IoT Based Smart Weather Monitoring System

Shrijan Rajbhandari

Bsc. (Hons) in Computer Science with Artificial Intelligence, Sunway College Kathmandu Birmingham City University, BCU Kathmandu – 14, Nepal

shrijan_24128470@sunway.edu.np

Abstract-The Smart Weather Monitoring System (SWMS) has been conceptualized to considerably bring in competence in collecting, processing, and use of real-time meteorological data for applications in different fields. In the aftermath of increasing awareness about climate change and its consequences, reliable and timely weather monitoring is emerging as very important. By integrating modern sensor technologies, IoT connectivity, and data analytics, SWMS provides accurate forecasts and insights into the weather. This approach provides conscious decision-making in the perspective of rapid environmental changes, thus enabling resilient community development combined with efficiency in risk management.

Keywords-IoT, Smart Weather Monitoring System, Arduino, Sensor

Contents

Introduction

IoT is an extended and an enlarged network-based system in the Internet, where, basically, it performs a number of modern technological techniques, especially to achieve the real-time interaction among objects, machines, and human beings. Weather conditions form the basis for many industries, ranging from agriculture to transport. Any unpredictability in weather tends to upset activities; hence, the need for effective weather monitoring systems. Using recent advancements in IoT, cost-effective real-time solutions for weather monitoring can be provided, offering better data accuracy and accessibility. . The Internet of Things (IoT) has been in existence roughly for almost twenty years and over time, A wide variety of industries have taken note or are aware of the concept of the Internet of Things including several academics, business executives, and government officials and are beginning to credit this technology for its essentiality in enhancing living conditions and standards of people (Wang et al., 2021). The protocols ensure proper dialogue between device interfaces. In the process, there is an exchange of global infrastructures with synchronous signals, actuator control, and sensor data. This paper describes the integration of various sensors in a weather monitoring system for measurement of rainfall, temperature, humidity, and pollution parameters. Bad weather affects transportation, industries, businesses, and day-to-day activities. To counter this, users are allowed to monitor the weather in real time. . Continuous monitoring across cities and nations generates vast amounts of data that require significant processing. Large data hulls might provide problems for system measurement and parameter assessment. A situation like this can negatively impact the measurement system's performance. Integrating cloud and internet-based measuring systems can improve data handling efficiency (Rao et al., 2020).

Literature Review

About IoT

- In modern times, the Internet of Things is among the most progressive technologies on Earth. The Internet of Things will be able to connect literally everything to the internet. IoT is foreseen to be one of the revolutions going to change our life. The industry of the so-called Internet of Things is presently on fire. According to the forecasts of analysts, a wide growth of IoT products and services is highly probable in the upcoming years. More than 50 billion devices are forecasted to be connected through IoT by 2025. It brings together several domains, including embedded systems, communication systems, sensors/actuators, the World Wide Web, and mobile apps. However, IoT continues to face many problems and limits owing to a variety of reasons that limit its full potential. Kevin Ashton of the United States pioneered the Internet of Things idea in 1999.

Global Internet of Things market expected to reach $318 billion (Mohamed, 2021). IoT is presently being utilized for applications ranging from weather prediction to health monitoring. The ability to handle large-scale agricultural output is highly dependent on weather forecasting. Several approaches have been designed based on different types of mathematical and statistical models with the purpose of weather condition forecast.

Innovations In IoT

- IoT technologies also disrupt entire industries. The trends at the forefront in this regard include smart products, the demurring of boundaries between industries, a rush of new business models, and changing regulations. Similarly, rapid digital change and newer competitors also jeopardize traditional businesses. To deal with disruption problems, incumbents need to adopt next generation technologies that allow the development of new value propositions and business models, as well as the development of new organizational skills, to survive and prosper in an unpredictable and continuously changing environment (Sabin and Glovatchi., 2021).

