.

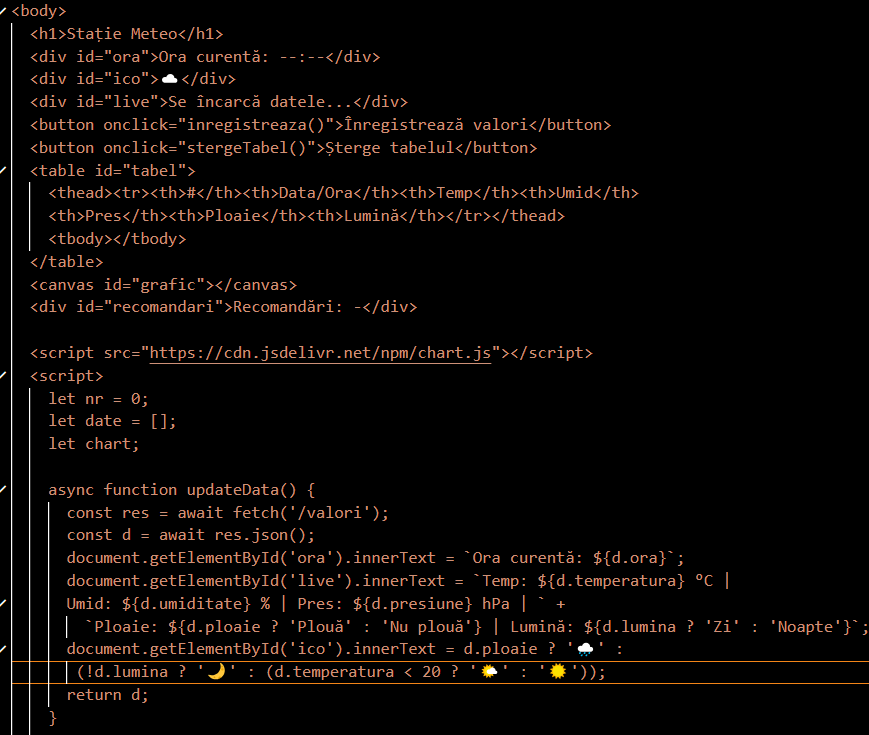
Această secțiune se încheie cu eticheta ***</style>*** și poate fi extrasă pentru documentație de la începutul fișierului HTML până la linia care conține ***</style>.***  
Exemplu delimitare: Secvența de cod 6.1 – Structura generală și stilul interfeței



**Secvența de cod 6.1 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

După definirea stilurilor, urmează construirea structurii funcționale a interfeței, prezentată în **secvența de cod 6.2**. Aceasta începe cu afișarea titlului **Stație Meteo** și a unui element pentru data și ora curentă – evidențiat prin ***<div id="ora">***. Apoi, prin ***<div id="ico">*** se afișează o **pictogramă meteo** care se schimbă automat în funcție de starea vremii (soare, nor, lună etc.).

Este inclusă o zonă live – ***<div id="live">***, unde sunt afișate datele actuale preluate de la senzori. Sub aceasta, două butoane – ***inregistreaza()*** și ***stergeTabel()***– permit salvarea sau ștergerea înregistrărilor din tabelul cu date meteo, generat dinamic în ***<table id="tabel">***. Urmează un grafic combinat pentru temperatură și umiditate în elementul ***<canvas id="grafic">*** și o zonă de recomandări în funcție de valorile măsurate.

;LNK;NNPIPINNIPKActualizarea dinamică a acestor elemente este realizată prin funcția asincronă ***updateData()***, care aduce datele de la ESP32 și actualizează atât valorile afișate, cât și pictograma în timp real.

