

Sistem inteligent de Conservarea a vinurilor în crame

Introducere

În ultimii ani, tehnologia a început să joace un rol din ce în ce mai important în domeniul viticulturii și al producției de vinuri, ajutând producătorii să optimizeze procesul de vinificație și să îmbunătățească calitatea produselor. O aplicație inovatoare ce utilizează senzori de temperatură, umiditate și presiune poate transforma modul în care sunt monitorizate condițiile de depozitare a vinurilor, în special în cramele și vinăriile care necesită un control strict al mediului pentru a asigura o calitate superioară a vinului. În acest context, integrarea unui sistem de monitorizare automată cu alerte poate preveni riscurile asociate cu condițiile de mediu nefavorabile, protejând astfel vinul de deteriorări și optimizând procesul de maturare.



Descrierea Funcționalității:

Sistemul propus pentru monitorizarea condițiilor de depozitare a vinului utilizează senzori de temperatură, umiditate și presiune amplasați strategic în cramele sau încăperile de depozitare a vinurilor. Acești senzori sunt conectați la o rețea wireless, care permite colectarea continuă a datelor de mediu și transmiterea acestora către o aplicație centralizată.

1. **Senzorul de temperatură** măsoară constant temperatura din interiorul cramei sau al depozitului de vinuri, având în vedere că temperatura are un impact direct asupra procesului de fermentație și maturare a vinului. Dacă temperatura depășește un prag critic (de exemplu, 25°C), sistemul va emite o alarmă pentru a preveni deteriorarea vinului.
2. **Senzorul de umiditate** monitorizează nivelul de umiditate din încăpăre, având în vedere că un nivel prea scăzut sau prea ridicat poate afecta calitatea vinului, provocând evaporarea excesivă sau favorizând dezvoltarea mușgaiului. Dacă umiditatea depășește valorile recomandate, se va activa un sistem de alertă.
3. **Senzorul de presiune** este utilizat pentru a monitoriza fluctuațiile de presiune atmosferică care pot influența modul în care vinul se maturează și poate contribui la apariția unor defecte. Acesta va emite alerte atunci când se detectează schimbări rapide ale presiunii atmosferice.

Aceste date sunt transmise în timp real către o aplicație software dedicată, care le procesează și le vizualizează pe un dashboard intuitiv, folosind grafice și diagrame pentru a permite utilizatorilor să monitorizeze condițiile ambientale. În plus, aplicația va trimite alerte prin e-mail sau notificări mobile atunci când condițiile devin nesigure pentru vinuri, permitând astfel intervenția rapidă.



Beneficii:

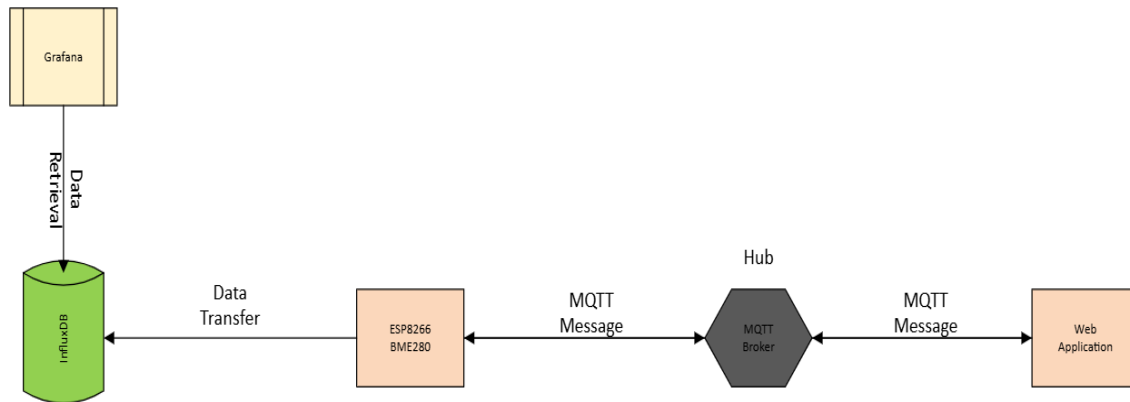
Monitorizarea constantă a temperaturii, umidității și presiunii asigură **control precis al mediului**, ajutând la menținerea unui mediu optim pentru depozitarea vinurilor, asigurându-se că acestea sunt protejate de condiții care ar putea afecta negativ calitatea lor. De asemenea, **prevenirea deteriorării vinului** prin detectarea rapidă a abaterilor de la parametrii optimi, sistemul poate preveni riscurile de deteriorare a vinului, cum ar fi oxidarea sau apariția unor mirosuri neplăcute, care ar putea compromite calitatea produsului final. Nu în ultimul rând, datele colectate pot fi stocate și analizate pentru a crea rapoarte detaliate privind condițiile de mediu din crama, oferind o **transparență** mai mare și o bază solidă de **raportare** pentru deciziile de îmbunătățire a proceselor de producție.

Arhitectura Sistemului

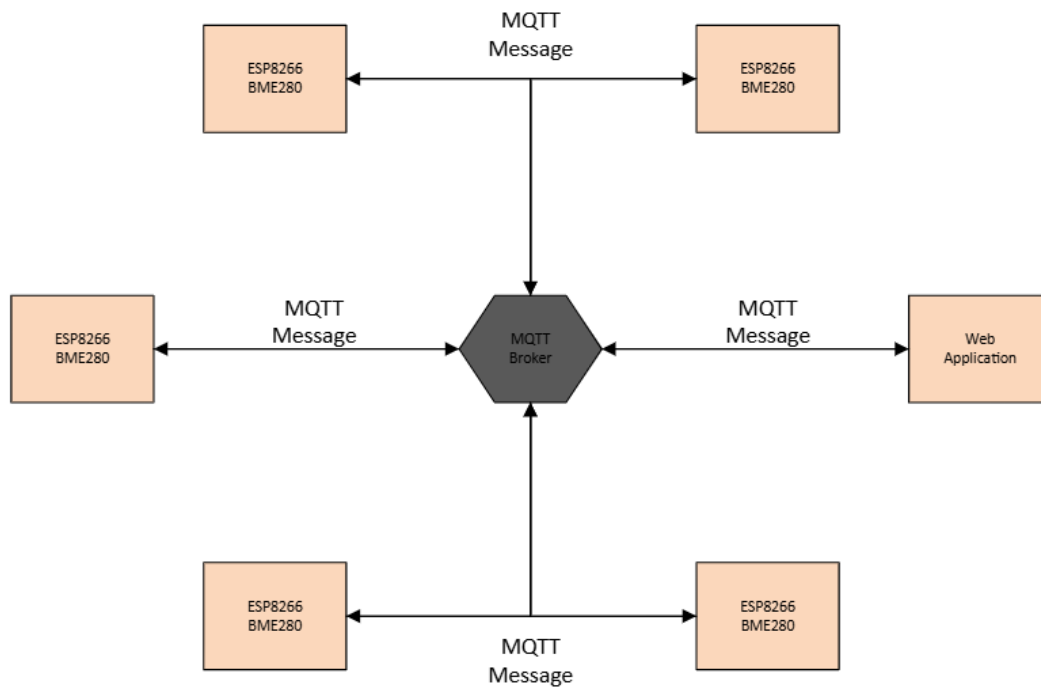
Arhitectura sistemului conține următoarele componente principale:

1. **Senzorii de umiditate** pentru colectarea datelor despre umiditate și detectarea inundațiilor. Datele sunt transmise către microcontroler.
2. **ESP2866**, microcontroller-ul principal ce procesează datele transmise de sensor, procesează datele și declanșează alertele locale (buzzer și LED). Datele procesate sunt trimise către baza de date din Influxdb și pe topicul aferent prin MQTT.
3. **Github Pages**, serverul cloud ce găzduiește aplicația web și asigură interfața prin care utilizatorul poate opri alarma.

4. **Broker-ul public MQTTX** pentru comunicația sigură prin web sockets cu aplicația web și prin portul dedicat cu microcontroller-ul esp8266.
5. **Baza de date Influxdb** ce stochează datele procesate de sensor.
6. **Grafana** pentru vizualizarea datelor și alertarea utilizatorului în cazul anomaliilor.



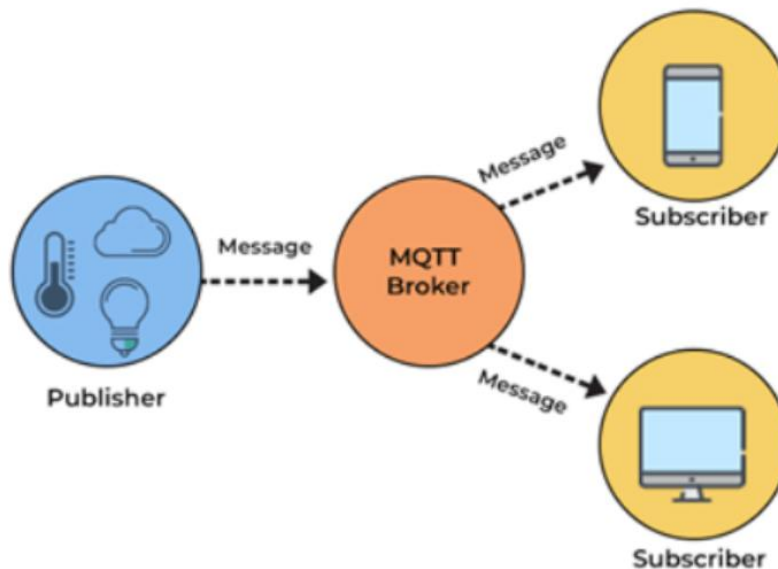
Schema Topologiei Sistemului: Stea



Topologia sistemului este de tip stea în rețea. Hub-ul central este broker-ul public MQTT, iar nodurile se vor conecta la broker folosind protocolul MQTT pentru a trimite datele spre vizualizare în aplicația web și pentru comunicarea pentru controlul actuatorilor de alarmă.

Protocoale

1. **Protocolul MQTT (Message Queueing Telemetry Transport)** este un protocol de comunicație lightweight, utilizat pe scară largă în IoT pentru transmiterea eficientă a mesajelor între dispozitive conectate. Acesta urmează un model publish/subscribe, ceea ce înseamnă că dispozitivele (clienții) pot publica date către un broker centralizat, iar alte dispozitive interesate de aceste date (abonate) le pot primi în timp real. În cazul nostru, atât aplicația web, cât și senzorul, joacă rol și de client și de abonat.



Avantaje:

- **Consum redus de bandă:** Protocolul este optimizat pentru rețele cu lățime de bandă limitată, ceea ce îl face ideal pentru dispozitive IoT, cum ar fi senzorii de temperatură, umiditate și presiune.
- **Low-power:** Dispozitivele care utilizează MQTT pot consuma mai puțină energie, prelungind durata de viață a bateriilor în cazul senzorilor wireless.
- **QoS (Quality of Service):** MQTT oferă trei niveluri de calitate a serviciului (QoS 0, QoS 1 și QoS 2), permițând dezvoltatorilor să ajusteze fiabilitatea transmisiei în

funcție de nevoile aplicației. Pentru proiect, un nivel QoS 1 (mesajul este livrat cel puțin o dată) este suficient pentru a asigura monitorizarea în timp real.

- **Scalabilitate:** Sistemul poate fi extins cu ușurință pentru a integra mai mulți senzori sau dispozitive fără modificări majore ale infrastructurii.
- **Latentă scăzută:** Asigură transmiterea rapidă a datelor, esențială pentru funcționalitatea sistemului de alerte.

Relevanță în proiect:

MQTT permite colectarea și transmiterea continuă a datelor de la senzori către aplicația centrală și, ulterior, către platforma de vizualizare (ex. Grafana). De asemenea, protocolul asigură comunicarea eficientă pentru declanșarea alarmelor atunci când condițiile depășesc pragurile setate.

2. Protocolul Wi-Fi (IEEE 802.11) este utilizat pentru conectarea senzorilor și dispozitivelor IoT la rețea, asigurând transferul de date către brokerul MQTT sau către serverul web.



Avantaje:

- **Rată de transfer mare:** Oferă viteze ridicate de transfer, ceea ce permite transmiterea datelor în timp real fără întârzieri semnificative.