Security

- From smartphones and tablet computers to TVs, vehicles, and even smart home appliances, IoT technologies are becoming an integral part of both our public and private lives. In many instances, these advantages come hand in glove with immense security risks, usually ignored or dismissed. Quite surprisingly, the threat landscape for IoT involves several hardware and software components that make it wide-ranging and complex. Security in such architecture thus becomes the centerpiece of any interaction by applications and services running at various IoT devices and cloud/edge infrastructure. IoT offers benefits, but also raises security risks. Existing security methods and procedures are inadequate for IoT networks due of their heterogeneity. IoT devices and the data they transfer are vulnerable to various dangers (Dhar and Bose. 2020).

Present & Future Trends In IoT

- Among the emerging technologies, perhaps the IoT represents the most promising in this information age. IoT provides pervasive data gathering and network connectivity, bringing considerable and needed ease and insight to everyday living and industrial activities. However, IoT still faces a variety of obstacles and issues which should be addressed immediately. In fact, there are many problems that IoT infrastructures are facing, such as counterfeit hardware, software flaws, communication security, system management difficulties, and data privacy concerns. Meanwhile, blockchain, as a new information technology, has gained much attention due to the decentralization, transparency, and security of the information contained in it, and great potential has been demonstrated. The features of blockchain seem to fit in nicely with IoT, and the use of blockchain in the context of IoT addresses some of the issues listed above.

Methodology

Objective of the project

- This solution will drive the development of an integrated real-time environmental monitoring solution that will be deployed using varied sensors for continuous assessment and reporting of critical weather parameters, including temperature, humidity, rainfall, and air quality in terms of CO2 levels. By implementing this system, we aim to:-

    1. **Improvement in Awareness**: Having proper knowledge of the weather in due time helps people make informed decisions on daily activities, traveling, and agricultural practices.

    2. **Improve Safety**: The detection of adverse weather conditions of high humidity or extreme temperatures or high levels of pollution through warnings will reduce health risks and improve safety.

    3. **Data Collection and Analysis**: A large quantum of environmental data is collected that would allow analysis with respect to weather trends and patterns, which would support research and predictive modeling.

    4. **User-Friendly Interface**: Provide an easy-to-use and intuitive interface for the user to visualize in real time through data and alerts via a buzzer or a display for being more proactive in the management of weather.

    5. **Remote Monitoring**: Let the implementation of remote monitoring capabilities be accessed anywhere so that users are prepared for any variable weather conditions.

    6. **Environmental Awareness**: Educate on the impact of weather and air quality on health and the environment to aid responsible behavior and practices.

By realizing these objectives, the Smart Weather Monitoring System shall support improving the quality of life, safety, and sustainability in such sectors as agriculture, transportation, and public health.

Proposed System

- The Smart Weather Monitoring System is able to monitor real-time environmental parameters with the integration of various sensor technologies and microcontrollers. The Arduino UNO microcontroller will be controlled by an ATmega328P at the heart of it, gathering data from various sensors, including temperature, humidity, rainfall, and CO2 sensors, with one ultrasonic sensor for detecting the distance. The system is integrated with an LCD for displaying current weather data and warnings. This display will work along with a buzzer that provides audible notification for critical, dangerous weather conditions. Power for it can either come through a battery or by using DC power supply for continued indoor and outdoor operations. The system embeds a wireless module that would avail remote monitoring and access to data through a mobile application or web interface. The system carries out continuous collection, processing of data from sensors, and real-time data analysis to trigger an alert when certain thresholds have been exceeded. Data logging further contributes to the analysis of historical weather trends, upon which informed decisions can be made by the end user. The broad vision of the Smart Weather Monitoring System is to enhance environmental awareness and living for safety, comfort, and convenience with ease of operation, thereby equipping individuals with the skills to deal successfully with inclement weather.

