**A logo for a university

Description automatically generatedبسم الله الرحمن الرحيم**

**MIDDLE EAST UNVERSITY**

**Collage: information technology**

**Specialization: Computer Science**

**Supervised By Dr:**

**Laith W. AL Shehab**

**Prepared by:**

**Shams Al-Din Abdin**

**201810454**

**Mohammad Abu Gharbiyya**

**201910372**

**2024م**

**1 Problem Statement:**

**In contemporary living spaces, maintaining an optimal indoor environment is crucial for comfort, health, and energy efficiency. Temperature and humidity levels play pivotal roles in achieving this balance. The objective of this project is to develop a Smart Temperature and Humidity Monitoring System that provides real-time information about the indoor climate and alerts users to potential issues.**

**Project Objectives:**

**1-Real-time Monitoring:**

* **Design a sensor system capable of accurately measuring and monitoring both temperature and humidity levels in real-time.**

**2-User-Friendly Output:**

* **Create a user-friendly output mechanism to display current temperature and humidity readings. Consider using an easily readable display, such as an LCD or LED matrix.**

**3-Alert Mechanism:**

* **Implement an alert system to notify users when the temperature or humidity exceeds predefined thresholds. Alerts could be visual (LED indicators) or audible (buzzer), providing immediate feedback.**

**4-Remote Access (Optional):**

* **Explore options for remote access to the monitoring system, enabling users to check temperature and humidity levels through a mobile app or a web interface.**

**5-Problem Detection and Notification:**

* **Develop an algorithm to detect potential issues related to extreme temperature or humidity variations. When an issue is detected, the system should send notifications to users, indicating the nature of the problem.**

**6-Energy Efficiency:**

* **Implement power-saving features to ensure the system operates efficiently. Consider options such as low-power modes or automatic shut-off during periods of inactivity.**

**2 Methodology:**

**Methodology for Smart Temperature and Humidity Monitoring System**

**1. Define Project Scope and Objectives:**

* **Clearly outline the scope of the project, specifying the target environment and the desired outcomes. Define the key objectives, such as real-time monitoring, alert mechanisms, and optional features like remote access.**

**2. Select Hardware Components:**

* **Choose appropriate temperature and humidity sensors, a microcontroller (e.g., Arduino or Raspberry Pi), and any additional components such as a display, alert system (LEDs, buzzer), and power supply.**

**3. Design Sensor Integration:**

* **Connect the temperature and humidity sensors to the microcontroller. Ensure proper wiring and communication between the components. Implement a calibration process if required for accurate readings.**

**4. User Interface Design:**

* **Develop the user interface for displaying temperature and humidity readings. Choose a suitable display technology (e.g., LCD, LED matrix) that is easy to read and understand.**

**5. Alert Mechanism Implementation:**

* **Integrate an alert system to notify users when the temperature or humidity exceeds predefined thresholds. Implement visual and/or audible alerts for immediate user attention.**

**6. Remote Access (Optional):**

* **If remote access is part of the project, integrate wireless connectivity (e.g., Wi-Fi) and design a user-friendly mobile app or web interface for users to monitor readings remotely.**

**7. Problem Detection Algorithm:**

* **Develop an algorithm to detect potential issues based on temperature and humidity variations. Define thresholds for normal, warning, and critical levels. Implement a notification system to inform users about detected problems.**

**8. Data Logging (Optional):**

* **If including a data logging feature, set up a mechanism to store historical temperature and humidity data. Choose a storage solution such as an SD card or cloud-based storage.**

**9. Power Management:**

* **Implement power-saving features to optimize energy consumption. Consider sleep modes for the microcontroller and sensors during periods of inactivity.**

**10. Enclosure Design:**

* **Design a protective and weather-resistant enclosure for the sensor system. Consider the environmental conditions of the deployment location.**

1. **Design:**
2. **ESP32:**

**A close-up of a microchip

Description automatically generated**

1. **DHT Sensor:**

**A blue and black electronic device

Description automatically generated**

1. **Jumper wires:**

**A close-up of a cable

Description automatically generated**

1. **Bread Board:A close-up of a white breadboard

   Description automatically generated**
2. **Contribution:**

Shamsaldin:

worked, methodology & contribution, problem and statement & design and made the presentation.

Mohammad:

Project Work He purchased the project pieces and put them together.

**CODE:**

#include <WiFi.h>

#include <WiFiClient.h>

#include <WebServer.h>

#include <ESPmDNS.h>

#include <DHT.h>

const char \*ssid = "Mohammad’s iphone";

const char \*password = "12345678";

WebServer server(80);

DHT dht(26, DHT11);

void handleRoot() {

  char msg[1500];

  snprintf(msg, 1500,

           "<html>\

  <head>\

    <meta http-equiv='refresh' content='4'/>\

    <meta name='viewport' content='width=device-width, initial-scale=1'>\

    <link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.2/css/all.css' integrity='sha384-fnmOCqbTlWIlj8LyTjo7mOUStjsKC4pOpQbqyi7RrhN7udi9RwhKkMHpvLbHG9Sr' crossorigin='anonymous'>\

    <title>ESP32 DHT Server</title>\

    <style>\

    html { font-family: Arial; display: inline-block; margin: 0px auto; text-align: center;}\

    h2 { font-size: 3.0rem; }\

    p { font-size: 3.0rem; }\

    .units { font-size: 1.2rem; }\

    .dht-labels{ font-size: 1.5rem; vertical-align:middle; padding-bottom: 15px;}\

    </style>\

  </head>\

  <body>\

      <h2>ESP32 DHT Server!</h2>\

      <p>\

        <i class='fas fa-thermometer-half' style='color:#ca3517;'></i>\

        <span class='dht-labels'>Temperature</span>\

        <span>%.2f</span>\

        <sup class='units'>&deg;C</sup>\

      </p>\

      <p>\

        <i class='fas fa-tint' style='color:#00add6;'></i>\

        <span class='dht-labels'>Humidity</span>\

        <span>%.2f</span>\

        <sup class='units'>&percnt;</sup>\

      </p>\

  </body>\

</html>",

           readDHTTemperature(), readDHTHumidity()

          );

  server.send(200, "text/html", msg);

}

void setup(void) {

  Serial.begin(115200);

  dht.begin();

  WiFi.mode(WIFI\_STA);

  WiFi.begin(ssid, password);

  Serial.println("");

  // Wait for connection

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

    Serial.print(".");

  }

  Serial.println("");

  Serial.print("Connected to ");

  Serial.println(ssid);

  Serial.print("IP address: ");

  Serial.println(WiFi.localIP());

  if (MDNS.begin("esp32")) {

    Serial.println("MDNS responder started");

  }

  server.on("/", handleRoot);

  server.begin();

  Serial.println("HTTP server started");

}

void loop(void) {

  server.handleClient();

  delay(2);//allow the cpu to switch to other tasks

}

float readDHTTemperature() {

  // Sensor readings may also be up to 2 seconds

  // Read temperature as Celsius (the default)

  float t = dht.readTemperature();

  if (isnan(t)) {

    Serial.println("Failed to read from DHT sensor!");

    return -1;

  }

  else {

    Serial.println(t);

    return t;

  }

}

float readDHTHumidity() {

  // Sensor readings may also be up to 2 seconds

  float h = dht.readHumidity();

  if (isnan(h)) {

    Serial.println("Failed to read from DHT sensor!");

    return -1;

  }

  else {

    Serial.println(h);

    return h;

  }

}