- **Disponibilitate:** Wi-Fi este disponibil pe scară largă și este ușor de implementat în infrastructuri existente.
- **Cost redus:** Majoritatea dispozitivelor IoT sunt deja compatibile cu Wi-Fi, eliminând nevoia de echipamente suplimentare costisitoare.

Limitări:

- **Consum ridicat de energie:** Comparativ cu alte tehnologii wireless (ex. Zigbee), Wi-Fi poate consuma mai multă energie, ceea ce ar putea afecta durata de viață a bateriilor senzorilor.
- **Distanță limitată:** Performanța poate scădea în spații mari sau în zone cu obstacole, dar acest lucru poate fi compensat prin amplasarea de repetitoare sau puncte de acces suplimentare.

Relevanță în proiect:

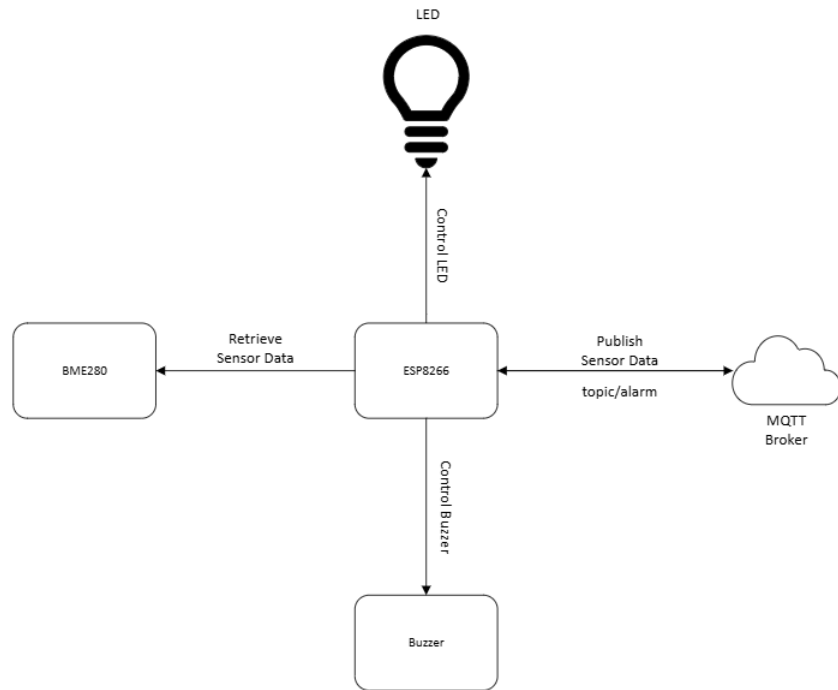
Wi-Fi asigură conectivitatea între senzorii de temperatură, umiditate și presiune și sistemul central de monitorizare. Este esențial pentru transmiterea rapidă a datelor și declanșarea alarmelor în timp real.

Implementare

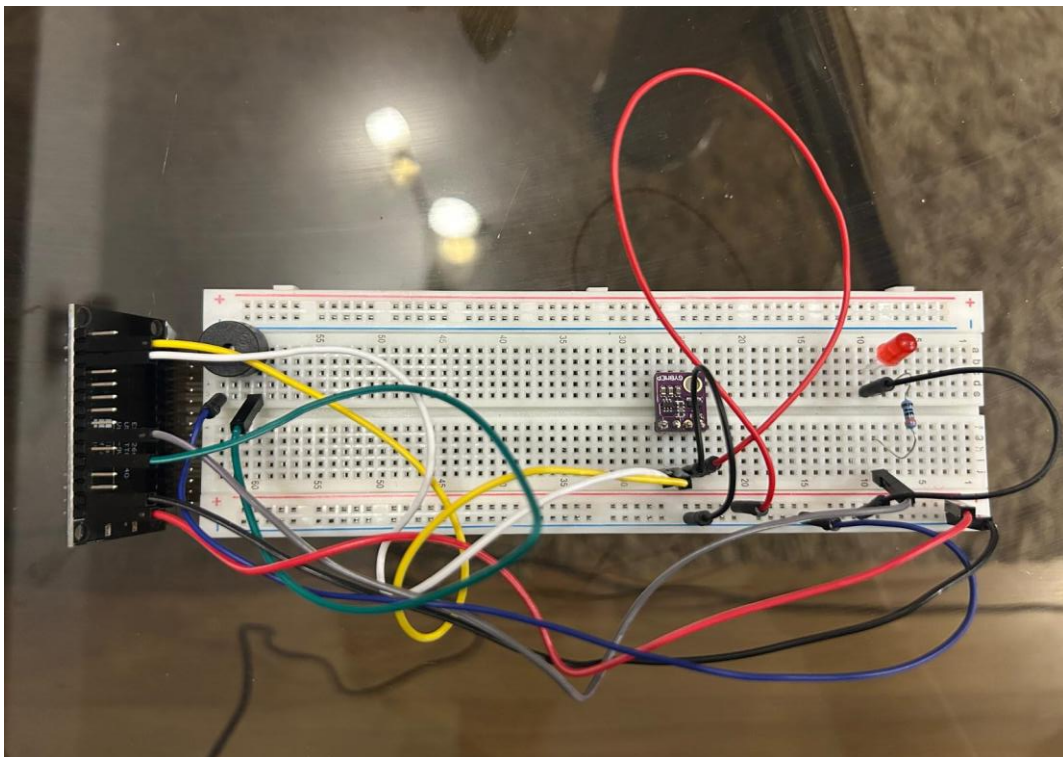
Resurse Hardware:

- Microcontroler: **ESP8266** (pentru conectivitate WiFi).
- Senzor de umiditate: **BME280** (pentru măsurarea nivelului de umiditate).
- **Buzzer** activ (pentru emiterea alertelor sonore).
- **LED** (pentru semnalizare vizuală).
- Rezistențe: **330Ω** (pentru LED).