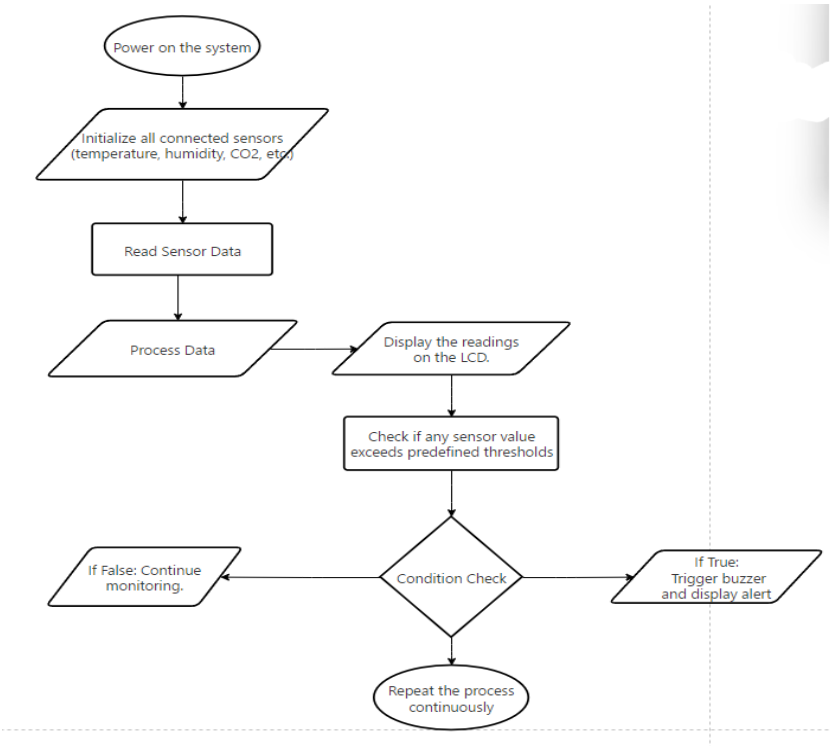

Fig: Flowchart diagram of the system

Logic of System

1. **Inputs:**
    - **T**: Temperature Sensor (1 if high, 0 if normal)
    - **H**: Humidity Sensor (1 if high, 0 if normal)
    - **C**: CO2 Sensor (1 if high, 0 if normal)
    - **U**: Ultrasonic Sensor (1 if obstacle detected, 0 if clear)
2. **Outputs:**
    - **B**: Buzzer (1 if triggered, 0 if not)
    - **D**: LCD Display (1 if displaying data, 0 if not displaying)

Truth Table:

T	H	C	U	B (Buzzer)	D (LCD Display)
0	0	0	0	0	1
0	0	0	1	0	1
0	0	1	0	1	1
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	1	1
1	1	0	0	1	1
1	1	0	1	1	1
1	1	1	0	1	1
1	1	1	1	1	1

Explanation:

- **Buzzer (B)** will trigger (1) if either temperature or CO2 levels are high (1).
- The **LCD Display (D)** will always be on (1) if at least one of the sensors is reading, but will display data only when conditions are normal.
- When all sensors are at normal levels (0), the LCD can still display a message, but the buzzer is off.

