PROJECT REPORT

IoT-Based Temperature and Humidity Monitoring using Arduino and ESP8266

NAME : MOHAMMED MOKHTAR ELBARBER

Embedded Systems Summer Training Report

IoT-Based Temperature and Humidity Monitoring Using Arduino and ESP8266

TEAM MEMBERS :

محمد مختار فتحي سالم البربير IS

عبد الرحمن مسعد سعد الكامل CS

خالد أحمد محمد النيلي CS

Supervised By :

PROF / DR TAMER EMARA

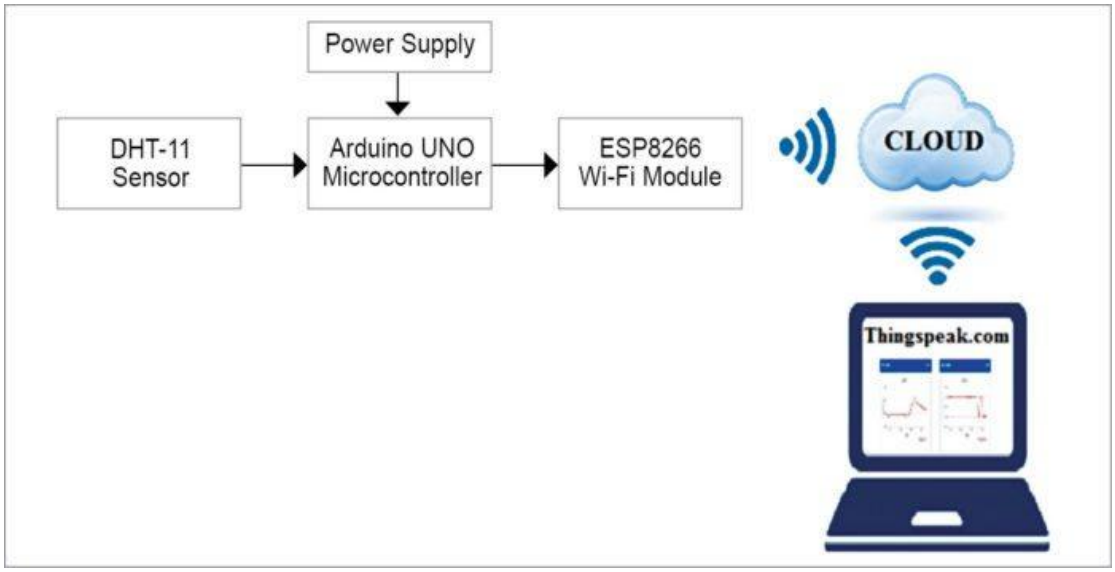
Abstract:

This project presents the design and implementation of an IoT-based system for monitoring temperature and humidity using the Arduino UNO, ESP8266 Wi-Fi module, and DHT11 sensor.

By this method, we can monitor our DHT11 sensor's temperature and humidity data over the internet using the ThingSpeak IoT server. And we can view the logged data and graph overtime on the Thingspeak website.

The main objective was to create a cost-effective and efficient system for continuous monitoring, which can be used in various applications such as smart homes, greenhouses, and industrial environments. The project successfully demonstrates the integration of IoT technologies for environmental monitoring.

ThingSpeak is an open data platform for monitoring your data online where you can set the data as private or public according to your choice.



Introduction

Problem Statement:

The need for a reliable and low-cost solution to monitor temperature and humidity data in real-time is evident in many environments, including smart homes, greenhouses, and server rooms. The traditional methods of monitoring are often inefficient, labor-intensive, and prone to errors. This project aims to address these issues by creating an automated IoT-based monitoring system that uploads data to the cloud for real-time access and analysis.

Objectives:

- To design and implement a temperature and humidity monitoring system using Arduino UNO and ESP8266.
- To upload the collected data to ThingSpeak for real-time monitoring and visualization.
- To create a low-cost, scalable solution for various applications that require environmental monitoring.