Circuit Design Hardware:



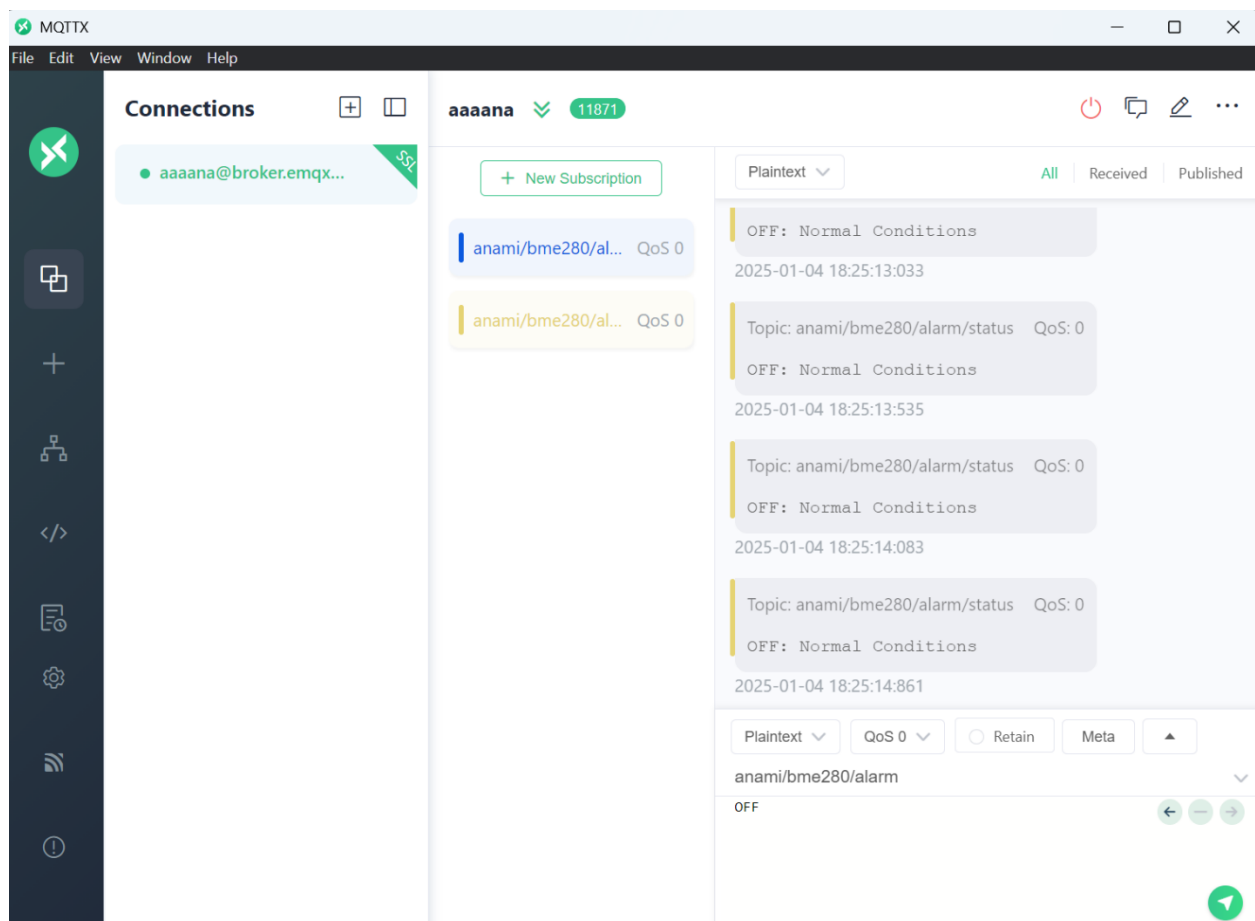
Pentru detalii legat de implementarea hardware, se pot consulta link-urile adăugate sub secțiunea de resurse.



Resurse software:

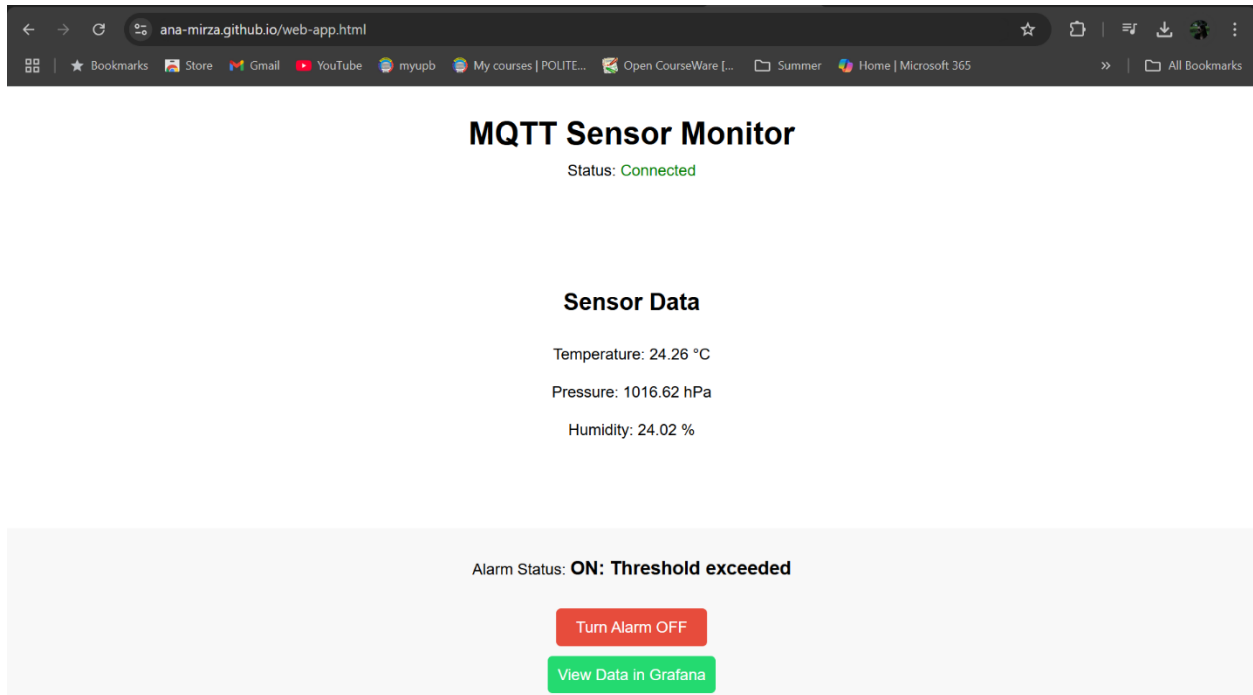
- **Arduino IDE** (pentru programarea microcontrolerului ESP8266)
- Server web in cloud (pentru găzduirea aplicației web): **Github Pages**
- Aplicație web: **HTML, CSS, JavaScript** (pentru interfața utilizator)
- Broker MQTT Public în Cloud: **MQTTX**
- Vizualizare: **Grafana**
- Stocare de date: **InfluxDB**

