Introduzione

Questo progetto IoT utilizza una scheda Arduino R4 WiFi, un sensore di temperatura e umidità DHT11, un buzzer e tre pulsanti per configurare offset di temperatura e umidità. L'obiettivo è monitorare la temperatura e l'umidità, inviare i dati a un server MQTT e visualizzarli su una web app. Se la temperatura o l'umidità superano i valori di offset impostati, il buzzer si attiva.

Componenti utilizzati:

- **Arduino R4 WiFi**
- **Sensore di temperatura e umidità DHT11**
- **Buzzer**
- **Tre pulsanti**
- **Display LCD I2C**
- **WiFi**
- **Server MQTT (RabbitMQ)**
- **PHP per il backend**
- **Database MySQL**

Collegamenti Hardware:

- **Buzzer**: Collega il pin positivo al pin digitale 4 di Arduino e il pin negativo a GND.
- **Sensore DHT11**: Collega il pin di uscita al pin digitale 8 di Arduino.
- **Pulsanti**:
    - Button Up: Collega un terminale al pin digitale 7 di Arduino e l'altro a GND.
    - Button Down: Collega un terminale al pin digitale 6 di Arduino e l'altro a GND.
    - Button Select: Collega un terminale al pin digitale 5 di Arduino e l'altro a GND.
- **Display LCD I2C**: Collega SDA e SCL ai pin corrispondenti su Arduino (A4 e A5).

Funzionamento

Configurazione degli Offset

All'avvio del sistema, l'utente ha la possibilità di configurare gli offset di temperatura e umidità utilizzando i pulsanti forniti (Up, Down e Select). Gli offset rappresentano una deviazione dalla temperatura e dall'umidità rilevata dal sensore. Ad esempio, se l'utente imposta un offset di temperatura di +2°C, l'allarme verrà attivato solo quando la temperatura rilevata supererà di 2 gradi il valore impostato.

Durante la configurazione degli offset, i valori selezionati sono visualizzati sul display LCD, permettendo all'utente di monitorare e regolare con precisione i parametri desiderati. Una volta confermati gli offset, vengono memorizzati per l'utilizzo successivo.

Monitoraggio e Allarme

Una volta che gli offset sono stati impostati, Arduino inizia il monitoraggio continuo della temperatura e dell'umidità ambientale utilizzando il sensore DHT11. Questi valori sono letti a intervalli regolari e confrontati con gli offset precedentemente configurati.

Se la temperatura o l'umidità superano i valori di offset, il buzzer si attiva emettendo un segnale acustico di avviso. Questo avviso serve a segnalare all'utente che i parametri ambientali hanno superato la soglia impostata, potenzialmente indicando un'eventuale variazione indesiderata delle condizioni ambientali.

Invio dei Dati al Server MQTT

Parallelamente al monitoraggio locale, Arduino invia periodicamente i dati di temperatura e umidità a un server MQTT. Il server MQTT funge da intermediario per la comunicazione tra il dispositivo Arduino e la web app PHP.

La web app PHP si connette al server MQTT per ricevere i dati di temperatura e umidità in tempo reale. Una volta ricevuti, i dati vengono elaborati e salvati in un database MySQL per consentire una conservazione permanente e la possibilità di analisi storiche. Infine, la web app PHP visualizza i dati sul proprio frontend, consentendo all'utente di monitorare le condizioni ambientali da qualsiasi dispositivo connesso a Internet.

Web App e Visualizzazione Grafica

Struttura della Web App

La web app è stata sviluppata utilizzando PHP per il backend, che gestisce la connessione al server MQTT e l'interazione con il database MySQL. La parte frontend utilizza Google Charts per visualizzare i dati di temperatura e umidità in tempo reale.

Visualizzazione dei Dati

Grafici

Sono stati implementati due grafici principali per la visualizzazione dei dati:

1. **Grafico della Temperatura**:
    - **Dati**: Visualizza le letture della temperatura nel tempo.
    - **Asse X**: Rappresenta il timestamp dei dati raccolti, visualizzato in formato `HH:MM:SS`.
    - **Asse Y**: Rappresenta i valori di temperatura in gradi Celsius.
2. **Grafico dell'Umidità**:
    - **Dati**: Visualizza le letture dell'umidità nel tempo.
    - **Asse X**: Rappresenta il timestamp dei dati raccolti, visualizzato in formato `HH:MM:SS`.
    - **Asse Y**: Rappresenta i valori di umidità in percentuale.