Materials and Methods

Software Used:

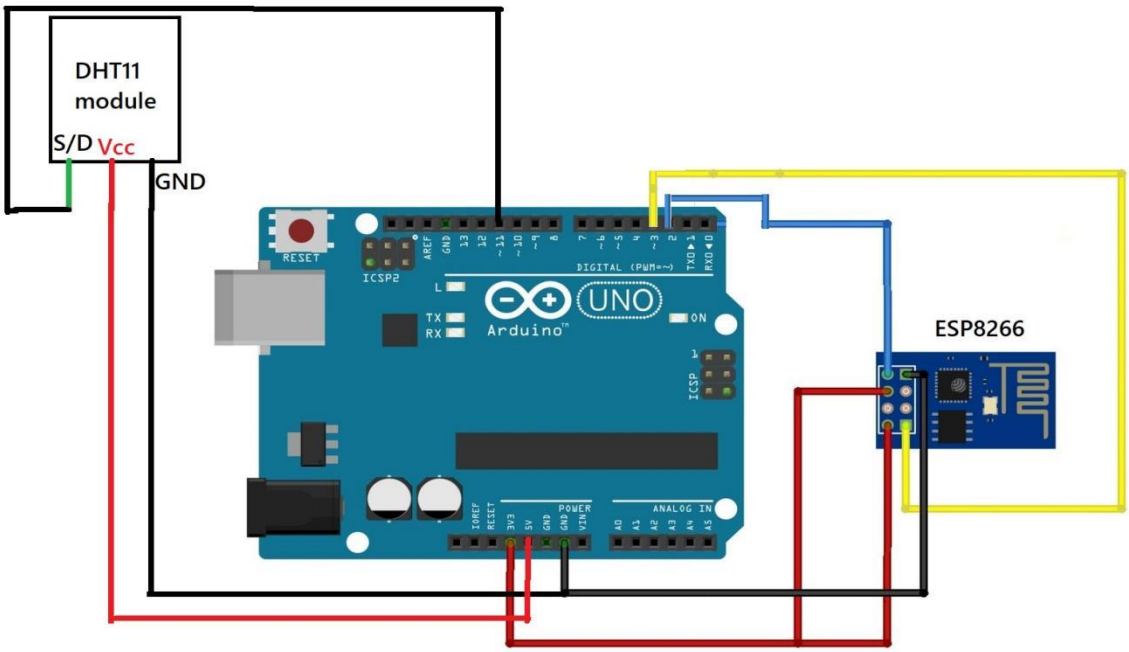
- Arduino IDE: For coding and uploading the sketch to the Arduino UNO.
- ThingSpeak: IoT platform used for data storage and visualization.
- DHT and ESP8266 Libraries: Arduino libraries used to interface with the DHT11 sensor and ESP8266 module.

Components Used:

- Arduino UNO: Microcontroller used to read sensor data and control the ESP8266 module.
- ESP8266 Wi-Fi Module: Enables wireless communication between the Arduino and ThingSpeak.
- DHT11 Sensor: Measures temperature and humidity.
- Breadboard and Jumper Wires: Used for connecting the components.
- USB Cable: For powering the Arduino and uploading code.

Implementation

Temperature and Humidity Monitoring System Circuit Diagram :



System Design:

The system is built around an Arduino UNO, an ESP8266 Wi-Fi module, and a DHT11 sensor. The DHT11 sensor measures the temperature and humidity, which are then processed by the Arduino UNO. The ESP8266 Wi-Fi module is used to transmit the data to the ThingSpeak platform via HTTP GET requests. The system is designed to operate in a loop, continuously sending sensor data to ThingSpeak at regular intervals.

Connections are given in below table:

S.NO.	Pin Name	Arduino Pin
1	ESP8266 VCC	3.3V
2	ESP8266 RX	Pin 3 TX
3	ESP8266 TX	Pin 2 RX
4	ESP8266 GND	GND
5	DHT-11 VCC	5V
6	DHT-11 Data	5
7	DHT-11 GND	GND

Coding:

The code initializes the ESP8266 module and connects it to a Wi-Fi network. It then reads temperature and humidity data from the DHT11 sensor and sends this data to ThingSpeak using an HTTP GET request. The sendCommand() function is used to send AT commands to the ESP8266 module, ensuring the correct setup and data transmission. The code handles reconnections and errors efficiently.