**Secvența de cod 6.2 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

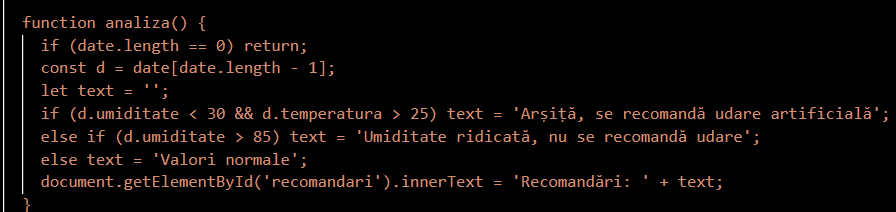
În continuarea structurii HTML, interfața include două funcționalități esențiale pentru interacțiunea utilizatorului, implementate prin funcțiile ***stergeTabel()*** și ***inregistreaza()***, prezentate în **secvența de cod 6.3** de mai jos.

Funcția ***stergeTabel()*** este responsabilă cu resetarea completă a tabelului, a graficului și a zonei de recomandări. Aceasta elimină toate rândurile din tabel – prin ***innerHTML = ''***, resetează valorile stocate și reinițializează graficul, golind datele din axele X și Y. Practic, se revine la starea inițială a interfeței, fără înregistrări vizuale sau analitice.

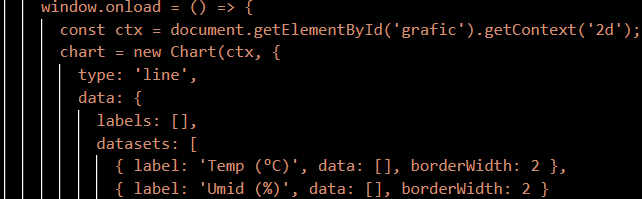
Funcția ***inregistreaza()*** este activată la apăsarea butonului de salvare. Ea folosește funcția asincronă ***updateData()*** pentru a prelua cele mai recente valori de la ESP32. Ulterior, dacă nu s-a atins limita maximă de 7 înregistrări, valorile sunt stocate într-un obiect ***date[]***, iar un rând HTML nou este generat în tabel cu informații precum numărul înregistrării, data și ora, temperatura, umiditatea, presiunea, starea ploii și nivelul de lumină.

Simultan, graficul este actualizat adăugând noile date de temperatură și umiditate, prin ***chart.data.datasets[...]***, iar la final este apelată funcția ***analiza()***, responsabilă cu interpretarea valorilor înregistrate. Această logică adaugă funcționalitate practică interfeței, permițând atât vizualizarea istorică cât și reprezentarea grafică în timp real a condițiilor meteo.

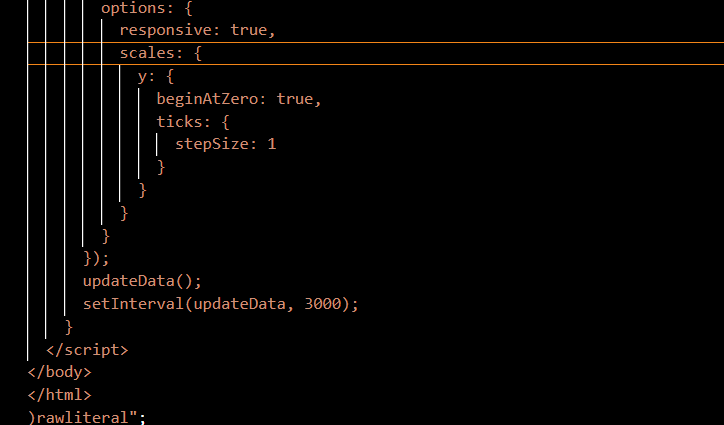
**Secvența de cod 6.3 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

Această ultimă secțiune de cod, ilustrată în **secvența 6.4**, finalizează logica interfeței grafice printr-o componentă de analiză și interpretare automată a valorilor înregistrate. Funcția ***analiza()*** este declanșată la fiecare nouă înregistrare și se ocupă cu generarea unui mesaj sugestiv în zona de recomandări, adaptat în funcție de condițiile meteo detectate. Pentru a stabili mesajul, funcția preia ultima înregistrare din lista de date și evaluează valorile de umiditate și temperatură. Dacă se înregistrează o umiditate scăzută (sub 30%) combinată cu o temperatură ridicată (peste 25°C), se consideră că este arșiță, iar utilizatorului i se recomandă udarea artificială. În schimb, dacă umiditatea este prea mare (peste 85%), se consideră excesivă și sistemul afișează o sugestie de evitare a udării. În celelalte cazuri, este returnat mesajul "Valori normale", indicând o stare echilibrată a mediului.