Interfața Aplicației MQTTX:

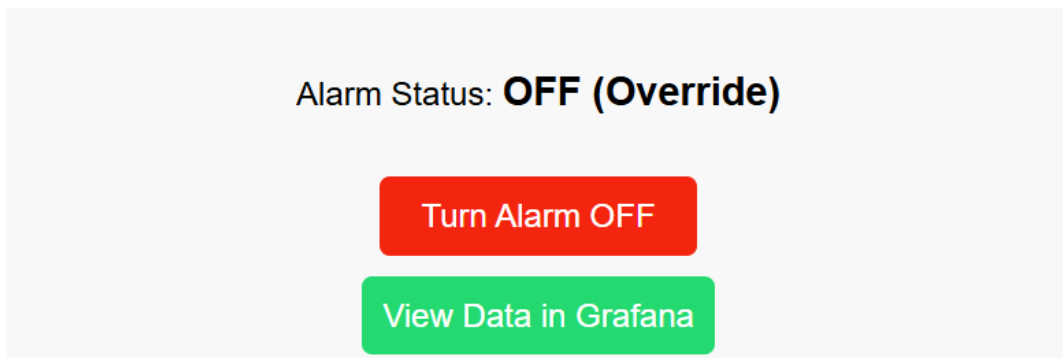


Din aplicație se poate testa conexiunea cu senzorul și aplicația web, trimițând mesaje de test pe canalele folosite și vizualizând traficul de mesaje MQTT.

Interfața Aplicației Web:



Atunci când alarma este oprită din aplicație, se va afișa un status ce indică aceasta.



Conexiune MQTT prin Web Sockets:

```
<script>
const broker = "wss://broker.emqx.io:8084/mqtt"; // WebSocket connection to EMQX
const topicAlarm = "anami/bme280/alarm";
const topicTemp = "anami/bme280/temperature";
const topicPressure = "anami/bme280/pressure";
const topicHumidity = "anami/bme280/humidity";
const topicStatus = "anami/bme280/alarm/status";

const client = mqtt.connect(broker);
```

Pentru conexiunea cu brokerul public de MQTT, se folosește un script JavaScript. Pentru mai multe detalii legate de implementarea conexiunii, se pot consulta link-urile atașate din secțiunea de resurse.

```
let isAlarmOn = false; // Track the alarm state

client.on("connect", () => {
  statusEl.textContent = "Connected";
  statusEl.style.color = "green";
  client.subscribe([topicAlarm, topicTemp, topicPressure, topicHumidity, topicStatus]);
});

client.on("message", (topic, message) => {
  const msg = message.toString();
  if (topic === topicTemp) tempEl.textContent = msg;
  if (topic === topicPressure) pressureEl.textContent = msg;
  if (topic === topicHumidity) humidityEl.textContent = msg;
  if (topic === topicStatus) alarmStateEl.textContent = msg;
  if (topic === topicStatus) {
    alarmStateEl.textContent = msg;

    // Update local alarm state based on received status
    if (msg.startsWith("ON")) {
      isAlarmOn = true;
    } else if (msg.startsWith("OFF")) {
      isAlarmOn = false;
    }
  }
});
```

Aplicația web contorizează o apăsare pe butonul de oprire a alarmei și trimite pe topicul de alarmă semnalul de off. Alarma rămâne închisă pentru o perioadă determinată de timp, după care se reactivează dacă motivul alarmei nu a dispărut. De asemenea, aplicația nu permite apăsarea butonului de alarmă cât timp alarma nu este pornită.

```
btnOff.addEventListener("click", () => {
  if (isAlarmOn) {
    // Publish the override only if the alarm is currently on
    client.publish(topicAlarm, "OFF");
    client.publish(topicStatus, "OFF (Override)");
  } else {
    console.log("Alarm is already off, no override needed.");
  }
});
```

Controlul LED-ului și a buzzer-ului:

Senzorul este subscris la topicul de alarmă și primește mesajele de control trimise de aplicație, activînd sau oprind LED-ul și buzzer-ul de alarmă.

```
240
241 void mqttCallback(char *topic, byte *payload, unsigned int length) {
242     String message;
243     for (int i = 0; i < length; i++) {
244         message += (char)payload[i];
245     }
246
247     Serial.print("Message received on topic ");
248     Serial.print(topic);
249     Serial.print(": ");
250     Serial.println(message);
251
252     // Check the alarm control message
253     if (String(topic) == mqtt_topic) {
254         if (message == "ON") {
255             digitalWrite(ledPin, HIGH); // Turn LED on
256             buzzer.playMelody(melody, noteDurations, noteLength);
257             Serial.println("Alarm ON");
258
259             if (!alarmOn) {
260                 mqtt_client.publish("anami/bme280/alarm/status", "ON");
261             }
262             alarmOn = true;
263         } else if (message == "OFF" && manualOverride == false && alarmOn) {
264             alarmOn = false;
265
266             digitalWrite(ledPin, LOW); // Turn LED off
267             buzzer.stop();
268             Serial.println("LED OFF (Manual Override)");
269             manualOverride = true;
270             overrideStartTime = millis(); // Start override timer
271         }
272     }
273 }
```