Questi grafici forniscono una visione chiara e immediata delle variazioni di temperatura e umidità nel tempo, permettendo agli utenti di monitorare facilmente le condizioni ambientali.

Ultimi Valori

Accanto ai grafici, la web app visualizza gli ultimi valori di temperatura e umidità rilevati, offrendo un rapido riepilogo delle condizioni attuali. Questa sezione è posizionata a sinistra del grafico per un accesso immediato e una visualizzazione bilanciata.

file progetto_iot.ino

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>

LiquidCrystal_I2C lcd(0x27, 20, 4);

const char* ssid = "PosteMobile-90353337";           // SSID WIFI
const char* password = "2022AleChrisMir#2022";  // psw
const char* mqtt_server = "192.168.1.5";  // Indirizzo IP del server
WiFiClient wifiClient;
PubSubClient client(wifiClient);
#define DHTPIN 8      // Pin digitale al quale è collegato il DHT11
#define DHTTYPE DHT11 // Tipo di sensore DHT

DHT dht(DHTPIN, DHTTYPE);

int buttonUp = 7;
int buttonDown = 6;
int buttonSelect = 5;

bool selectedTemp = false;
```

```
bool selectedHum = false;

bool esciCiclo = false;



int tempOffset = 0;

int humOffset = 0;

int buzzer = 4;



void setup() {

  lcd.init();

  lcd.backlight();

  lcd.setCursor(0,0);



  pinMode(buttonUp, INPUT);

  pinMode(buttonDown, INPUT);

  pinMode(buttonSelect, INPUT);

  pinMode(buzzer, OUTPUT);



  Serial.begin(115200);

  Serial.begin(9600);

  dht.begin();

  setup_wifi();

  client.setServer(mqtt_server, 1883);

  lcd.print("temp di offset: ");



  while(!esciCiclo){
```

```
  if(!selectedTemp){
    lcd.setCursor(0,1);
    if(digitalRead(buttonUp)){
      Serial.println("hai premuto il pulsante");
      lcd.clear();
      lcd.setCursor(0, 0);
      tempOffset +=2;
      lcd.print("temp di offset: ");
      lcd.setCursor(0,1);
      lcd.print(tempOffset);
      Serial.println(tempOffset);
    }
    else if(digitalRead(buttonDown)){
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("temp di offset: ");
      tempOffset -=2;
      lcd.setCursor(0, 1);
      lcd.print(tempOffset);
    }
    else if(digitalRead(buttonSelect)){
      selectedTemp = true;
      lcd.clear();
      lcd.setCursor(0, 0);
```

```
        lcd.print("umid di offset: ");
        Serial.println(selectedHum);
        Serial.println(selectedTemp);
      }
    }
    else if(selectedTemp && !selectedHum){
      Serial.println("seleziona umidità");
      if(digitalRead(buttonUp)){
        Serial.println("hai premuto il pulsante up");
        humOffset +=2;
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("umid. di offset: ");
        lcd.setCursor(0, 1);
        lcd.print(humOffset);
      }
      else if(digitalRead(buttonDown)){
        humOffset -=2;
        lcd.print(humOffset);
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("umid. di offset: ");
        lcd.setCursor(0, 1);
        lcd.print(humOffset);
      }
```

```
      else if(digitalRead(buttonSelect)){
        esciCiclo = true;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("salvando....");
        delay(1000);
        lcd.clear();
        lcd.print("salvato");
        delay(1000);
        lcd.clear();
        Serial.println("seleziona umidità");
      }
    }
    delay(100);
  }
}
void setup_wifi() {
  lcd.clear();
  lcd.setCursor(0, 0);
  delay(10);
  Serial.println();
  Serial.print("Connessione a ");
  lcd.print("connessione a:");
  lcd.setCursor(0, 1);
  lcd.print(ssid);
```