**Secvența de cod 6.4 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

În încheierea componentei HTML, este inclusă o funcționalitate esențială pentru vizualizarea intuitivă a parametrilor monitorizați de senzorii stației meteo: afișarea datelor într-un grafic de tip linie. Inițializarea acestui grafic se face în momentul încărcării paginii, folosind metoda **window.onload**, iar așezarea vizuală se realizează cu ajutorul bibliotecii externe ***Chart.js***. În cadrul funcției, este obținut contextul 2D al elementului ***<canvas id="grafic">****,* iar ulterior este creat un obiect de tip **Chart**, având două seturi de date (**temperature** și **umiditate**). Acestea sunt reprezentate prin linii distincte și actualizate dinamic pe măsură ce utilizatorul înregistrează noi măsurători. S-a specificat ca axa **Y** să pornească de la zero și să aibă un **pas de 1 unitate**, oferind o reprezentare clară și detaliată a variației valorilor înregistrate. Pentru a menține informațiile afișate cât mai recente, se apelează periodic funcția **updateData()**, cu o frecvență de **3 secunde**, lucru evidențiat în **secvențele 6.5 și 6.6** de mai jos.

**Secvența de cod 6.5 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

**Secvența de cod 6.6 – Structura generală și stilul interfeței-** *sursă: proiect Arduino IDE*

**Afișarea valorilor în interfață:**

În prima secțiune a interfeței web, valorile colectate de la senzori sunt afișate în timp real pentru a oferi utilizatorului o imagine clară și rapidă asupra stării mediului înconjurător. Acestea includ temperatura, umiditatea relativă, presiunea atmosferică, starea de ploaie și nivelul de lumină. Informațiile sunt actualizate automat și prezentate într-un mod organizat, alături de ora curentă preluată de la modulul RTC.

Un element vizual suplimentar, extrem de util, este pictograma meteo din centrul interfeței, care se modifică dinamic în funcție de parametrii detectați. Astfel, utilizatorul poate înțelege imediat dacă este zi sau noapte, dacă plouă sau dacă vremea este senină. Acest mod de afișare facilitează luarea deciziilor rapide, chiar și de la distanță.

Aspectul acestei secțiuni este ilustrat în **Figura 24**, unde poate fi observat un exemplu concret al modului în care sunt prezentate aceste informații în interfață.

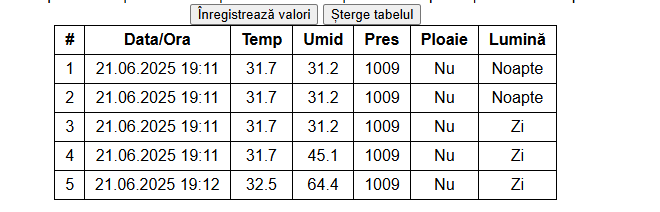


**Figura 24–Interfață STAȚIE METEO -***sursă: fotografie personală*

Următoarea componentă a interfeței grafice este reprezentată de tabelul de înregistrare a valorilor. La apăsarea butonului **„Înregistrează valori”**, datele provenite de la senzori sunt salvate în mod static într-un tabel HTML, împreună cu data și ora exactă a momentului înregistrării. Această funcționalitate oferă o evidență clară și structurată a variațiilor de mediu pe parcursul timpului.

