Servers-Eco System

Key words: Data Monitoring, Servers, Data visualization, Battery banks, Solar cell, Correlation

Ahmed Mohamed El Sayed
Mohamed Ashraf Hussien
Mohamed Abdelaty

12326

Sensor Integration

The system contains two DHT11 temperature and humidity sensors, a sound sensor, an ESP32 microcontroller, and LEDs. Two DHT11 sensors measure environmental temperature and humidity. The sound sensor measures noise levels, identifying any damage to the servers. Communication between sensors and the ESP32 utilizes standard protocols like I2C and UART, ensuring reliable data acquisition.

To implement this system,

the following setup was used:

- Wi-Fi Module: The ESP32 microcontroller connects to a Wi-Fi network using the wifi.
- DHT11 Temperature and Humidity Sensors: The temperature sensor is connected to GPIO 4, and the humidity sensor to GPIO 5. Both are managed using the DHTesp library.
- Sound Sensor: The sound sensor is connected to GPIO 34 and provides digital input values for noise detection.
- LED Indicators: LEDs on GPIO 18, 32, and 33 signal conditions for temperature, humidity, and sound, respectively.

Code Implementation The system's functionality is encapsulated in the following features:

- Reading data from the DHT11 sensors and sound sensor.
- Controlling LEDs based on sensor limit (temperature ≥ 21.6°C, humidity > 43%, sound level ≥ 1).
- Sending data to a Python server via HTTP POST requests in JSON format.

---

2. Signal Interpretation Techniques

2.1 Data of Sensor signals are sampled every 15 seconds, balancing real-time monitoring with energy efficiency. This interval is designed to minimize power consumption while measuring meaningful variations in environmental data.

2.2 Data Analysis Correlations between temperature, humidity, and sound levels are evaluated to predict data. Sound intensity is calibrated to know when a component in the server is damaged.

Code Implementation

- The system reads temperature and humidity data using the getTempAndHumidity method from the DHTesp library.
- The sound sensor provides a digital value.
- Processed data is packaged into a JSON payload and sent to a Python server for storage and further analysis.

---

3. System Performance in a Test Environment

3.1 Test Setup The system was deployed in a controlled environment (server room simulation), replicating operational conditions. Variables such as temperature, humidity, and sound levels were manipulated to assess system adaptability. The system used the following network and hardware setup:

- Wi-Fi SSID: "ASHRAF123"
- Python Server URL: http://192.168.137.1:5000/data

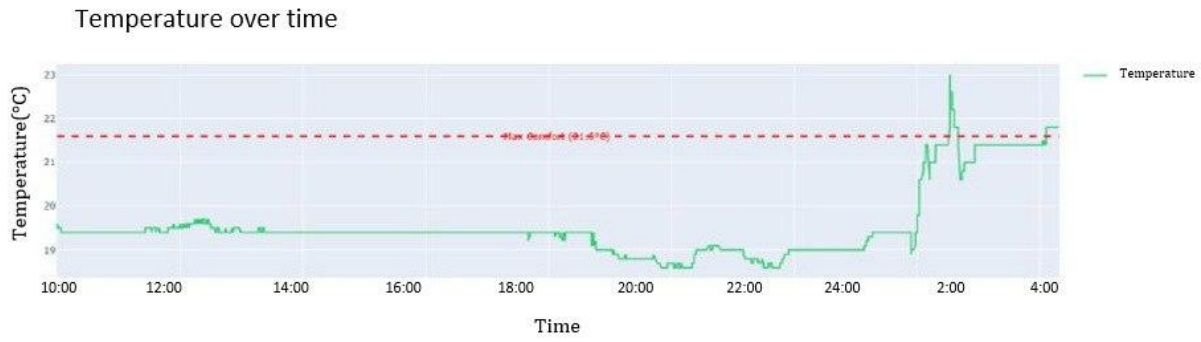
3.2 Performance Metrics

- Accuracy: Temperature measurements were accurate within ±1°C, and humidity readings maintained a ±5% RH precision. Sound sensor outputs were reliable for qualitative analysis.
- Response Time: Data visualization achieved a delay of less than 30 seconds (15 seconds), meeting design objectives.
- Power Efficiency: The system power system (battery bank and the solar panel for charging them) with a six-month battery life goal under continuous operation.

Code Implementation

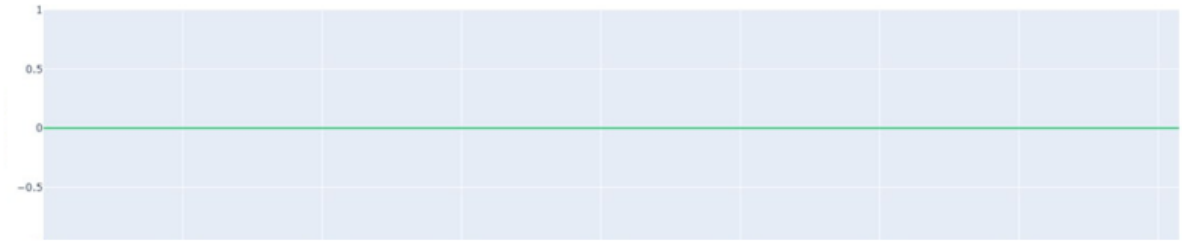
- The system connects to Wi-Fi and checks connectivity before sending data.
- HTTPClient library manages POST requests, effectively handling server responses and error cases.

Data collected by the system.



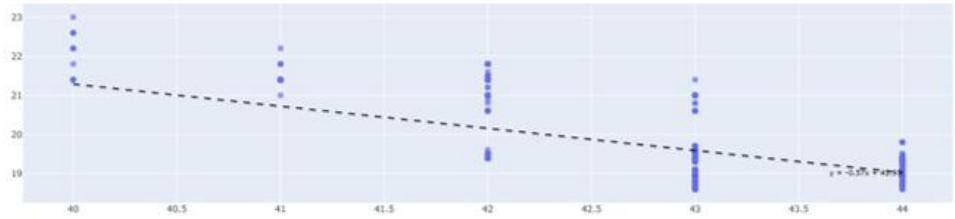
The graph above shows temperature sensor readings.

The graph above shows humidity over an interval of time.



Representation of sound with respect to time

The graph above shows the reading of the sound sensor.



: Humidity vs temperature with correlation lines

The graph above shows the direct relation between both temperature and humidity.