Logic Circuit

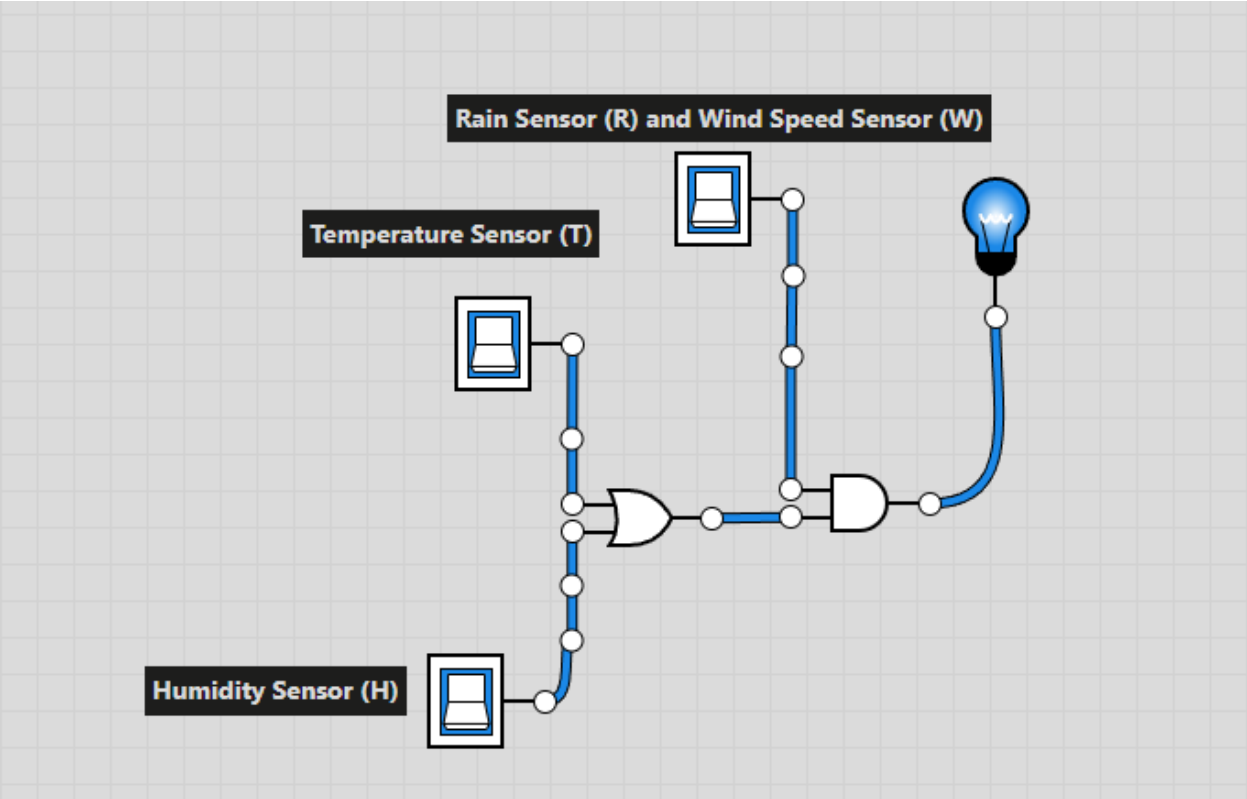

Fig: Logic circuit of the system

Hardware Required

Microcontroller:

- **Arduino UNO**: This is the central unit that gathers information from sensors and drives the system.

Sensors:

- **Temperature Sensor**: This measures ambient temperature, which could be any: DHT11 or DS18B20.
- **Humidity Sensor**: This measures air humidity-in most instances, this comes integrated with the temperature sensors: DHT11.

- **Rainfall Sensor**: This detects rainfall. Rain gauge or a simple rain sensor module

- **CO2 sensor**: This reads the air quality in terms of Carbon Dioxide: MH-Z19.

Distance Measurement

- Ultrasonic Sensor, for example, HC-SR04, which is useful in the detection of obstacles or features of the environment.

Output Modules:

- **LCD Display**-to visualize real-time data; 16x2 LCD
- **Buzzer**-to provide an audible notification in case of critical weather conditions

Power Supply:

- **Battery or DC Power Supply**- This system should be running continuously. For portable systems, batteries may be rechargeable.

Communication Module:

- **Wi-Fi Module**: ESP8266 or ESP32-This module will help in wireless connectivity in order to make the system monitor and send data remotely.
- **Bluetooth Module**-e.g., HC-05: A substitute for a microcontroller in cases of communication with mobile devices over a small radius.

Breadboard and Jumper Wires:

- **Breadboard**: A development board on which prototyping is done very fast, involving the assembly and connecting of parts without soldering.
- **Jumper Wires**: These connect the microcontroller to sensors and output modules.