Tabelul este gândit să păstreze un număr limitat de linii (maxim 7), astfel încât afișarea să rămână clară și ușor de urmărit. Fiecare rând conține o înregistrare completă ce include temperatura, umiditatea, presiunea, starea de ploaie și nivelul de lumină, iar utilizatorul are și posibilitatea de a șterge complet istoricul prin intermediul butonului **„Șterge tabelul”**.

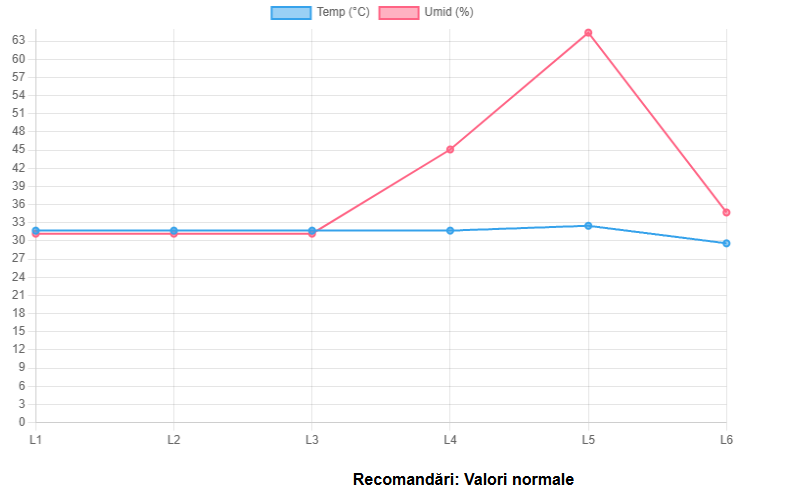
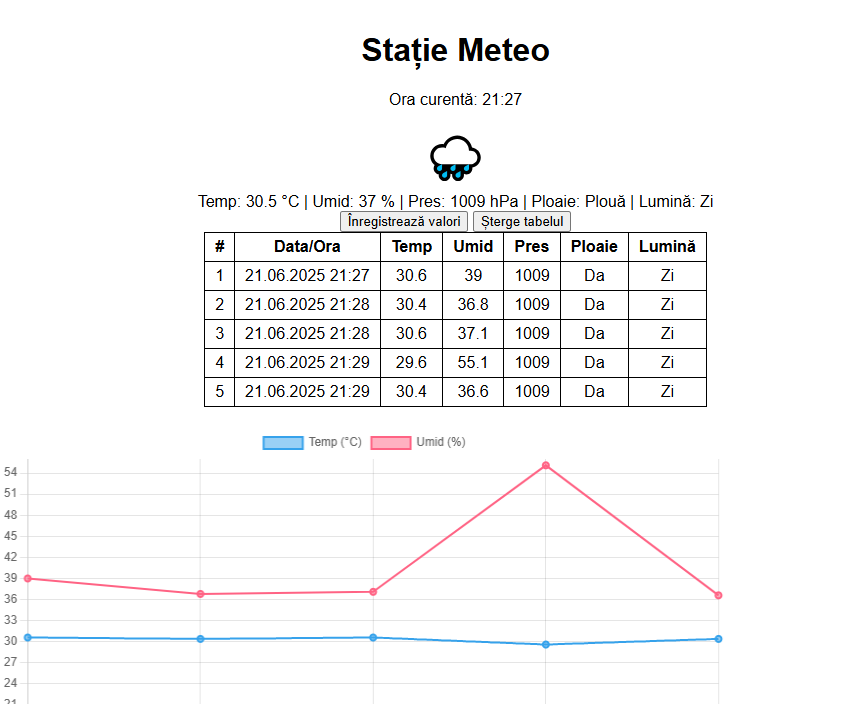
Această funcționalitate este ilustrată în **Figura 25**, unde se observă mai multe înregistrări succesive efectuate într-un interval scurt, demonstrând utilitatea și claritatea modului de prezentare a datelor în tabel.

