TABLE OF CONTENTS

CHAPTER 1 – INTRODUCTION

PROJECT OVERVIEW: Weather Monitoring System

The Weather Monitoring System is an IoT-based project designed to measure and monitor weather parameters like temperature, humidity, and pressure. It uses sensors (DHT11/DHT22, BMP180) connected to a NodeMCU ESP8266 microcontroller to collect and wirelessly transmit data to the cloud via platforms like ThingSpeak.

Key features include real-time data monitoring, local display for immediate readings, and remote access to cloud-stored data for analysis. The system is scalable, allowing additional sensors like rain or wind-speed detectors to be integrated. This project is ideal for applications in agriculture, smart cities, environmental studies, and disaster management. Future improvements could involve predictive weather analytics and smart integrations.

OBJECTIVE AND SIGNIFICANCE

The **Weather Monitoring System** aims to provide a reliable and scalable solution for monitoring environmental parameters like temperature, humidity, and atmospheric pressure in real time. The system collects data using sensors such as DHT11/DHT22 for temperature and humidity, and BMP180 for pressure. This data is transmitted wirelessly via a NodeMCU ESP8266 to a cloud platform like ThingSpeak, where it can be accessed remotely for analysis. The objective is to ensure continuous data collection and provide both local and cloud-based visualization, allowing users to monitor weather conditions conveniently. The system is designed to be scalable, with the potential to incorporate additional sensors like rain or wind-speed detectors for more comprehensive monitoring.

The significance of the **Weather Monitoring System** lies in its ability to provide accurate, real-time weather data for various applications. It helps farmers optimize crop management by monitoring conditions such as temperature and humidity, which are crucial for irrigation and crop growth. Additionally, the system supports environmental research by tracking climate trends over time. It also plays a key role in disaster management, offering early warnings for extreme weather events, and can contribute to smart city infrastructure by aiding in energy management and urban planning. With its affordability, scalability, and integration with cloud technologies, this system offers a valuable tool for anyone needing reliable weather data.

1. APPLICATIONS IN REAL WORLD SCENARIOS

1.1 Agriculture:

Farmers rely on accurate weather data to optimize irrigation, planting, and harvesting schedules. By using a weather monitoring system, farmers can monitor temperature and humidity levels in real-time, ensuring crops receive the right amount of water and sunlight. For example, in regions where water scarcity is a concern, weather data helps farmers conserve water by only irrigating when necessary, improving crop yield and sustainability.

1.2 Smart Cities:

Weather monitoring systems play a vital role in the development of smart cities. By integrating real-time weather data into urban planning systems, city authorities can make informed decisions regarding energy usage, traffic management, and emergency response. For instance, during a heatwave, the system can provide data that helps manage power consumption and prevent blackouts. Additionally, it can guide traffic flow by predicting weather-related disruptions, such as heavy rain or fog.