Resistors and Capacitors:

- **Resistors**: They are used to limit the current or pull up/down signals as might be required by individual sensors.
- **Capacitors**: They stabilize the signal, especially in the case of analog sensor circuits.

Miscellaneous:

- **LEDs**: Optional-use for a visual indication of various statuses or alerts
- **Real-Time Clock Module**: To provide timestamping on data to show more accurate tracking (optional)

TinkerCAD

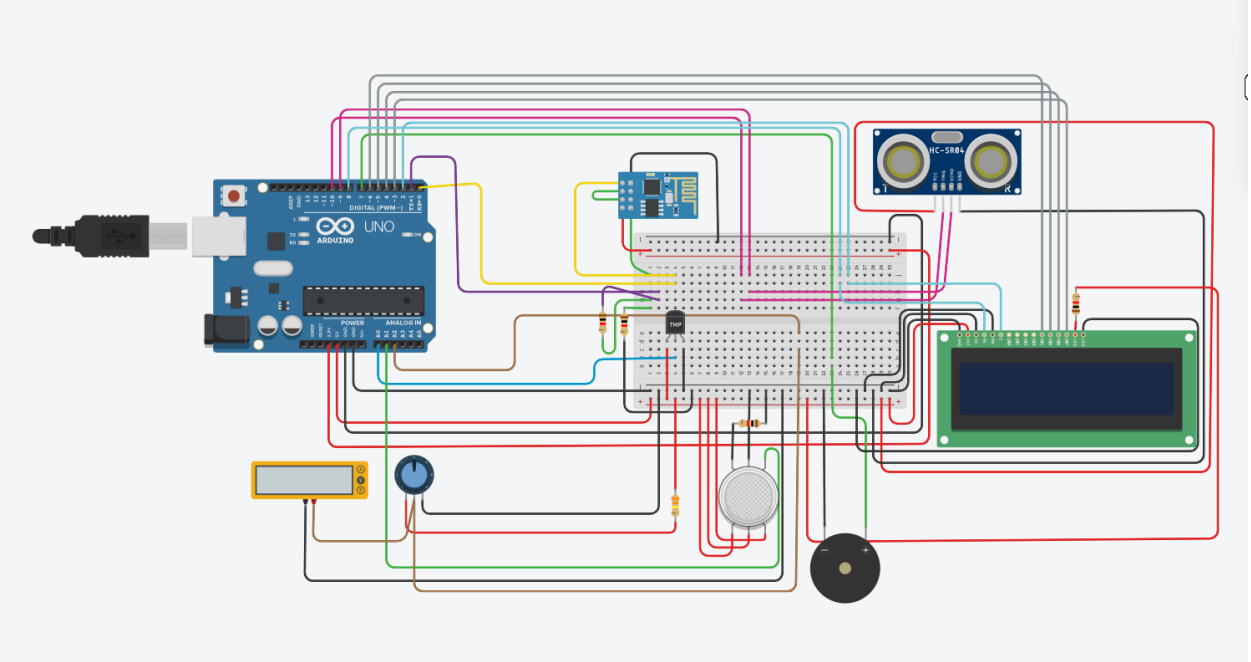

Fig: TinkerCAD of Smart Weather Monitoring System

Block Diagram

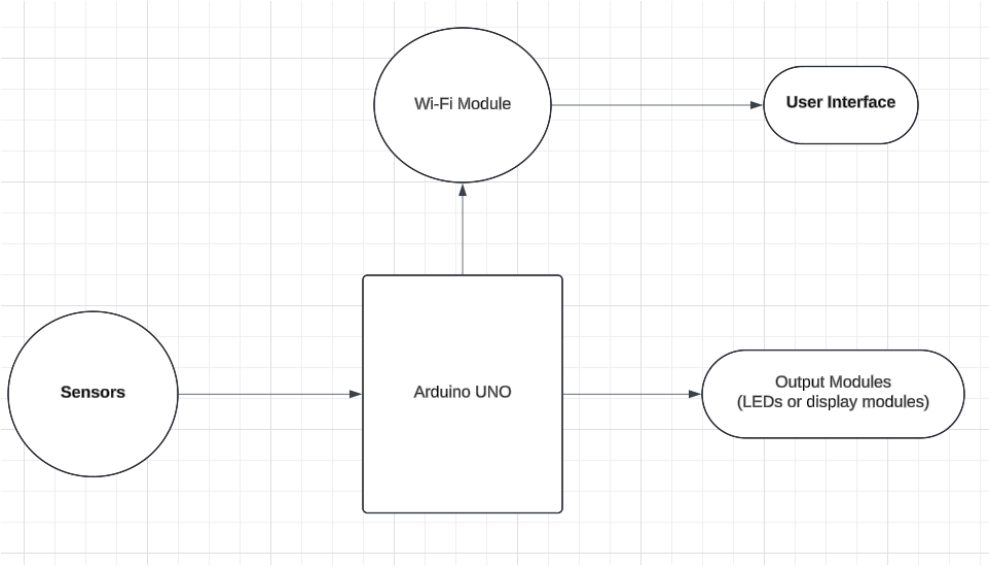

Fig: Block Diagram of the system

Code

MATLAB:-

```
readChannelID = 1480637;

% Field ID

TemperatureFieldID = 1;

gas_sensorFieldID = 2;

rainfall_FieldID = 3;

wind_FieldID = 4;



readAPIKey = 'Q90LQOIGATHEW7SP';



[tempF,timeStamp1] = thingSpeakRead(readChannelID,'Fields',TemperatureFieldID, ...
                                    'numDays',1,'ReadKey',readAPIKey);
[pollutionF,timeStamp2] = thingSpeakRead(readChannelID,'Fields',gas_sensorFieldID, ...
                                    'numDays',1,'ReadKey',readAPIKey);
[rainfallF,timeStamp3] = thingSpeakRead(readChannelID,'Fields',rainfall_FieldID, ...
                                    'numDays',1,'ReadKey',readAPIKey);
[windF,timeStamp4] = thingSpeakRead(readChannelID,'Fields',wind_FieldID, ...
                                    'numDays',1,'ReadKey',readAPIKey);

[maxTempF,maxTempIndex] = max(tempF); % Calculate the maximum and minimum temperatures

[minTempF,minTempIndex] = min(tempF);

[maxpollutionF,maxpollutionIndex] = max(pollutionF); % Calculate the maximum and minimum Pollution