Project Code:

```
#include <SoftwareSerial.h>
#include "DHT.h"

#define RX 2
#define TX 3
#define dht_apin 11 // Pin sensor is connected to

// Initialize the DHT object with the pin and sensor type
DHT dhtObject(dht_apin, DHT11);

String AP = "MFM";       // AP NAME
String PASS = "998877**";         // AP PASSWORD
String API = "MIN5UIF5RLHDFB62";  // Write API KEY
String HOST = "api.thingspeak.com";
String PORT = "80";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
int valSensor = 1;

SoftwareSerial esp8266(RX, TX);

void setup() {
  Serial.begin(9600);
  esp8266.begin(115200);

  // Initialize the DHT sensor
  dhtObject.begin();

  sendCommand("AT", 5, "OK");
  sendCommand("AT+CWMODE=1", 5, "OK");
  sendCommand("AT+CWJAP=\"" + AP + "\",\"" + PASS + "\"", 20, "OK");
}

void loop() {
  String getData = "GET /update?api_key=" + API + "&field1=" + getTemperatureValue() +
"&field2=" + getHumidityValue();
  sendCommand("AT+CIPMUX=1", 5, "OK");
```

```
  sendCommand("AT+CIPSTART=0,\"TCP\",\"" + HOST + "\"," + PORT, 15, "OK");
  sendCommand("AT+CIPSEND=0," + String(getData.length() + 4), 4, ">");
  esp8266.println(getData);
  delay(1500);
  countTrueCommand++;
  sendCommand("AT+CIPCLOSE=0", 5, "OK");
}

String getTemperatureValue() {
  float temperature = dhtObject.readTemperature();
  Serial.print(" Temperature(C)= ");
  Serial.println(temperature);
  return String(temperature);
}

String getHumidityValue() {
  float humidity = dhtObject.readHumidity();
  Serial.print(" Humidity in %= ");
  Serial.println(humidity);
  return String(humidity);
}

void sendCommand(String command, int maxTime, char readReplay[]) {
  Serial.print(countTrueCommand);
  Serial.print(". at command => ");
  Serial.print(command);
  Serial.print(" ");
  while (countTimeCommand < (maxTime * 1)) {
    esp8266.println(command);
    if (esp8266.find(readReplay)) {
      found = true;
      break;
    }
    countTimeCommand++;
  }

  if (found == true) {
    Serial.println("DONE");
    countTrueCommand++;
    countTimeCommand = 0;
  } else {
    Serial.println("Fail");
    countTrueCommand = 0;
    countTimeCommand = 0;
  }

  found = false;
}
```

ThingSpeak Setup for Temperature and Humidity Monitoring

To effectively monitor and visualize temperature and humidity data, ThingSpeak was utilized as the IoT platform. The following steps were taken to set up ThingSpeak for this project:

1. **Create a ThingSpeak Account:**
Visit ThingSpeak.com and create an account. If you already have an account, log in using your credentials.

2. **Create a New Channel:**
After logging in, navigate to the "Channels" tab and click "New Channel."
Provide a name for the channel, such as "Temperature and Humidity Monitoring."
Enable two fields by checking the boxes:
Field 1: Temperature (°C)
Field 2: Humidity (%)
Save the channel settings.

3. **Obtain the Write API Key:**
After creating the channel, go to the "API Keys" tab. Copy the Write API Key, which is required for sending data from the Arduino to ThingSpeak.

4. **Configure the Arduino Code:**
Replace the placeholder API key in your Arduino code with the Write API Key from your ThingSpeak channel.
Ensure the Wi-Fi SSID and password match your network settings.

5. **Visualize the Data:**
Once the code is running and data is being sent to ThingSpeak, you can monitor the data on the "Private View" tab of your channel.

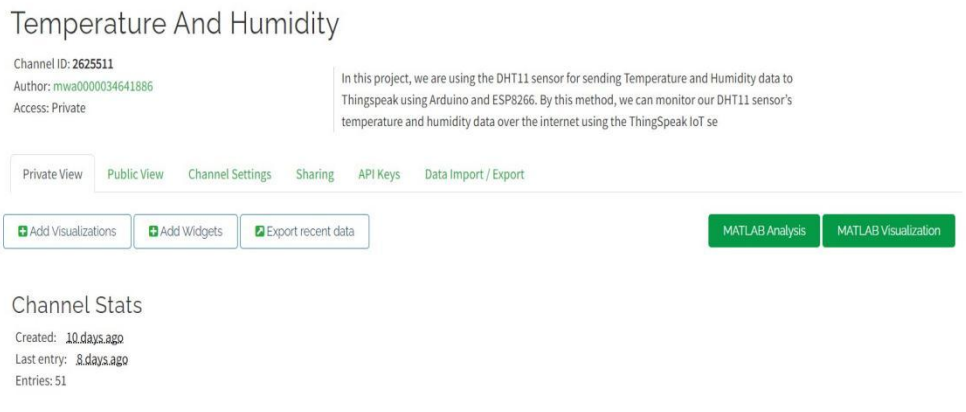
ThingSpeak will automatically generate graphs for both temperature and humidity, allowing real-time monitoring and analysis of trends.