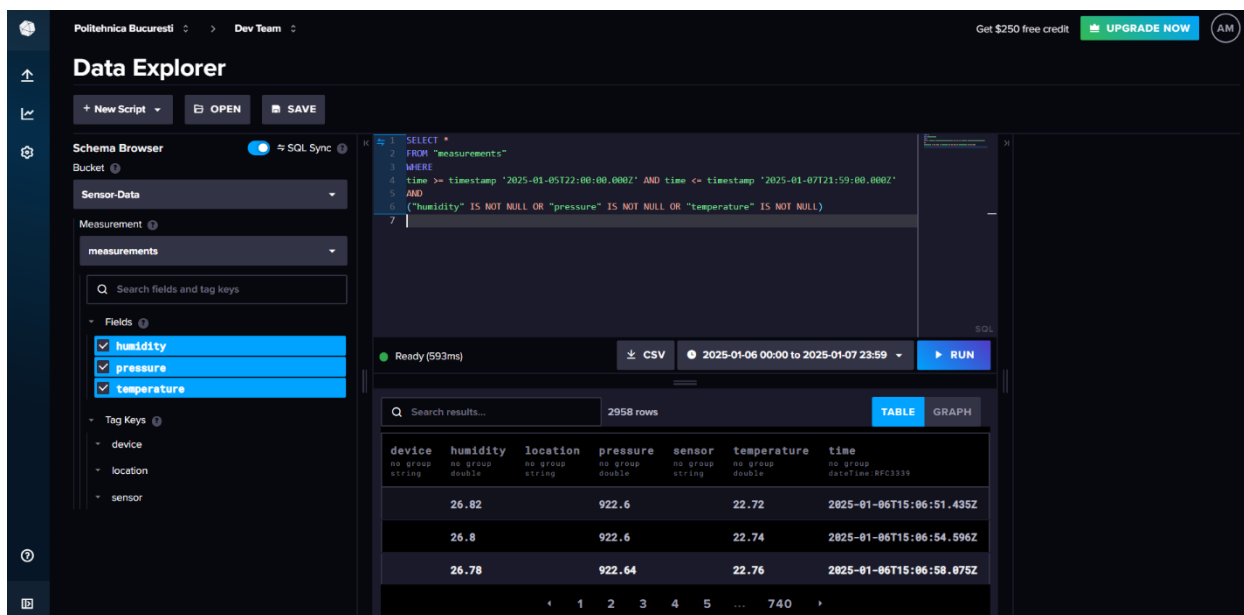
Vizualizare și Procesare de Date

În cadrul proiectului nostru, colectarea, procesarea și vizualizarea datelor provenite de la senzorii de temperatură, umiditate și presiune sunt realizate printr-o combinație de tehnologiile MQTT, InfluxDB și Grafana, alături de o aplicație web interactivă care permite vizualizarea datelor în timp real și gestionarea alarmelor.

InfluxDB - Baza de Date pentru Date Temporale

InfluxDB este o bază de date special concepută pentru stocarea și gestionarea datelor temporale (time-series data), ceea ce o face ideală pentru proiecte care colectează date din senzorii care furnizează măsurători constante, precum temperatură, umiditate sau presiune. InfluxDB permite stocarea și interogarea rapidă a datelor pe intervale de timp, oferind un suport excelent pentru analizele în timp real.

- **Eficiență în stocare:** InfluxDB este optimizat pentru a gestiona cantități mari de date generate de senzori, utilizând un model de date care permite stocarea eficientă și rapidă a informațiilor. Acest lucru este esențial pentru aplicațiile care monitorizează continuu mediul sau diverse parametri de funcționare, precum în cazul proiectului nostru.
- **Interogări rapide:** Datorită arhitecturii sale, InfluxDB permite efectuarea rapidă a interogărilor complexe pe date temporale. De exemplu, se pot obține rapid statistici precum medii, maxime, minime, variații sau tendințe pe intervale de timp specificate, ceea ce este esențial pentru analiza datelor provenite de la senzorii de temperatură, umiditate și presiune.
- **Scalabilitate:** InfluxDB poate gestiona volume mari de date în timp real, ceea ce este un avantaj semnificativ într-un sistem IoT sau de monitorizare continuă, unde se pot adăuga cu ușurință noi surse de date fără a afecta performanța.



The screenshot displays the InfluxDB Data Explorer interface. On the left, the 'Schema Browser' shows a bucket named 'Sensor-Data' with a measurement named 'measurements'. The 'Fields' section lists 'humidity', 'pressure', and 'temperature', all of which are checked. The 'Tag Keys' section lists 'device', 'location', and 'sensor'. The main area shows a SQL query:

```
SELECT * FROM "measurements" WHERE time >= timestamp '2025-01-05T22:00:00.000Z' AND time <= timestamp '2025-01-07T21:59:00.000Z' AND ("humidity" IS NOT NULL OR "pressure" IS NOT NULL OR "temperature" IS NOT NULL)
```

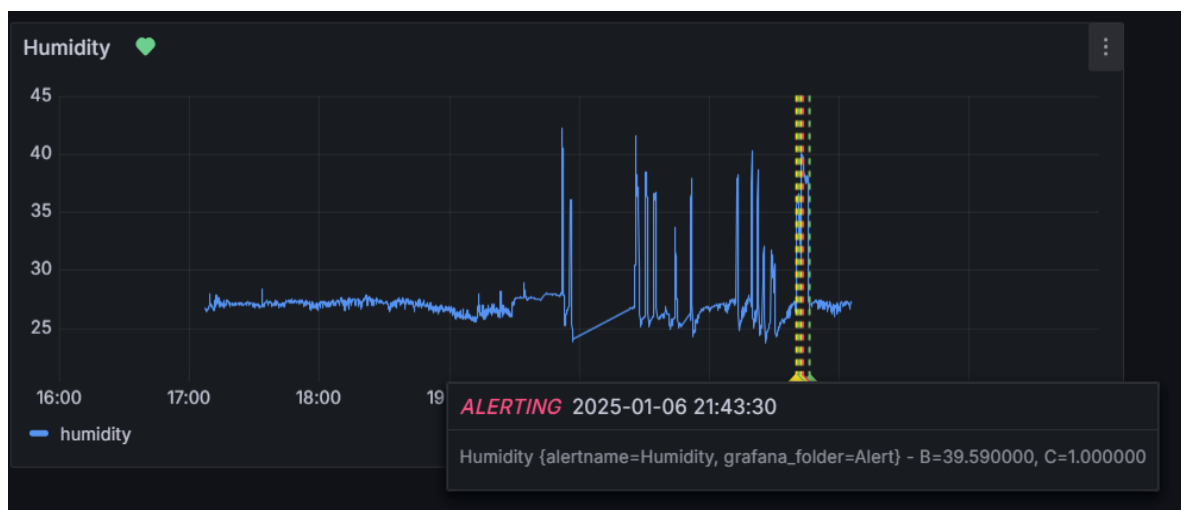
. Below the query, the status 'Ready (593ms)' is shown. The results are displayed in a table with 2958 rows. The table has columns: device, humidity, location, pressure, sensor, temperature, and time. The first three rows of data are visible.