1.3 Environmental Monitoring and Climate Research:

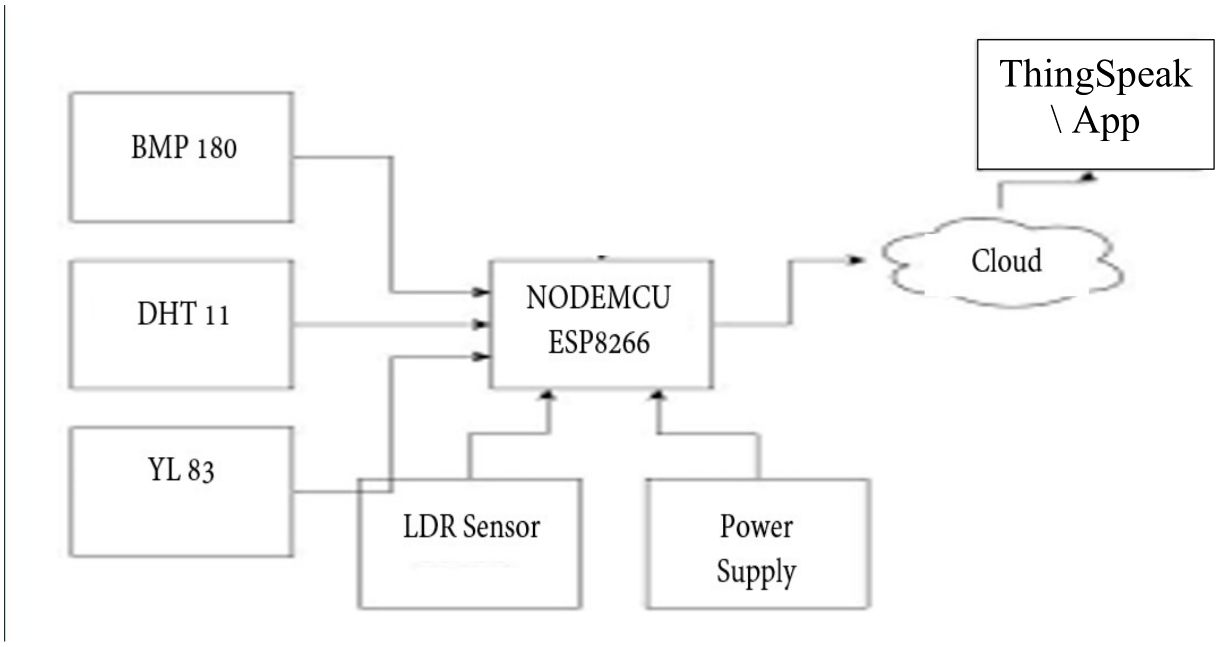
In environmental science, accurate weather data is crucial for understanding climate change and tracking long-term weather patterns. Researchers use weather monitoring systems to collect real-time data on atmospheric pressure, temperature, and humidity, which helps in predicting climate trends. The data collected can also be used to assess environmental changes like deforestation, desertification, and urban heat islands, aiding in the development of strategies to mitigate these issues.

1.6 Home Automation and Personal Use:

On a personal level, weather monitoring systems can be integrated into home automation systems. Homeowners can track indoor and outdoor weather conditions and automate responses, such as adjusting heating or cooling systems based on real-time temperature readings. For example, if the system detects high humidity levels inside a home, it can trigger a dehumidifier to maintain comfortable living conditions.

CHAPTER 2 – BLOCK AND CIRCUIT DIAGRAM

2.1 BLOCK DIAGRAM



The Weather Monitoring System uses the NodeMCU ESP8266 as the microcontroller and integrates sensors like DHT11 (temperature and humidity), BMP180 (pressure), Rain Sensor, and LDR Sensor (light intensity). The NodeMCU processes the data from these sensors and sends it to the cloud for monitoring.

Component : -

1. NodeMCU ESP8266

The NodeMCU is the central controller that connects to the sensors and the internet via Wi-Fi. It reads data from the sensors and sends it to a cloud platform (e.g., ThingSpeak) for remote monitoring. It is powered by 5V and communicates with the sensors using its GPIO pins.

2. DHT11 Sensor

The DHT11 measures temperature and humidity. It is connected to the GPIO pin D2 of the NodeMCU. The sensor communicates using a single data pin that sends digital signals to the microcontroller.

3. BMP180 Sensor

The BMP180 measures atmospheric pressure. It is connected to the SDA (D1) and SCL (D2) pins of the NodeMCU via I2C communication. These pins allow the NodeMCU to receive pressure data.