**Figura 25–Interfață STAȚIE METEO -***sursă: fotografie personală*

În continuarea interfeței, datele înregistrate în tabel sunt transpuse automat într-un grafic combinat ce afișează evoluția temperaturii și a umidității în funcție de înregistrările efectuate. Fiecare linie salvată este reprezentată în grafic ca un punct distinct, cu o culoare specifică (albastru pentru temperatură și roz pentru umiditate), facilitând astfel analiza vizuală a variațiilor de mediu.

Această reprezentare grafică permite utilizatorului să observe rapid creșteri sau scăderi bruște ale valorilor înregistrate și să tragă concluzii cu privire la tendințele generale ale microclimatului monitorizat. Totodată, sub grafic este afișată o zonă dedicată recomandărilor inteligente, generate pe baza ultimei înregistrări. În funcție de combinația valorilor temperatură-umiditate, se oferă sugestii precum *„Arșiță, se recomandă udare artificială”* sau *„Umiditate ridicată, nu se recomandă udare”*.

Toate aceste funcționalități sunt vizibile în **Figura 26**, care exemplifică modul în care valorile din tabel sunt reflectate în timp real în grafic și în secțiunea de recomandări.

**Figura 26.1–Interfață STAȚIE METEO -***sursă: fotografie personală*

**Figura 26.2–Interfață STAȚIE METEO -***sursă: fotografie personală*

# 3. CONCLUZII ȘI IMPRESII

În cadrul acestei lucrări de licență a fost proiectată și realizată o stație meteo cu funcționalitate extinsă, având la bază o arhitectură hardware și software dedicată monitorizării parametrilor de mediu. Pentru atingerea obiectivului propus, au fost parcurse etape clare, începând cu selecția și caracterizarea componentelor electronice, continuând cu realizarea montajului fizic, testarea individuală a senzorilor, dezvoltarea programului principal și finalizând cu implementarea unei interfețe grafice web pentru afișarea și interpretarea datelor în timp real.

## PAȘII DE REALIZARE A PROIECTULUI

Realizarea acestui proiect a presupus mai mulți pași, urmărind o ordine firească de lucru, astfel încât fiecare etapă să contribuie la funcționarea corectă a ansamblului final. Totul a început cu alegerea temei, justificată prin utilitatea practică a unei stații meteo personale și prin oportunitatea de a lucra cu platforma ESP32, apreciată pentru flexibilitatea și resursele sale. A urmat o perioadă de documentare pe partea de componente electronice și soluții tehnice disponibile, etapă care m-a ajutat să stabilesc clar ce senzori și module sunt potrivite pentru proiect. După ce lista de componente a fost definitivată, acestea au fost achiziționate, iar montajul fizic a putut fi realizat pe breadboard, respectând schema, simulată în platforma Wokwi.

Pentru a apropia proiectul cât mai mult de condițiile reale de utilizare, ansamblul a fost montat într-o carcasă, astfel încât să fie protejat și să poată simula mai ușor diverse medii de funcționare. Pe baza aplicației Arduino IDE s-a dezvoltat codul necesar și s-a creat o interfață web simplă și intuitivă, prin care valorile înregistrate de senzori pot fi vizualizate în timp real.

La finalul proiectului a fost obținut un sistem complet funcțional, care îmbină partea hardware cu partea software, într-o formă practică și ușor de utilizat. Montajul electronic a fost realizat în carcasă, astfel încât toate componentele să fie protejate și să se poată simula condiții cât mai apropiate de cele reale.

Stația meteo permite nu doar monitorizarea valorilor în timp real, ci și vizualizarea lor local pe afișaje, precum și accesul la date prin intermediul interfeței web. În plus, sistemul are capacitatea de a simula prezența apei, prin controlul unui umidificator, oferind astfel o metodă simplă, dar eficientă de a testa reacția senzorilor în diverse situații. Toate aceste funcționalități contribuie la un ansamblu complet, care poate fi folosit atât pentru monitorizare, cât și în scop educativ sau demonstrativ.