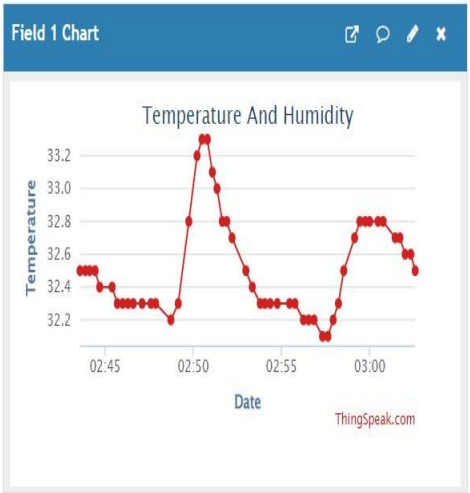
Results:

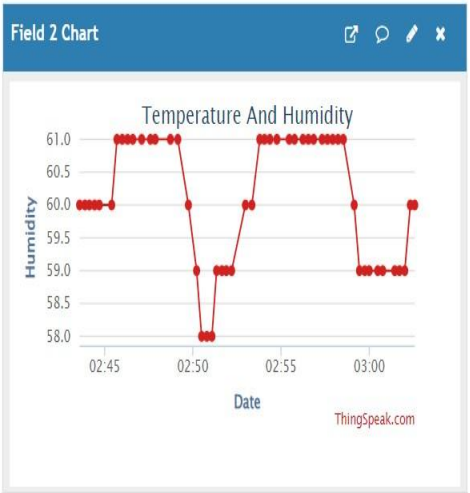
Data Visualization:

The data was successfully sent to ThingSpeak, where it was visualized using dynamic graphs. The platform provided real-time updates of temperature and humidity, displaying trends over time. Below are some key observations:

Temperature Trends: The temperature showed expected variations based on the time of day, with higher temperatures during the day and lower at night.
Humidity Trends: Humidity levels were relatively stable, with minor fluctuations.







Conclusion:

Summary:

This project successfully demonstrated an IoT-based system for monitoring temperature and humidity using Arduino UNO, ESP8266, and ThingSpeak. The system provided real-time data collection, storage, and visualization, making it a valuable tool for various applications.

Challenges:

Some challenges were encountered during the project, such as configuring the ESP8266 module and ensuring stable Wi-Fi connectivity. These issues were resolved through troubleshooting and testing.

Significance:

The project offers a scalable and low-cost solution for environmental monitoring, with potential applications in smart homes, agriculture, and industrial settings.

Future Work

Improvements:

Future iterations of this project could include the following improvements:

- Battery Power: Implement a battery-powered solution for remote monitoring.
- Additional Sensors: Add sensors for monitoring other environmental parameters such as air quality or light intensity or buzzer for alerts.
- Data Analytics: Implement advanced data analytics and machine learning models for predictive analysis.

Additional Features:

- Mobile App Integration: Develop a mobile app for real-time alerts and monitoring.
- Data Logging: Store data locally on an SD card as a backup.

References:

- Arduino IDE Documentation.
https://www.arduino.cc/en/Guide/HomePage

- ThingSpeak API Documentation.
https://thingspeak.com/docs

- DHT11 Sensor Datasheet.
https://components101.com/sensors/dht11-temperature-humidity-sensor

- ESP8266 Wi-Fi Module Documentation.
https://www.espressif.com/en/products/socs/esp8266

- IOT DESIGN PRO Article
https://iotdesignpro.com/projects/temperature-humidity-monitoring-over-thingspeak-using-arduino-esp8266