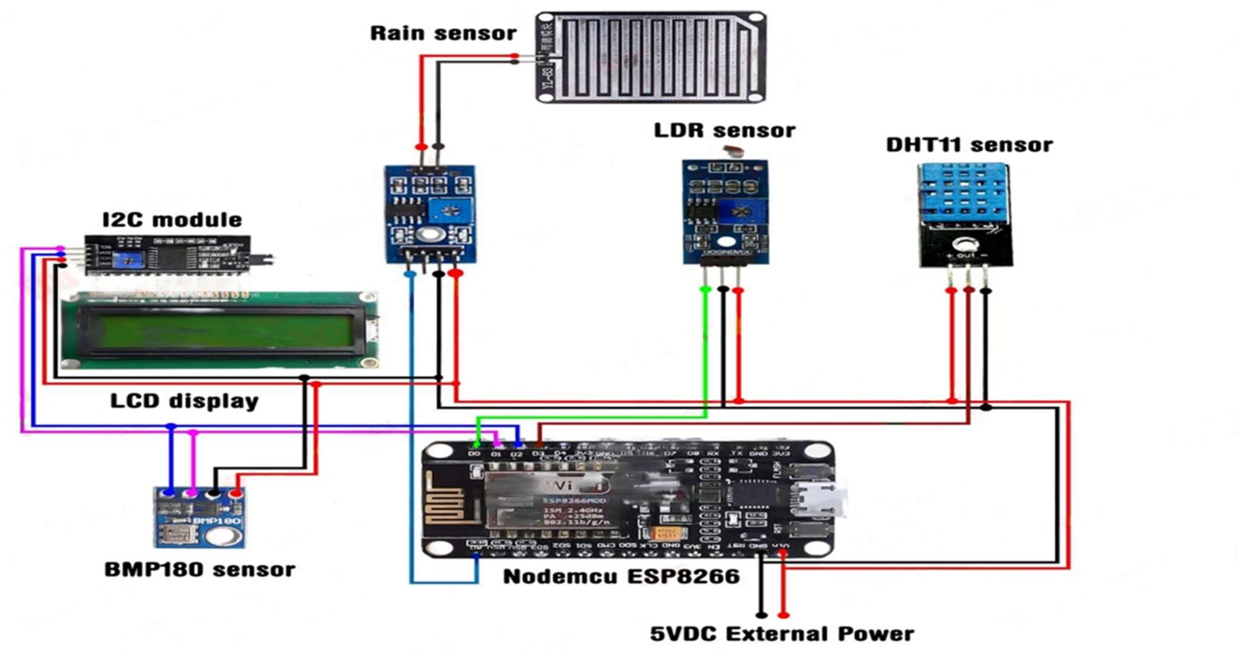
4. Rain Sensor

The Rain Sensor detects rainfall. It has an output pin connected to GPIO pin D5 of the NodeMCU. The sensor sends a high or low signal to the microcontroller based on rain detection.

5. LDR Sensor

The LDR Sensor measures light intensity. It is connected to the A0 (Analog pin) of the NodeMCU. The LDR changes its resistance based on light, and the analog signal is read by the NodeMCU.

2.2 CIRCUIT DIAGRAM



In the Weather Monitoring System, the NodeMCU ESP8266 microcontroller connects to the following sensors: DHT11 (temperature and humidity), BMP180 (pressure), Rain Sensor, and LDR Sensor. Below is the explanation of the connections:

Component :-

1. NodeMCU ESP8266:
   - Power Supply: The NodeMCU is powered by a 5V supply connected to the Vin pin.
   - GND: Connected to the ground of all components (DHT11, BMP180, Rain Sensor, LDR).
   - GPIO Pins: These pins are used to connect the sensors for data input/output.
2. DHT11 Sensor (Temperature and Humidity):
   - VCC: Connected to the 3.3V pin of the NodeMCU.
   - GND: Connected to the GND of the NodeMCU.
   - Data Pin: The DATA pin of the DHT11 is connected to GPIO pin D2 of the NodeMCU for digital data transmission.
3. BMP180 Sensor (Pressure):
   - VCC: Connected to the 3.3V pin of the NodeMCU.
   - GND: Connected to the GND of the NodeMCU.
   - SDA (Data): Connected to GPIO pin D1 (SDA) of the NodeMCU for I2C communication.
   - SCL (Clock): Connected to GPIO pin D2 (SCL) of the NodeMCU for I2C communication.
4. Rain Sensor:
   - VCC: Connected to the 3.3V pin of the NodeMCU.
   - GND: Connected to the GND of the NodeMCU.
   - Signal Pin: The Signal pin of the rain sensor is connected to GPIO pin D5 of the NodeMCU. This pin sends a digital signal indicating the presence or absence of rain.
5. LDR Sensor (Light Intensity):
   - VCC: Connected to the 3.3V pin of the NodeMCU.
   - GND: Connected to the GND of the NodeMCU.
   - Signal Pin: The Signal pin is connected to the A0 (Analog pin) of the NodeMCU, which measures light intensity based on the resistance of the LDR.