## 3.2 AVANTAJE ȘI DEZAVANTAJE

\*"Stația meteo realizată în cadrul acestei lucrări prezintă o serie de avantaje care o diferențiază față de alte soluții similare, fie ele comerciale sau educaționale.

Un prim aspect care merită menționat este integrarea completă a componentelor, de la senzori și afișaje până la interfața web și simularea ploii, totul fiind reunit într-un sistem compact și ușor de utilizat. Spre deosebire de proiectele educaționale de bază, cum sunt cele realizate cu platforma Arduino UNO ***[20]***, proiectul prezentat oferă funcționalități suplimentare, cum ar fi accesul wireless la date și posibilitatea de a simula condiții meteo prin activarea umidificatorului.

Un alt avantaj important este utilizarea microcontrolerului ESP32, care, spre deosebire de soluții precum Arduino UNO sau sisteme comerciale dedicate, oferă mai multe resurse hardware și conectivitate Wi-Fi integrată ***[4], [19]***. Astfel, proiectul poate fi extins ușor cu funcții suplimentare sau conectat la o rețea IoT, așa cum este descris în literatura de specialitate ***[18]***.

Totuși, sistemul prezintă și câteva limitări. Fiind un prototip, carcasa și modul de fixare al componentelor pot fi îmbunătățite, pentru o rezistență sporită în mediile exterioare. De asemenea, alimentarea sistemului se face prin cablu USB, ceea ce limitează portabilitatea. În plus, interfața web este simplă și funcțională, dar poate fi extinsă cu opțiuni suplimentare de stocare sau vizualizare a datelor pe termen lung.

Ca direcții de dezvoltare, proiectul poate fi îmbunătățit prin integrarea unui sistem de alimentare autonom, de exemplu pe bază de panouri solare, prin adăugarea unor senzori suplimentari (cum ar fi un senzor de vânt) sau prin conectarea la o platformă IoT completă, așa cum se practică în sistemele profesionale de monitorizare a mediului ***[19]***.

## 3.3 IMPRESII PERSONALE

Realizarea acestui proiect mi-a oferit ocazia să aplic concret cunoștințele acumulate în anii de studiu și să înțeleg mai bine provocările practice care apar în dezvoltarea unui sistem funcțional. Îmbinarea componentelor hardware cu partea de programare și cu dezvoltarea unei interfețe grafice m-a ajutat să văd întregul ansamblu ca pe un sistem complet, nu doar ca pe o colecție de module separate.

Deși proiectul poate fi considerat un prototip, sunt mulțumit de rezultatul obținut și consider că acest tip de abordare practică este cea mai bună metodă de a învăța și de a înțelege cu adevărat funcționarea sistemelor embedded. În același timp, am conștientizat importanța documentării corecte, a testării etapizate și a atenției la detalii, aspecte esențiale în orice proiect tehnic, indiferent de complexitate.