```
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("wifi connesso");
  delay(1500);
  lcd.clear();
  Serial.println("");
  Serial.println("WiFi connesso");
  Serial.println("Indirizzo IP: ");
  Serial.println(WiFi.localIP());
}
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  delay(10000);

  // Leggi l'umidità
  float h = dht.readHumidity();
```

```
// Leggi la temperatura in Celsius (impostazione predefinita)
float t = dht.readTemperature();
// Leggi la temperatura in Fahrenheit (isFahrenheit = true)
float f = dht.readTemperature(true);


// Controlla se le letture hanno avuto successo
if (isnan(h) || isnan(t) || isnan(f)) {
  Serial.println("Errore nella lettura del sensore DHT!");
  return;
}




Serial.print("Temperatura: ");
Serial.print(t);
Serial.print(" °C, Umidità: ");
Serial.print(h);
Serial.println("%");
String payload = String(t) + "," + String(h);
char msg[100];
payload.toCharArray(msg, 100);
client.publish("sensor_data", msg);
Serial.println("publish on sensor_data");


lcd.clear();
```

```
  lcd.setCursor(0,0);
  lcd.print("temp: ");
  lcd.print(t);
  lcd.setCursor(0,1);
  lcd.print("umidita: ");
  lcd.print(h);


  if(t > tempOffset){
    tone(buzzer, 1000);
    delay(1000);
    noTone(buzzer);
    delay(1000);
  }
  else if(h > humOffset){
    tone(buzzer, 1000);
    delay(1000);
    noTone(buzzer);
    delay(1000);
  }


}
void reconnect() {
  while (!client.connected()) {
    Serial.print("Connessione al server MQTT...");
    if (client.connect("arduinoClient", "mqtt-test", "mqtt-test")) {
```

```
      lcd.print("connesso server mqtt");
      delay(1500);
      lcd.clear();
      Serial.println("Connesso");
    } else {
      lcd.print("connessione fallita");
      Serial.print("Fallito, rc=");
      Serial.print(client.state());
      Serial.println(" Riprovo fra 5 secondi");
      delay(5000);
      lcd.clear();
    }
  }
}
```

file arduino_mqtt.php

```php
<?php

require __DIR__ . '/vendor/autoload.php'; // Includi il file
autoload.php di Composer

use Bluerhinos\phpMQTT;

require_once('connectDB.php');



$host = "localhost"; // Indirizzo IP del server MQTT

$port = 1883;

$username = "mqtt-test";

$password = "mqtt-test";

$client_id = "altro";

$topic = "sensor_data";

$latest_temperature = "50";

$latest_humidity = "50";



function procmsg($topic, $msg)

{

    global $latest_temperature, $latest_humidity;

    $data = explode(",", $msg);

    $latest_temperature = $data[0];

    $latest_humidity = $data[1];

    saveSensorData($latest_temperature, $latest_humidity);

}
```

```php
$mqtt = new phpMQTT($host, $port, $client_id);

if (!$mqtt->connect(false, NULL, $username, $password)) {

    exit(1);

}



$topics[$topic] = array('qos' => 0, 'function' => 'procmsg');

$mqtt->subscribe($topics, 0);

$start_time = time();



while ($mqtt->proc()) {

    if (time() - $start_time >= 9) {

        break;

    }

}

$mqtt->close();

?>
```

file connectDB.php

```php
<?php
session_start();

$conn = new mysqli("localhost", "root", "", "AmbientMonitoringDB");
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

function saveSensorData($temperature, $humidity)
{
    global $conn;
    // Utilizzo di istruzioni preparate per evitare SQL injection
    $stmt = $conn->prepare("INSERT INTO sensordata (Temperature, Humidity, Timestamp) VALUES (?, ?, CURRENT_TIMESTAMP())");
    $stmt->bind_param("dd", $temperature, $humidity);
    if (!$stmt->execute()) {
        echo "Error: " . $stmt->error;
    }
    $stmt->close();
}
?>
```

file index.php

```
<?php
include __DIR__ .'/arduino_mqtt.php';
require_once('connectDB.php');
?>