[minpollutionF,minpollutionIndex] = min(pollutionF);
```

```
[maxrainfallF,maxrainfallIndex] = max(rainfallF); % Calculate the maximum and minimum rainfall

[minrainfallF,minrainfallIndex] = min(rainfallF);


[maxwindF,maxwindIndex] = max(windF); % Calculate the maximum and minimum rainfall

[minwindF,minwindIndex] = min(windF);


timeMaxTemp = timeStamp1(maxTempIndex); % Select the timestamps at which the maximum and
minimum temperatures were measured

timeMinTemp = timeStamp1(minTempIndex);


timeMaxpollution = timeStamp2(maxpollutionIndex); % Select the timestamps at which the maximum
and minimum pollution were measured

timeMinpollution = timeStamp2(minpollutionIndex);


timeMaxrainfall = timeStamp3(maxrainfallIndex); % Select the timestamps at which the maximum and
minimum rainfall were measured

timeMinrainfall = timeStamp3(minrainfallIndex);


timeMaxwind = timeStamp4(maxwindIndex); % Select the timestamps at which the maximum and
minimum wind were measured

timeMinwind = timeStamp4(minwindIndex);


display(maxTempF,'Maximum Temperature for the past 24 hours is');

display(minTempF,'Minimum Temperature for the past 24 hours is');


display(maxpollutionF,'Maximum pollution for the past 24 hours is');

display(minpollutionF,'Minimum pollution for the past 24 hours is');


display(maxrainfallF,'Maximum rainfall for the past 24 hours is');
```

```
display(minrainfallF,'Minimum rainfall for the past 24 hours is');

display(maxwindF,'Maximum wind for the past 24 hours is');
display(minwindF,'Minimum wind for the past 24 hours is');

  writeChannelID = [1480637];
writeAPIKey = 'O1WB76S0Z5G24Y0O';
```

