KGISL INSTITUTE OF TECHNOLOGY, COIMBATORE

AIR QUALITY MONITORING

Submitted by

ADSHAYA S S **[711721106007]**
LAKSHMI PRABHA M **[711721106056]**
LOKESH KIRAN C K **[711721106057]**
PRABHAHAR S **[711721106307]**
KAUNG KYAW KHANT **[711721106303]**
MITHUN KUMAR S **[711721106306]**

CONTENTS

ABSTRACT

One of the largest hazards to the environment in the modern era is air pollution. Air pollution affects everyone on a daily basis, including people, animals, crops, cities, forests, and aquatic environments. In addition, it should be kept under control at a particular level to stop the rate of world warming. Designing an IOT-based air pollution monitoring system is the goal of this project monitoring the air quality using the internet from any location and a computer or mobile device context and environment. Various techniques and tools are available for the air quality evaluation and monitoring

CHAPTER-1
AIM OF PROJECT

The increase in dangerous gas and particulate concentrations in the environment, car emissions, and industrial release of toxic gases are all contributing to air pollution. Due to causes including industries, urbanization, population growth, and vehicle use, which might have an impact on human health, the level of pollution is rising quickly. Particulate matter is one of the most crucial factors contributing significantly to the rise in air pollution. Due to this, real-time air quality monitoring requires measurement and analysis in order for fast, suitable judgments to be made. In this paper, real-time standalone air quality monitoring is presented. The Internet of Things (IoT) is currently finding extensive use in every industry and is essential to our system for monitoring air quality. The configuration will display the air quality in PPM on the website so that we can readily monitor it. With this IoT project, you can use your computer or mobile device to check the pollution level from anywhere.

CHAPTER 2

PROJECT OBJECTIVES

1. **Real-time Air Quality Monitoring:**
    - Use IoT devices to set up a continuous measurement system for important air quality metrics, such as carbon dioxide (CO2), nitrogen dioxide (NO2), sulfur dioxide (SO2), ozone (O3), and particulate matter (PM2.5 and PM10), carbon monoxide (CO) and other pertinent contaminants.
    - Verify that the devices deliver precise and timely data updates for monitoring in real time.
2. **Sharing of Data:**
    - Create a strong infrastructure for data sharing to enable the smooth interchange of air quality information. information between the central platform and IoT devices.
    - Use established, secure data transport methods to guarantee data integrity and confidentiality.

3. **Public Awareness:**
    - Create an intuitive user interface for the web platform so that the general public may obtain data on air quality.
    - Use tools like charts, alerts, and visualizations to increase public awareness of air quality conditions of quality.Include instructional materials on the platform to educate users on the effects of air quality on the environment and human health.

4. **Health Impact Evaluation:**
    - Use models or algorithms to evaluate how the observed air quality conditions might affect human health.Inform the public, especially vulnerable groups, on the health dangers connected to present air quality levels.

2.1 IOT Devices and Designs

1.Sensor Selection:

- Take accuracy, sensitivity, and calibration needs into account while selecting premium sensors for each air quality measurement.
- Make sure the sensors can function in a range of environmental circumstances.

2.Energy Source:

- When designing systems, take battery life and the availability of other power sources, such as solar energy, into account.

3.Interaction:

- Include dependable communication modules (such as cellular, Wi-Fi, and LoRa) to facilitate the transfer of data from sensors to the central platform.

4.Protection and Enclosure:

- Create robust, weatherproof enclosures that shield Internet of Things devices from the elements.
- Put precautions in place to stop vandalism and tampering.

2.2 Data Sharing Platform

1. User Interface:

- Create a user-friendly, responsive website that makes it simple to navigate and comprehend data on air quality.
- Provide dashboards and maps that may be customized to accommodate various user preferences.

2. Processing and Storing of Data:

- Establish a safe and scalable database system to hold both historical and current information on air quality.
- Create real-time data processing algorithms to produce insightful findings.

3.Notification System:

- Include an alerting system that uses multiple channels (such as email, SMS, and push notifications) to inform users of critical air quality situations.

4.Security of Access:

- Put in place user roles-based authentication and authorization systems to restrict access to various tiers of air quality data.

2.3 Integration Approach

1.Data Transmission Protocol:

- Establish a common data transmission protocol for communication between IoT devices and the data-sharing platform (e.g., MQTT, HTTP, CoAP).

2. API Creation:

- Provide thoroughly explained APIs to enable smooth communication between the web platform and IoT devices.