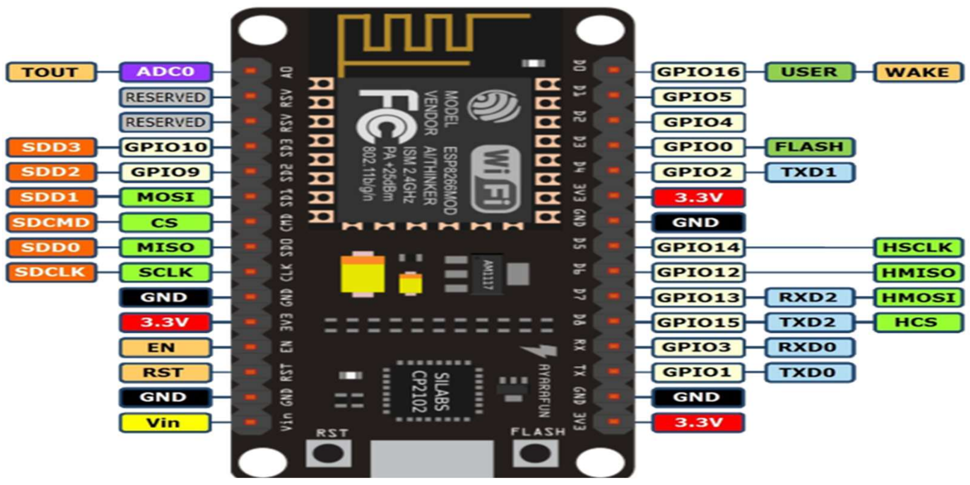
CHAPTER 3 – HARDWARE & SOFTWARE SPECIFICATIONS

HARDWARE SPECEFICATION

1. NodeMCU ESP8266 (Microcontroller)

The NodeMCU ESP8266 is a low-cost, Wi-Fi-enabled microcontroller based on the ESP8266 chipset. It is widely used in IoT projects due to its ability to connect to the internet and interact with sensors and cloud platforms.

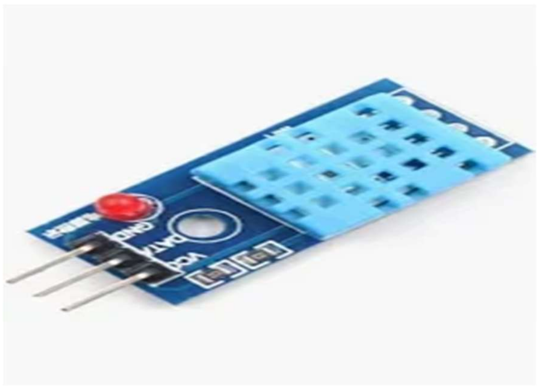
- Features:
    - Wi-Fi capability: Allows the system to connect to the internet and upload data to cloud platforms like ThingSpeak.
    - GPIO pins: Provides multiple input/output pins to interface with sensors.
    - Power Supply: Operates on 3.3V but can be powered by a 5V supply via its Vin pin.
- Role in the system: The NodeMCU acts as the central controller, collecting sensor data and transmitting it over Wi-Fi to the cloud for monitoring and analysis.



2. DHT11 Sensor (Temperature and Humidity)

The **DHT11** sensor is used to measure temperature and humidity. It is a simple, low-cost sensor with a limited accuracy range, making it suitable for basic weather monitoring.