device	humidity	location	pressure	sensor	temperature	time
no group	no group	no group	no group	no group	no group	no group
string	double	string	double	string	double	datetime RFC3339
	26.82		922.6		22.72	2025-01-06T15:06:51.435Z
	26.8		922.6		22.74	2025-01-06T15:06:54.596Z
	26.78		922.64		22.76	2025-01-06T15:06:58.075Z

Grafana - Vizualizarea Datelor

Grafana este o platformă de vizualizare a datelor open-source, care se integrează perfect cu InfluxDB pentru a crea panouri interactive și vizualizări detaliate ale datelor colectate de senzorii din proiect. Grafana este utilizată pentru a crea dashboard-uri care permit utilizatorilor să monitorizeze parametrii relevanți într-un mod intuitiv, având posibilitatea de a vizualiza datele în timp real.

- **Crearea de Dashboard-uri Interactive:** Grafana permite crearea de dashboard-uri personalizate care afișează grafice, diagrame și alerte pe baza datelor din InfluxDB. Aceste dashboard-uri sunt utile pentru a monitoriza continuu valorile senzorilor și pentru a observa orice fluctuație semnificativă în parametrii măsurați, cum ar fi temperaturile sau umiditatea dintr-un mediu monitorizat.
- **Interfață Intuitivă:** Grafana oferă o interfață prietenoasă, care permite utilizatorilor să vizualizeze rapid evoluția datelor în timp. Acest lucru este important pentru a asigura o monitorizare eficientă a mediului sau a parametrilor, fiind posibilă chiar personalizarea vizualizărilor pentru a evidenția aspectele cele mai importante.
- **Alerte și Notificări:** Grafana permite setarea de alerte care pot fi declanșate atunci când valorile senzorilor depășesc anumite praguri. De exemplu, în cazul unui senzor de temperatură sau umiditate, un prag poate fi setat pentru a emite o alarmă atunci când temperatura depășește o valoare critică. Alertele pot fi trimise prin diverse canale, cum ar fi email, Slack sau chiar SMS, astfel încât utilizatorul să poată reacționa rapid în cazul în care sunt detectate anomalii.



[FIRING:1] Humidity Alert [Message primite x](#)

Grafana <grafana@anamirza2002.grafana.net>
către eu

Tradu în română

lun., 6 ian., 21:44 (acum 6 zile) ☆ 😊 ↶ ⋮

Grafana

Alert > Humidity

1 firing instances

Firing	Humidity	View alert				
<p>Summary</p> <p>Humidity above threshold.</p> <p>Values</p> <p>B=39.59 C=1</p> <p>Labels</p> <table><tr><td>alertname</td><td>Humidity</td></tr><tr><td>grafana_folder</td><td>Alert</td></tr></table> <p>Silence View dashboard View panel</p>			alertname	Humidity	grafana_folder	Alert
alertname	Humidity					
grafana_folder	Alert					

Integrarea InfluxDB și Grafana în Proiect

În proiectul nostru, InfluxDB joacă rolul principal în stocarea și gestionarea datelor provenite de la senzorii de temperatură, umiditate și presiune. Datele sunt colectate și trimise într-un flux continuu către baza de date, unde sunt organizate pe intervale de timp. Acest proces asigură o arhivare precisă a tuturor măsurărilor, facilitând analizele ulterioare și monitorizarea evoluției parametrilor în timp.

Ulterior, Grafana preia aceste date din InfluxDB și le transformă într-o interfață vizuală ușor de înțeles, sub formă de grafice și diagrame. Dashboard-urile create permit monitorizarea continuă a parametrilor în timp real, iar alertele configurate în Grafana asigură notificarea imediată în cazul în care sunt detectate valori anormale sau periculoase, cum ar fi temperaturi extreme sau niveluri de umiditate care pot afecta siguranța sau performanța unui sistem.



Beneficiile Utilizării InfluxDB și Grafana

- **Vizualizare în Timp Real:** Utilizatorii pot vizualiza continuu datele provenite de la senzorii de temperatură, umiditate și presiune, asigurându-se că parametrii sunt menținuți în intervalele dorite.
- **Analiză Detaliată:** InfluxDB permite realizarea de analize complexe asupra datelor istorice, iar Grafana facilitează compararea valorilor pe diferite perioade de timp pentru a observa tendințele și a anticipa comportamentele viitoare.
- **Răspuns Rapid:** Alertele în Grafana permit reacția rapidă în cazul în care parametrii măsurați ies din limitele normale, prevenind astfel posibile probleme sau daune.
- **Ușurință în Configurare și Extindere:** Ambele tehnologii sunt ușor de configurat și scalabil, ceea ce înseamnă că proiectul poate fi extins ușor pe măsură ce sunt adăugați noi senzori sau cerințele se modifică.

Aplicația Web - Vizualizare și Control

Aplicația web joacă un rol central în integrarea acestor tehnologii, având următoarele funcționalități și caracteristici:

Vizualizarea datelor senzorilor în timp real:

Aplicația web se conectează la brokerul MQTT pentru a primi datele în timp real de la senzorii de temperatură, umiditate și presiune. Aceste date sunt afișate pe interfața web într-un format ușor de înțeles. Valorile sunt actualizate automat fără a fi necesar să reîmprospătezi pagina.

Integrarea cu Grafana pentru vizualizare avansată:

Aplicația web include un link către un dashboard Grafana, care vizualizează datele istorice ale senzorilor și permite utilizatorilor să analizeze tendințele pe perioade mai lungi de timp. Grafana permite crearea de grafice dinamice, de exemplu, pentru a vizualiza cum au evoluat temperatura și umiditatea în ultimele zile sau săptămâni.