3. **Safety Procedures:**
    - To safeguard data transfer, use end-to-end encryption.To stop illegal access to the data-sharing platform, use secure authentication methods.
4. **Ability to Scale:**
    - Create a scalable system that can eventually support a growing number of IoT users and devices.
5. **Redundancy and Trustworthiness:**
    - Put redundancy mechanisms in place to guarantee platform availability and data integrity in the event of device or network failure.

You may create a thorough and efficient system for data sharing, public awareness, health impact assessment, and real-time air quality monitoring by taking these goals and factors into account.

CHAPTER 3

DECRIPTION OF THE COMPONENTS

3.1 Components used

Hardware components :

1. NodeMCU V3

2. DHT11 Sensor Module

3. MQ-135 Gas Sensor Module

4. Veroboard(KS100)

5. Breadboard

6. Connecting Wires

7. AC-DC Adapters

8. LEDs emitting green, yellow and red colours

9. Resistors

Software components :

1. ThinkSpeak Cloud

2. Arduino IDE

3.2 Brief description of the Components

- **Node MCU V3**

A CH340G USBTTL Serial chip is included in the open-source ESP8266 development kit known as NodeMCU V3. The ESP8266 Wi-Fi SoC from Espressif Systems powers its firmware. CH340 is extremely reliable even in industrial applications while being less expensive. Testing shows it to be stable on all systems that are supported as well. It is easily programable with the Arduino IDE. It uses relatively little current—between 15 A and 400 mA**.**

- **DHT11 Sensor Module**

The DHT11 is a temperature and humidity sensor that provides a voltage-based digital output. It measures using a thermistor and a capacitive humidity sensor. the environment, we must provide a 5V voltage. connect it (DC) to the GND pin and the Vcc pin. The Data pin in the sensor can be quickly used to read the output. voltage terms (in digital mode). The humidity sensing capacitor is used to measure humidity. has two electrodes and a substrate that can store moisture. between them, a dielectric The capacitance value changes when the changes in the humidity. The IC measures these **revised resistance** values, processes them, and then creates a digital version of them.

Temperature Measurement: The DHT11 sensor employs an infrared sensor to measure the negative

- **MQ -135 Gas Sensor Module**

SnO2, the substance that makes up MQ135, is unique in that it conducts very little heat when exposed to pure air but produces a lot of heat when placed in an environment with flammable gas. nice conductivity performance. Simply create a basic electronic circuit, turn the a conductivity change that results in a signal at the output. MQ135 gas sensor has a range of Ammonia, hydrogen sulfide, benzoene, steam, and other dangerous gasses. used by the family hazardous gas detector for the environment that works with ammonia, aromatics, benzene vapor, sulfur dioxide, and other dangerous gases and smoke, gas detection, and concentration tests 10 to 1000 ppm range. In a typical setting, the setting that doesn't have the analog output of the sensor was set as the reference voltage when gas was detected. About 1V will be the voltage.

- **Veroboard**

The initial prototyping board is called Veroboar These, often known as "stripboard" or "matrix board," provide complete freedom for hard wiring. separate elements. constructed from a Epoxy-based or laminated copper sheet .It is available in single- and double-sided forms and has a substrate. Vero boards come in a range of sizes. a variety of board sizes in both imperial and metric With metric pitch, Veroboard makes a great foundation. circuit design and provides even more flexibility utilizing our selection of terminal pins and assemblies. Similarly to other stripboards, when using .To create the Veroboard, the components are placed appropriately and connected to the wires. necessary circuit. To divide the strips, breaks can be made in the tracks, typically around holes. into several electrical nodes, enabling more complicated circuits.

- **AC -DC Power Adapter**

An electrical device known as an AC-DC power supply or converter takes electricity from a grid-based power source and changes its current, frequency, and voltage. AC-DC Power supplies are required to deliver the proper power requirements of an electrical component.

Electrical current is delivered to devices by the ACDC power supply. typically be powered by batteries or not have any other power supply.

- **LED (Red, Green &Yellow)**

A semiconductor light source called a light-emitting diode (LED) produces light when current passes through it. Recombining electrons and electron holes in the semiconductor results in the release of energy in the form of photons. The color of the light (which reflects the intensity of the energy needed for electrons to pass the band gap of an object (and thus the number of photons) is based on the semiconductor. A layer of light-emitting phosphor or several semiconductors can be used to create white light on a semiconductor device. LEDs have numerous benefits over among the benefits of incandescent light sources are their lower power usage, longer lifetime, and improved physical durability, compactness, and quick switching. In return for these typically Positive qualities and negative aspects of LEDs low voltage electrical restrictions are among them. universal incapacity to DC (not AC) power

- **Resistors**

A resistor is a passive two-terminal electrical component used in circuits to implement electrical resistance. Resistors are used in

electrical circuits to utilized to modify signal levels, decrease current flow, and split voltages, apply active element bias, and shut off among other things, transmission cables. High-power resistors capable of dissipating many watts of electricity Using power as heat as part of a motor is possible. controllers, in power delivery systems, or during testing a generator's loads. Resistances exist in fixed resistors. that barely alter with changes in operation voltage, duration, or temperature.