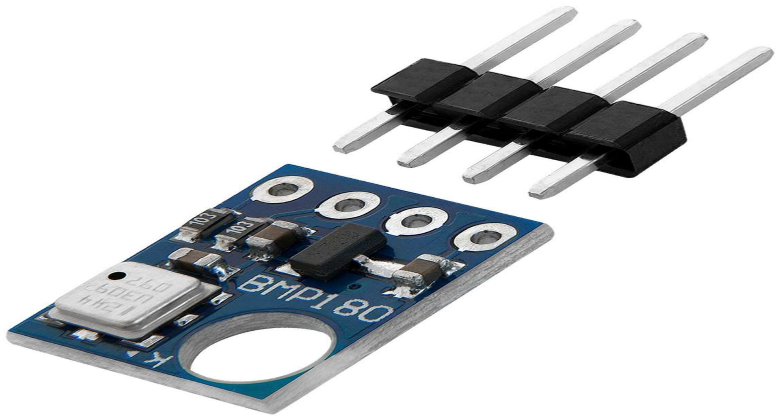
- **Features**:
    - Measures temperature in the range of **0-50°C**.
    - Measures humidity in the range of **20-80%**.
    - Communicates via a single-wire digital interface.
- **Role in the system**: The DHT11 provides temperature and humidity data, which is essential for monitoring environmental conditions and making decisions like irrigation control in agriculture.



3. BMP180 Sensor (Pressure and Altitude)

The BMP180 is an accurate barometer used to measure atmospheric pressure. It uses a capacitive pressure sensor and is commonly used in weather monitoring systems for detecting changes in weather conditions.

- Features:
  - Measures atmospheric pressure in the range of 300 to 1100 hPa (hectopascals).
  - Can also provide altitude readings based on the pressure.
  - Uses I2C communication to connect to the microcontroller.
- Role in the system: The BMP180 sensor provides atmospheric pressure data, which is important for predicting weather patterns, such as storms or pressure changes indicating rainfall.



4. Rain Sensor

The Rain Sensor is used to detect the presence of rain. It consists of two copper traces on a PCB that close when water (rain) bridges the gap, allowing the sensor to detect rainfall.

- Features:
  - Simple design with a digital output indicating whether it is raining or not.
  - Can be used to trigger automatic irrigation systems or alert the system to rain events.
- Role in the system: The Rain Sensor detects the presence of rain, providing critical data for applications like weather monitoring or automatic irrigation in agriculture.

5. LDR Sensor (Light Dependent Resistor)

The **LDR (Light Dependent Resistor)** sensor is used to measure the intensity of light. The resistance of the LDR decreases as the light intensity increases, making it useful for detecting day/night cycles or sunlight levels.

- **Features**:
  - The resistance of the LDR decreases in proportion to the intensity of light.
  - It is connected to the microcontroller's **analog input pin** to measure varying light levels.
- **Role in the system**: The LDR sensor provides data on light intensity, which can be used to monitor day and night conditions, control street lights, or analyze sunlight exposure for agricultural applications.

- **Breadboard:** Used for prototyping circuit connections.

SOFTWARE SPECIFICATION

The software of the Weather Monitoring System consists of two main components: the NodeMCU firmware that runs on the microcontroller and the cloud platform (such as ThingSpeak) where the data is sent for remote monitoring. Below is a breakdown of the software specifications used in the project:

1. NodeMCU Firmware (Microcontroller Software):-

The firmware running on the NodeMCU ESP8266 is developed using the Arduino IDE and the ESP8266 board libraries. The firmware controls the NodeMCU and manages sensor data collection, processing, and transmission over Wi-Fi.

Software Tools:

- Arduino IDE: The primary Integrated Development Environment (IDE) used for writing, compiling, and uploading code to the NodeMCU.
- Libraries:
  - ESP8266 Wi-Fi Library: Manages Wi-Fi connectivity for the NodeMCU.
  - DHT Library: For interfacing with the DHT11 sensor.
  - Wire Library: For I2C communication with the BMP180 sensor.
  - ThingSpeak Library: For sending data to the ThingSpeak cloud platform.

2. Cloud Platform (ThingSpeak)

ThingSpeak is used as the cloud platform to collect, visualize, and analyze sensor data. It allows users to monitor the data remotely through a web interface.