Arduino code:-

```
//Smart weather reporting system
#include <LiquidCrystal.h>

//temperature sensor variables
float temp_vout;
float temp;
float voltage;

//gas sensor variables
int gas_sensor_port = A1;
int gas_sensor_value = 0;

//rainfall measurement variables
float rain;
```

```
const int triggerPin = 10;

const int echoPin = 9;

long duration;


//wind speed measurement variables

float V_wind = 0;

float Windspeedfloat;

int Windspeedint;

//LCD

LiquidCrystal lcd(8, 2, 6, 5, 4, 3); //Parameters: (rs, enable, d4, d5, d6, d7)


//WIFI module variables

String ssid     = "Simulator Wifi";  // SSID to connect to

String password = "";   //virtual wifi has no password

String host     = "api.thingspeak.com"; // Open Weather Map API

const int httpPort   = 80;

String url     = "/update?api_key=O1WB76S0Z5G24Y0O&field1="; //ThingSpeak Channel API Key



//setting up the wifi module

void setupESP8266(void)

{

  // Start our ESP8266 Serial Communication
```

```
 Serial.begin(115200);   // Baud rate

 Serial.println("AT");   // Serial connection on Tx / Rx port to ESP8266

 delay(10);             // Wait a little for the ESP to respond


 if (Serial.find("OK"))

   Serial.println("ESP8266 OK!!!");  // Connect to Simulator Wifi


 Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\""); //AT+CWLAP – list nearby
available WiFi networks

 delay(10);                    // Wait a little for the ESP to respond


 if (Serial.find("OK"))

   Serial.println("Connected to WiFi!!!");     // Open TCP connection to the host:


 //ESP8266 connects to the server as a TCP client.


 Serial.println("AT+CIPSTART=\"TCP\",\"" + host + "\"," + httpPort);

 delay(50);                           // Wait a little for the ESP to respond


 if (Serial.find("OK"))

   Serial.println("ESP8266 Connected to server!!!") ;

}
```

```
//Sends data to Thingspeak

void send_data(void)

{

 String httpPacket = "GET " + url + String(temp) + "&field2=" + String(gas_sensor_value) +
"&field3=" + String(rain) + "&field4=" + String(Windspeedfloat) + " HTTP/1.1\r\nHost: " + host +
"\r\n\r\n";

 int length = httpPacket.length();

 // Send our message length

 Serial.print("AT+CIPSEND=");

 Serial.println(length);

 delay(10);       // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

 // Send our http request

 Serial.print(httpPacket);

 delay(10);      // Wait a little for the ESP to respond

 if (Serial.find("SEND OK\r\n"))

  Serial.println("ESP8266 sends data to the server");

}


void setup()

{

 pinMode(A1, INPUT);     //gas sensor analog input

 pinMode(7, OUTPUT);     //gas sensor digital output

 pinMode(A0, INPUT);     //temperature sensor analog input

 pinMode(A2, INPUT);     //potentiometer analog input
```

```
  setupESP8266();

  lcd.begin(16,2);

  pinMode(triggerPin, OUTPUT);

  pinMode(echoPin, INPUT);


}


void loop()

{

  //for temperature sensor

  temp_vout = analogRead(A0);

  voltage = temp_vout * 0.0048828125; //convert analog value between 0 to 1023 with
5000mV/5V ADC

  temp = (voltage - 0.5) * 100.0;

  Serial.println("Current temperature: " + String(temp));

  //for gas sensor

  gas_sensor_value = analogRead(gas_sensor_port);

  Serial.println("Gas sensor value: " + String(gas_sensor_value));

  if (gas_sensor_value > 200)

  {

    tone(7,523,1000);

  }
```

```
//for rainfall measurement

digitalWrite(triggerPin, LOW);

delayMicroseconds(2);

// Sets the trigger pin to HIGH state for 10 microseconds

digitalWrite(triggerPin, HIGH);

delayMicroseconds(10);

digitalWrite(triggerPin, LOW);

// Reads the echo pin, and returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

rain = 0.01723 * duration;

delay(10); // Delay a little bit to improve simulation performance


lcd.setCursor(0,0); // Sets the location at which subsequent text written to the LCD will be
displayed

lcd.print("Rainfall: "); // Prints string "Rainfall" on the LCD

lcd.print(rain); // Prints the distance value from the sensor

Serial.println("Rainfall: " + String(rain));

delay(10);


float V_wind = analogRead(A2) * (5.0 / 1023.0);

Windspeedint = (V_wind - 0.4) * 10 * 2.025 * 2.237;     // For LCD screen output

Windspeedfloat = (V_wind - 0.4) * 10 * 2.025 * 2.237; // For Serial monitor output
```

```
lcd.setCursor(0,1);        // adjust cursor
lcd.print("Wind speed");
lcd.print(" ");
if (V_wind < 0.4)
{
  lcd.print("0");
}
else
{
  lcd.print(Windspeedint);
}
lcd.print("MPH");
//wind speed serial monitor output
Serial.print("Wind Speed: ");
if (Windspeedfloat <= 0)
{
  Serial.print("0.0");}
else{
  Serial.print(Windspeedfloat);}
Serial.println(" MPH");
delay(100);


Serial.print("Anemometer Voltage: ");
```

```
 if (V_wind > 2){
  Serial.println("Out of range!");
 }
 else if (V_wind < 0.4)
 {
  Serial.println("Out of range!");
 }
 else{
  Serial.print(V_wind);
  Serial.println(" V");}

 send_data();
 delay(1000);     // delay changed for faster analytics
}
```