- **Arduino IDE**

The open-source Arduino IDE is used to create and upload code to the Arduino boards. The IDE program is appropriate for various operating systems like Windows and Mac Linux and OS X. It backs up the C and other programming languages C++. Here, the acronym IDE is Comprehensive Development Environment. The software or Sketching is a term used to describe code created in the Arduino IDE. The Arduino needs to be connected. Requires an Arduino board equipped with the IDE to upload a sketch created using the IDE for Arduino. The sketch has the ".ino" file extension.

- **Think Speak Cloud**

ThingSpeak is free, open-source software that lets users talk to internet-connected gadgets. It was created in Ruby. It makes easier data

retrieval, logging, and access by providing a standardized API for devices and websites for social networks. The thingSpeak first introduced by ioBridge as a service for IoT application support. Support from the has been implemented into ThingSpeak. MATLAB is a numerical computing program. from MathWorks, enabling users of ThingSpeak without having to buy a license, one can use MATLAB to analyze and display uploaded data MathWork's license for MATLAB

3.3 Working Procedures

The primary regulating factor in this project is NodeMCU. It has been programmed such that it recognizes the sensory signals coming from the sensors and displays the quality level using led indicators. The DHT11 sensor module is utilized to gauge the environment's temperature and humidity. surroundings.The MQ-135 gas sensor module is used to detect air quality in ppm. These data are transmitted over the internet to the ThinkSpeak cloud. Additionally, we've offered LED signs to show the levels of safety.

STEP 1 :The MQ-135 gas sensor module is first calibrated in Step 1. The sensor is programmed to warm up for 24 minutes. When it happens, the software code is uploaded to The sensor calibration hardware circuit has been followed by NodeMCU performed.

STEP 2:The DHT11 sensor is then programmed to preheat for 10 minutes.

STEP 3:The final configuration of the device is made using the calibration results from STEP 1 operating code.

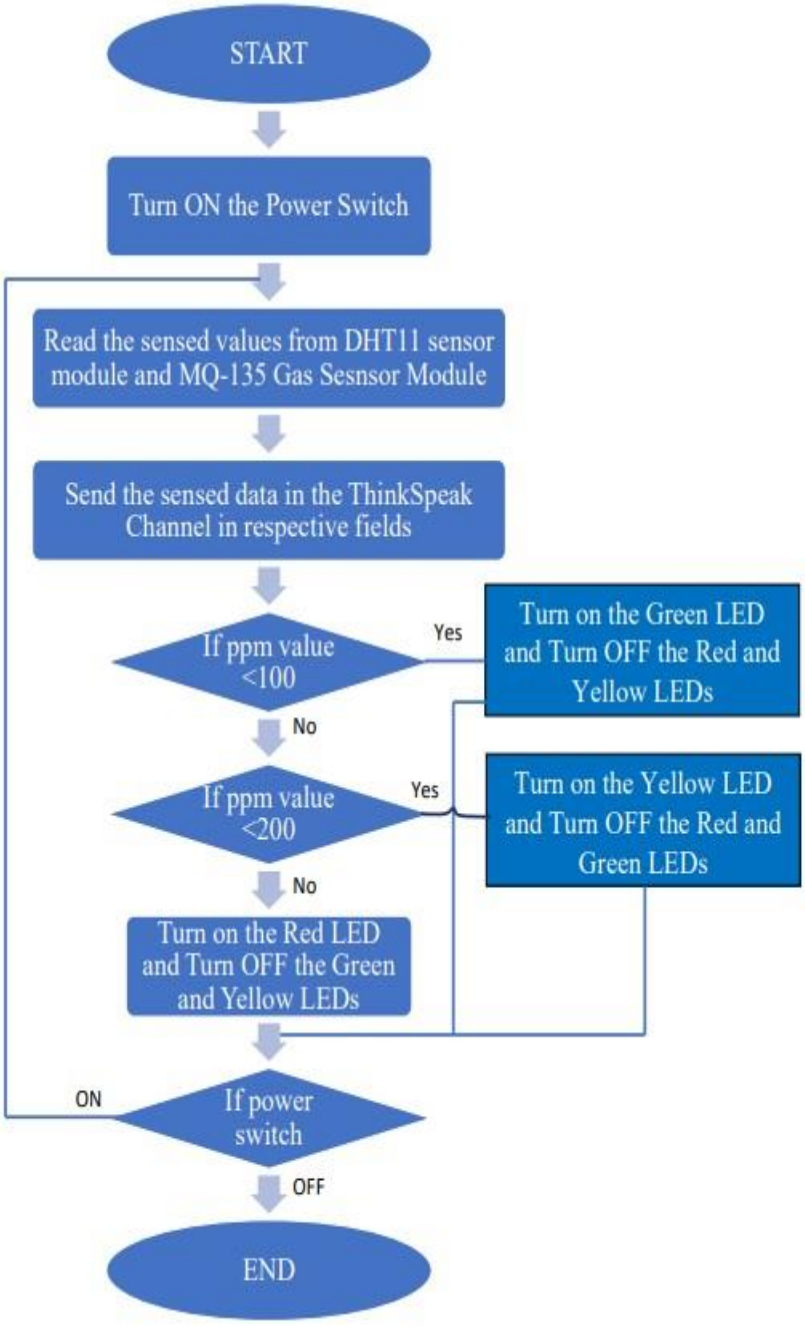
STEP 4:The NodeMCU is then updated with the finished, functional code.