Gestionarea alarmei:

Starea alarmei este afișată în aplicația web, iar utilizatorul poate opri alarma din interfață printr-un buton dedicat. Acest buton trimite un mesaj MQTT pentru a dezactiva alarma, dacă aceasta este activă. De asemenea, aplicația afișează statusul curent al alarmei (ON/OFF), pe baza mesajelor primite de la brokerul MQTT.

Vizualizarea ultimelor date și status alarmei:

Pe lângă afișarea în timp real a datelor de la senzori, aplicația web permite vizualizarea ultimelor valori măsurate (temperatură, umiditate și presiune). De asemenea, aplicația prezintă statusul alarmei, indicând dacă aceasta este activă sau inactivă, și dacă sunt necesare acțiuni suplimentare.

Securitate

Partea de securitate prezintă un subiect important pentru acest proiect. De aceea, sunt implementate mai multe măsuri de securitate pentru a proteja integritatea și confidențialitatea datelor. Detaliile sunt prezentate mai jos:

1. Conexiune securizată MQTT prin TLS/SSL

Pentru comunicarea cu brokerul MQTT, utilizăm un canal criptat bazat pe protocolul TLS (Transport Layer Security). Acest lucru asigură că toate datele transmise sunt protejate împotriva interceptării și modificării.

- **Certificat SSL:** Proiectul include certificatul autorității de certificare utilizate de brokerul MQTT. Acesta este stocat în cod sub forma unei constante `ca_cert`.
- **Validarea certificatelor:** Dispozitivul ESP8266 validează certificatul serverului MQTT folosind certificatul stocat local. Astfel, conexiunile către servere neautorizate sunt prevenite.
- **Autentificare la broker folosind credențiale:** Sunt utilizate un nume de utilizator (`mqtt_username`) și o parolă (`mqtt_password`) pentru autentificare suplimentară la brokerul MQTT.

Codul relevant pentru configurarea conexiunii securizate MQTT este:

```

219 void connectToMQTT() {
220     BearSSL::X509List serverTrustedCA(ca_cert);
221     espClient.setTrustAnchors(&serverTrustedCA); // Set the trusted root certificate
222     while (!mqtt_client.connected()) {
223         String client_id = "esp8266-client-" + String(WiFi.macAddress());
224         Serial.printf("Connecting to MQTT Broker as %s.....\n", client_id.c_str());
225         if (mqtt_client.connect(client_id.c_str(), mqtt_username, mqtt_password)) {
226             Serial.println("Connected to MQTT broker");
227             mqtt_client.subscribe(mqtt_topic);
228             mqtt_client.subscribe(mqtt_topic_alarm);
229         } else {
230             char err_buf[128];
231             espClient.getLastSSLError(err_buf, sizeof(err_buf));
232             Serial.print("Failed to connect to MQTT broker, rc=");
233             Serial.println(mqtt_client.state());
234             Serial.print("SSL error: ");
235             Serial.println(err_buf);
236             delay(2000);
237         }
238     }
239 }

```

2. Conexiune sigură la InfluxDB

Pentru integrarea cu baza de date InfluxDB, utilizăm protocolul HTTPS, care criptează toate comunicațiile între dispozitiv și serverul InfluxDB.

- **Certificare server:** Clientul InfluxDB validează conexiunea cu serverul utilizând un certificat de securitate preconfigurat (`InfluxDbCloud2CACert`).
- **Autentificare prin token:** Pentru a accesa baza de date, proiectul utilizează un token unic (`INFLUXDB_TOKEN`), care este asociat cu organizația și bucket-ul din InfluxDB.

Codul de configurare a clientului InfluxDB include:

```
203 void initInfluxdbClient() {
204     // Check server connection
205     if (client.validateConnection()) {
206         Serial.print("Connected to InfluxDB: ");
207         Serial.println(client.getServerUrl());
208     } else {
209         Serial.print("InfluxDB connection failed: ");
210         Serial.println(client.getLastErrorMessage());
211     }
212 }
```

Provocări și Soluții

Una din provocările întâmpinate la începutul proiectului a fost integrarea protocolului MQTT cu aplicația web, deoarece aceasta comunica prin HTTPS cu serverul. După puțină documentare pe google am descoperit că și aplicațiile web se pot conecta la brokerii MQTT, dar spre dezamăgirea mea, nu și la cel local care folosea Mosquito, deoarece acesta nu suporta conexiuni prin web sockets. Am găsit o soluție într-un final prin folosirea unui broker de MQTT public (în cloud) ce avea și opțiunea de web sockets. Astfel, am putut integra toate componentele rețelei IoT spre a folosi doar protocolul MQTT, fără HTTPS, care ar fi folosit mult mai multă memorie și lățime de bandă.

Încă o provocare întâlnită a fost răspunsul întârziat la comenzile date din aplicație, datorate comunicației supraîncărcate, deoarece microcontroler-ul trimitea datele citite către baza de date, citea mesajele primite de la borker-ul de mqtt și update starea internă a alarmei la fiecare iterație. Soluția găsită a constat în trimiterea datelor către InfluxDB la un interval de timp prestabilit, o dată la 5 secunde, în locul trimiterii la fiecare iterație din loop-ul interior.

Referințe:

Poze:

1. Vinărie: <https://cluboenologique.com/story/a-decade-of-hedonism-wines-shop/>
2. WiFy: <https://www.cdebyte.com/news/759>
3. MQTT: <https://oxeltech.de/en/mqtt-a-basic-guide-for-iot-developers/>

Implementare Hardware:

1. Tutorial bme280: <https://randomnerdtutorials.com/esp8266-bme280-arduino-ide/>
2. Tutorial buzzer: <https://newbiely.com/tutorials/esp8266/esp8266-piezo-buzzer>
3. Tutorial LED: https://www.emqx.com/en/blog/esp8266_mqtt_led

Implementare Software:

1. Tutorial control LED folosind MQTTX:
https://www.emqx.com/en/blog/esp8266_mqtt_led
2. Tutorial conexiune esp8266 la MQTTX: <https://www.emqx.com/en/blog/esp8266-connects-to-the-public-mqtt-broker>
3. Tutorial MQTT peste web sockets: <https://www.emqx.com/en/blog/connect-to-mqtt-broker-with-websocket>