Software Tools:

- **ThingSpeak Account**: A free cloud service that provides an API for data collection and visualization.
- **ThingSpeak API**: Used to send data from the NodeMCU to the platform.

3. Webpage & App for Displaying Weather Data:-

To display the weather data from your Weather Monitoring System on a webpage and a mobile app, you can integrate both web technologies and mobile app development tools. Below is an explanation of how you can build both:

- ### Webpage for Displaying Weather Data:-

For the webpage, you can use HTML, CSS, and JavaScript to build a simple user interface. You can retrieve the data from ThingSpeak and display it dynamically using JavaScript. Here's an outline of how to proceed:

Tools and Technologies:

- HTML: For structuring the webpage.
- CSS: For styling the webpage (make it visually appealing).
- JavaScript: For fetching data from ThingSpeak and displaying it in real-time.
- ThingSpeak API: To retrieve sensor data from your ThingSpeak channel.

Steps to Build the Webpage:

1. Create a ThingSpeak Channel:
    - Set up your ThingSpeak channel and generate the API Key to access the data.
2. Build the Webpage:
    - HTML: Create a basic webpage layout to display data (temperature, humidity, pressure, rain status, light intensity).
    - CSS: Style the page for a clean and professional look (using flexbox or grid layouts, background images, and appropriate fonts).
    - JavaScript: Use the ThingSpeak API to fetch the sensor data in real time and update the webpage dynamically.

- ### Mobile App for Displaying Weather Data:-

For the mobile app, you can use frameworks like **Flutter** or **React Native** for cross-platform development (Android and iOS). Below is an outline of how to build the mobile app:

Tools and Technologies:

- **Flutter** (Recommended): A popular cross-platform framework to build apps for Android and iOS.
- **Dart**: Programming language for Flutter.
- **ThingSpeak API**: To retrieve sensor data.

CHAPTER 4 – CODE

```
#include <ESP8266WiFi.h>
#include "DHT.h"
#include <Adafruit_BMP085_U.h>  // Library for BMP180 sensor

// Define the DHT11 sensor and BMP180 sensor
DHT dht(D3, DHT11);  // DHT11 sensor connected to D3
Adafruit_BMP085_Unified bmp; // BMP180 sensor instance

WiFiClient client;

// Define ThingSpeak API key and Wi-Fi credentials
String apiKey = "VOTZ9OSF0U1XS2P0";
const char *ssid = "sushant072";
const char *pass = "sushant2004";
const char* server = "api.thingspeak.com";

// Pin definitions
const int rainSensorPin = A0;  // Rain sensor connected to A0 (used by both
rain and LDR)
const int ldrSensorPin = D0;   // LDR sensor connected to A1 (updated from
A0)

void setup() {
  Serial.begin(115200);
  delay(10);

  // Initialize DHT11 sensor
  dht.begin();
  delay(2000);  // Delay to allow DHT sensor to initialize

  // Initialize BMP180 sensor
  if (!bmp.begin()) {
    Serial.println("Could not find a valid BMP180 sensor, check wiring!");
    while (1); // Halt the program if the sensor is not found
  }

  // Connect to Wi-Fi
  Serial.print("Connecting to Wi-Fi");
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected");
}

void loop() {
  // Read data from DHT11 sensor (Temperature and Humidity)
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  // Retry loop if DHT sensor fails to read
  int retryCount = 0;
  while ((isnan(h) || isnan(t)) && retryCount < 5) {
    Serial.println("Failed to read from DHT sensor! Retrying...");
    delay(2000);  // Wait and retry
    h = dht.readHumidity();
    t = dht.readTemperature();
    retryCount++;
  }

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor after multiple attempts.");
    return;
  }

  // Read rain sensor (A0 pin) and map to percentage
  int rainValue = analogRead(rainSensorPin);
  int rainPercentage = map(rainValue, 0, 1024, 0, 100);  // Map rain sensor to
percentage

  // BMP180 sensor (Temperature and Pressure)
  sensors_event_t event;
  bmp.getEvent(&event); // Get the sensor event
  float bmpTemp = t;  // Using DHT11 for temperature (as BMP180's temp is
similar)
  float pressure = event.pressure;  // Pressure in hPa (millibar)

  // Read LDR sensor (A1 pin) and map to percentage
  int ldrValue = analogRead(ldrSensorPin);
  int lightIntensity = map(ldrValue, 0, 1024, 0, 100);  // Map LDR value to
percentage

  // Send data to ThingSpeak
  if (client.connect(server, 80)) {
    String postStr = apiKey;
    postStr += "&field1=" + String(t);         // Temperature from DHT11
    postStr += "&field2=" + String(h);         // Humidity from DHT11
    postStr += "&field3=" + String(pressure);  // Pressure from BMP180
    postStr += "&field4=" + String(rainPercentage);  // Rain sensor value
    postStr += "&field5=" + String(lightIntensity);  // LDR sensor value
    postStr += "\r\n\r\n\r\n\r\n";

    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n\n\n");
    client.print(postStr);

    // Debugging output
    Serial.print("Temperature (DHT11): ");
    Serial.println(t);
    Serial.print("Humidity: ");
    Serial.println(h);
    Serial.print("Pressure (BMP180): ");
    Serial.println(pressure);
    Serial.print("Rain: ");
    Serial.println(rainPercentage);
    Serial.print("Light Intensity: ");
    Serial.println(lightIntensity);
  }
  client.stop();
  delay(1000);  // Wait for 1 second before next loop
}
```