Conclusion

One of the cost-effective and very flexible ways to implement data collection, analysis, and transmission of meteorological information is based on Arduino Uno. You are free to make a reliable, customized weather monitoring system that fits in most applications and situations by appropriately choosing sensors, programming Arduino Uno, implementing techniques for data transmission and processing, creating a user interface, and optimizing power usage. The system requires continuous testing, calibration, deployment, and maintenance in order to remain accurate and reliable over time.

References

1. Wang, J., Lim, M. K., Wang, C., & Tseng, M.-L. (2021). The evolution of the internet of things (IOT) over the past 20 years. Computers & Industrial Engineering, 155, 107174. https://doi.org/10.1016/j.cie.2021.107174

2. Rao, Y. N., Chandra, P. S., Revathi, V., & Kumar, N. S. (2020). Providing enhanced security in IOT based Smart Weather System. Indonesian Journal of Electrical Engineering and Computer Science, 18(1), 9. https://doi.org/10.11591/ijeecs.v18.i1.pp9-15

3. Mohamed, K. S. (2021). An introduction to IOT. Bluetooth 5.0 Modem Design for IoT Devices, 33–43. https://doi.org/10.1007/978-3-030-88626-4_2

4. Sabin, Foltean, and Bogdana Glovatchi. “Business Model Innovation for IoT Solutions: An Exploratory Study of Strategic Factors and Expected Outcomes.” Www.amfiteatrueconomic.ro, vol. 23, no. 57, May 2021, p. 392, https://doi.org/10.24818/ea/2021/57/392.

5. Dhar, Suparna, and Indranil Bose. “Securing IoT Devices Using Zero Trust and Blockchain.” Journal of Organizational Computing and Electronic Commerce, 17 Nov. 2020, pp. 1–17, https://doi.org/10.1080/10919392.2020.1831870.