<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="refresh" content="20">
    <meta    name="viewport"    content="width=device-width,    initial-
scale=1.0">
    <title>Interfaccia Web</title>
    <link rel="stylesheet" href="styles.css">
    <script                                          type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>
    <script type="text/javascript">
      google.charts.load('current', {'packages':['corechart']});
      google.charts.setOnLoadCallback(drawCharts);

      function drawCharts() {
        drawHumidityChart();
        drawTemperatureChart();
      }
```

```
      function drawHumidityChart() {

        var data = google.visualization.arrayToDataTable([

          ['Timestamp', 'Humidity'],

          <?php

                $query = "SELECT TIME(Timestamp) AS Time, Humidity FROM
sensordata ORDER BY Timestamp LIMIT 20";

                $res = mysqli_query($conn, $query);

                if ($res) {

                    while($data = mysqli_fetch_assoc($res)){

                        $timestamp = $data["Time"];

                        $humidity = $data["Humidity"];

                        echo "['$timestamp', $humidity],";

                    }

                }

          ?>

        ]);


        var options = {

          title: 'Humidity Data',

          curveType: 'function',

          legend: { position: 'bottom' },

          hAxis: {

              title: 'Timestamp'

          },
```

```
          vAxis: {
              title: 'Humidity (%)'
          },
          colors: ['#007BFF'],
          chartArea: { width: '80%', height: '70%' }
        };

        var chart = new google.visualization.LineChart(document.getElementById('humidity_chart'));

        chart.draw(data, options);
      }

      function drawTemperatureChart() {
        var data = google.visualization.arrayToDataTable([
          ['Timestamp', 'Temperature'],
          <?php
                $query = "SELECT TIME(Timestamp) AS Time, Temperature FROM sensordata ORDER BY Timestamp LIMIT 20";
                $res = mysqli_query($conn, $query);
                if ($res) {
                    while($data = mysqli_fetch_assoc($res)){
                        $timestamp = $data["Time"];
                        $temperature = $data["Temperature"];
                        echo "['$timestamp', $temperature],";
```

```
                    }

                }

            ?>

        ]);


        var options = {

            title: 'Temperature Data',

            curveType: 'function',

            legend: { position: 'bottom' },

            hAxis: {

                title: 'Timestamp'

            },

            vAxis: {

                title: 'Temperature (°C)'

            },

            colors: ['#FF0000'], // Colore rosso per la temperatura

            chartArea: { width: '80%', height: '70%' }

        };


        var chart = new google.visualization.LineChart(document.getElementById('temperature_chart'));


        chart.draw(data, options);

      }

    </script>
```

```
</head>

<body>

    <nav class="navbar">

        <div class="container">

            <h1 class="title">Arduino Sensor Data</h1>

            <div class="sensor-info">

                <p>Humidity: <?php echo $latest_humidity; ?> %</p>

                <p>Temperature:  <?php  echo  $latest_temperature;  ?>
°C</p>

            </div>

        </div>

    </nav>

    <div class="container">

        <div  id="humidity_chart"  style="height:  300px;"></div>  <!--
Grafico per l'umidità -->

        <div  id="temperature_chart"  style="height:  300px;"></div>  <!--
Grafico per la temperatura -->

    </div>

</body>

</html>
```

file style.css

```
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
}

.navbar {
    background-color: #007BFF;
    color: white;
    padding: 10px 0;
    text-align: center;
}

.navbar .container {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
}

.navbar .title {
```

```
    margin: 0;
}

.container {
    max-width: 800px;
    margin: 20px auto;
    padding: 20px;
    background-color: #fff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    text-align: center;
}

.sensor-info {
    text-align: left; /* Allineati a sinistra */
    margin-bottom: 20px;
}

.sensor-info p {
    font-size: 18px;
    color: #333;
    margin: 5px 0; /* Aggiunto margine per separare i valori */
}

#curve_chart {
```

```
    margin: 0 auto; /* Centrato orizzontalmente */
```

Documentazione scritta da Ostoni Christian

Progetto realizzato da:

Ostoni Christian, Blasio Lorenzo, Vitale Gioacchino(4A inf)

Gabriele Lodato, Spisso Luigi, Antonio Maiorino(5A inf)

Bruno, Alfonso Giulietti, Ivan Andrusenko(4B inf)