# BIBLIOGRAFIE

1. \*\*\* Stație meteo digitală Bresser cu afișaj color și senzori integrați, eMAG.ro, 2025, <https://www.emag.ro/statie-meteo-color-bresser/pd/DK6FF2MBM/>,(accesat 20.05.2025).
2. \*\*\* Moldova și Japonia vor realiza un proiect comun în domeniul meteorologiei, Agrobiznes.md, 2024, <https://agrobiznes.md/moldova-si-japonia-vor-realiza-un-proiect-comun-in-domeniul-meteorologiei.html>, (accesat 1.06.2025).
3. \*\*\* Stație meteo – Proiect PM 2021, Universitatea Politehnica București – OCW, 2021, <https://ocw.cs.pub.ro/courses/pm/prj2021/cghenea/statiemeteo>, (accesat 11.06.2025).
4. Espressif Systems, ESP32 Technical Reference Manual, 2024, <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/index.html>, (accesat 14.06.2025).
5. \*\*\* ESP32 Pinout Reference: Which GPIO Pins should you use?, Random Nerd Tutorials, 2024, <https://randomnerdtutorials.com/esp32-pinout-reference-gpios/>, (accesat 14.06.2025).
6. \*\*\* Kit breadboard 830 puncte cu modul de alimentare, eMAG.ro, 2025, <https://www.emag.ro/kit-breadboard-830-puncte-modul-alimentare-65-cabluri-jumper-arduino-uno-r3-mega-nano-raspberry-pi-pc0195/pd/DCL60KBBM/>, (accesat 12.06.2025).
7. \*\*\* Ecran OLED 0.96” I2C compatibil Arduino, eMAG.ro, 2025, <https://www.emag.ro/ecran-oled-0-96-ai409-s322-323-324/pd/D69S02MBM/>, (accesat 12.06.2025).
8. Adafruit Industries, Monochrome 0.96” 128x64 OLED Graphic Display, 2024, <https://learn.adafruit.com/monochrome-oled-breakouts>, (accesat 14.06.2025).
9. \*\*\* Modul BME280 5V pentru senzorul de temperatură, presiune și umiditate, eMAG.ro, 2025,<https://www.emag.ro/modul-bme280-5v-pentru-senzorul-de-temperatura-presiune-si-umiditate-arduino-5904501666077/pd/D13YFGYBM/>, (accesat 12.06.2025).
10. Bosch Sensortec, BME280 Combined Sensor Datasheet, 2024, <https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/>, (accesat 14.06.2025).
11. \*\*\* Rezistor sensibil la lumină (LDR 5 mm), eMAG.ro, 2025, <https://www.emag.ro/rezistor-sensibil-la-lumina-ldr-model-fotorezistor-5-mm-l141-ldr5/pd/D4SYM8YBM/>, (accesat 12.06.2025).
12. \*\*\* Modul senzor de ploaie compatibil Arduino, eMAG.ro, 2025, https://www.emag.ro/senzor-ploaie-arduino/pd/DRPLMBM/, (accesat 25.06.2025).
13. \*\*\* Modul umidificator cu ultrasunete 20 mm, Bitmi.ro, 2025, <https://www.bitmi.ro/module-electronice/modul-umidificator-cu-ultrasunete-si-micro-usb-20mm-11246.html>, (accesat 14.06.2025).
14. \*\*\* 20 mm Ultrasonic Humidifier Datasheet, DiyMore.cc, 2025, <https://www.diymore.cc/products/20mm-113khz-ultrasonic-humidifier-mist-maker-usb-ceramic-atomizer-transducer-humidified-plate-accessories-pcb-module-d20mm>, (accesat 14.06.2025).
15. Maxim Integrated, DS3231 Extremely Accurate RTC Datasheet, 2024, <https://datasheets.maximintegrated.com/en/ds/DS3231.pdf>, (accesat 14.06.2025).
16. \*\*\* Arduino IDE - Software open-source pentru dezvoltare embedded, Arduino.cc, 2025, https://www.arduino.cc/en/software, (accesat 25.06.2025).
17. McMillan, S. G., Introduction to Embedded Systems, Springer, 2020.
18. Davies, J., The Internet of Things: A Comprehensive Overview, Wiley, 2021.
19. \*\*\* Design and Implementation of an IoT-Based Weather Monitoring System\*, IEEE Xplore, 2023, <https://ieeexplore.ieee.org/document/10012345>, (accesat 14.06.2025).
20. \*\*\* Sistem de măsură a parametrilor dintr-o stație meteo folosind platforma Arduino Uno\*, 2022.
21. J. Duckett, *HTML and CSS: Design and Build Websites*, Wiley, 2011.  
    (*lucrare de bază despre structurarea și stilizarea interfețelor web*)