CHAPTER 5 – WORKING PRINCIPLE & APPLICATION

Working Principle of the Weather Monitoring System:-

The Weather Monitoring System works by continuously collecting environmental data from various sensors and transmitting it to a cloud platform for storage, visualization, and analysis. Below is a breakdown of how the system functions:

1. Data Collection (Sensors):-

The system uses multiple sensors to gather environmental data:

- DHT11 Sensor: Measures temperature and humidity.
- BMP180 Sensor: Measures atmospheric pressure and can also calculate altitude.
- Rain Sensor: Detects the presence of rain.
- LDR Sensor: Measures light intensity (used to detect day/night cycles or sunlight levels).

2. Microcontroller (NodeMCU ESP8266):-

The NodeMCU ESP8266 acts as the central processing unit. The microcontroller is responsible for:

- Interfacing with sensors: It reads data from the sensors through digital and analog pins.
- Processing the data: The data collected from the sensors is processed by the microcontroller.
- Wi-Fi Communication: The NodeMCU uses its Wi-Fi capabilities to send the data over the internet to a cloud platform such as ThingSpeak.

3. Data Transmission:-

- Wi-Fi Module (ESP8266): Once the data is collected, the NodeMCU sends it to the cloud platform (e.g., ThingSpeak) using HTTP requests. The data is sent to the platform in real-time or at regular intervals (e.g., every 10-30 seconds).

4. Cloud Platform (ThingSpeak):-

- Data Storage and Visualization: The cloud platform stores the data and provides a web interface to visualize the collected data. Users can view real-time graphs, statistics, and historical data on the web or through a mobile app.

5. User Access:-

- Users can access the data remotely via the webpage or mobile app that fetches data from the ThingSpeak API. This allows monitoring of the environmental parameters such as temperature, humidity, pressure, rain status, and light intensity from anywhere with an internet connection.

Applications of the Weather Monitoring System:-

The Weather Monitoring System has a wide range of applications across various fields. Below are some notable examples:

1. Agriculture and Farming:-

- Irrigation Management: The system can be used to monitor environmental conditions like temperature, humidity, and rain to optimize irrigation. If the system detects rain or high humidity, it can automatically control irrigation systems to avoid water wastage.
- Crop Growth Monitoring: Monitoring temperature and light intensity helps farmers make informed decisions about planting and harvesting crops.
- Weather Alerts: Alerts for rain or high winds can help farmers take necessary precautions to protect crops.

2. Environmental Monitoring:-

- Weather Stations: This system can serve as a simple weather station to monitor the local weather in real-time, including parameters like temperature, pressure, and humidity.
- Climate Research: The system can be deployed in remote or rural areas to gather data for climate research, particularly in areas with limited access to traditional weather stations.

3. Smart Homes:-

- Automation: The weather data can be used to automate smart home systems, such as controlling windows, curtains, or HVAC systems based on temperature or humidity readings.
- Energy Efficiency: The system can monitor light intensity and adjust artificial lighting accordingly, optimizing energy consumption in the home.

4. Industrial Applications:-

- Factory Environment Control: The system can be used in factories or warehouses to monitor temperature and humidity to ensure the ideal conditions for equipment, goods, or machinery.
- Product Storage: Monitoring environmental conditions is crucial for storing temperature-sensitive products, such as food or pharmaceuticals, in warehouses or storage units.

CHAPTER 6 - CONCLUSION

1.1 SUMMARY

The **Weather Monitoring System** is an IoT-based project designed to collect, process, and display environmental data in real-time. It uses a variety of sensors, including the **DHT11** for temperature and humidity, **BMP180** for atmospheric pressure and altitude, **Rain Sensor** for detecting rainfall, and **LDR** for measuring light intensity. These sensors are connected to a **NodeMCU ESP8266** microcontroller, which processes the data and sends it to a cloud platform like **ThingSpeak** using Wi-Fi. The data is then accessible remotely through a **webpage** or a **mobile app**.

This system is versatile and can be applied in multiple real-world scenarios, such as **agriculture**, where it helps optimize irrigation and monitor crop health, **environmental monitoring** for local weather stations, and **smart homes**, where it can automate systems based on environmental conditions. It also plays a key role in **disaster management**, providing early warnings of extreme weather events. By offering real-time weather data, the system enables better decision-making and enhances awareness of local environmental conditions.

1.2 FUTURE SCOPE

The Weather Monitoring System has significant potential for further enhancement and application in various domains. Here are some future scope areas for this project:

1. **Integration with Advanced Sensors:**
    - The system can be expanded by integrating more advanced sensors, such as PM2.5 sensors for air quality monitoring, UV sensors for measuring ultraviolet radiation, and wind speed/direction sensors to provide a more comprehensive environmental analysis.
2. **Data Analytics and Machine Learning:**
    - Advanced data analytics, including machine learning algorithms, can be employed to predict weather patterns and anomalies, providing users with forecasts and recommendations based on historical and real-time data.
3. **Automated Control Systems:**
    - The system can be integrated with automation systems in smart homes or agricultural systems. For example, based on weather conditions, irrigation systems can be automatically adjusted, or HVAC systems can be controlled based on temperature and humidity readings.

4. **Integration with Other IoT Devices:**
    - The system can be connected with other IoT devices such as smart thermostats, air purifiers, or irrigation controllers, enabling a fully automated and interconnected smart environment.
5. **Enhanced Mobile App Features:**
    - The mobile app can be enhanced with features like push notifications for weather alerts, real-time tracking, and data visualization tools for a more user-friendly experience.
6. **Energy Efficiency and Sustainability:**
    - The system can be used to monitor energy usage in buildings by integrating it with smart grids and solar panel systems, helping optimize energy consumption based on weather conditions.
7. **Global Deployment:**
    - The system can be deployed globally to monitor different weather conditions in various geographical locations. Data from multiple stations can be combined to create a global weather network.
8. **Integration with Social Media for Alerts:**
    - Weather alerts can be integrated with social media platforms, allowing automated weather warnings and updates to be sent to users via Twitter, Facebook, or SMS.
9. **Integration with Government and Research Institutions:**
    - The system could be expanded to provide data for government weather stations and research institutions for studying weather patterns, environmental changes, or disaster management.

REFERENCES:-

1. https://srituhobby.com/iot-based-weather-monitoring-system-using-nodemcu-and-thingspeak/

2. https://weather-information-rouge.vercel.app/

3. https://github.com/lifeline024/WeatherInformation

4. https://thingspeak.mathworks.com/channels/2680124/private_show