STEP 5:The hardware circuit is implemented in its entirety at this point.

CHAPTER 4

SOFTWARE IMPLEMENTATION

4.1 Working Algorithm



4.2 Software Code for Callibration of MQ135 Sensor:

```
void setup()
{
Serial.begin(9600); //Baud rate
19 | P a g e
pinMode(A0,INPUT);
}
void loop()
{
 float sensor_volt; //Define variable for sensor voltage
 float RS_air; //Define variable for sensor resistance
 float R0; //Define variable for R0
 float sensorValue=0.0; //Define variable for analog readings
Serial.print("Sensor Reading = ");
Serial.println(analogRead(A0));
for(int x = 0 ; x < 500 ; x++) //Start for loop
 {
sensorValue = sensorValue + analogRead(A0); //Add analog values of
sensor 500 times
 }
sensorValue = sensorValue/500.0; //Take average of readings
sensor_volt = sensorValue*(5.0/1023.0); //Convert average to voltage
RS_air = ((5.0*1.0)/sensor_volt)-1.0; //Calculate RS in fresh air
 R0 = RS_air/3.7; //Calculate R0
Serial.print("R0 = "); //Display "R0"
```

```
Serial.println(R0); //Display value of R0
delay(1000); //Wait 1 second
}
```

4.3 Execution of the Main Program

```
#include <ESP8266WiFi.h>
#include <DHT.h>
#include <ThingSpeak.h>
DHT dht(D5, DHT11);
#define LED_GREEN D2
#define LED_YELLOW D3
#define LED_RED D4
#define MQ_135 A0
int ppm=0;
float m = -0.3376; //Slope
float b = 0.7165; //Y-Intercept
float R0 = 3.12; //Sensor Resistance in fresh air from previous code
WiFiClient client;
long myChannelNumber = 123456; // Channel id
const char myWriteAPIKey[] = "API_Key";
void setup() {
 // put your setup code here, to run once:
Serial.begin(9600);
pinMode(LED_GREEN,OUTPUT);
pinMode(LED_YELLOW,OUTPUT);
pinMode(LED_RED,OUTPUT);
pinMode(MQ_135, INPUT);
WiFi.begin("WiFi_Name", "WiFi_Password");
```

```
while(WiFi.status() != WL_CONNECTED)
 {
delay(200);
Serial.print(".");
 }
Serial.println();
Serial.println("NodeMCU is connected!");
Serial.println(WiFi.localIP());
dht.begin();
ThingSpeak.begin(client);
}
void loop() {
 float sensor_volt; //Define variable for sensor voltage
 float RS_gas; //Define variable for sensor resistance
 float ratio; //Define variable for ratio
 int sensorValue;//Variable to store the analog values from MQ-135
 float h;
 float t;
 float ppm_log; //Get ppm value in linear scale according to the the ratio value
 float ppm; //Convert ppm value to log scale
 h = dht.readHumidity();
delay(4000);
 t = dht.readTemperature();
delay(4000);
sensorValue = analogRead(gas_sensor); //Read analog values of sensor
sensor_volt = sensorValue*(5.0/1023.0); //Convert analog values to voltage
RS_gas = ((5.0*1.0)/sensor_volt)-1.0; //Get value of RS in a gas
```

```
 ratio = RS_gas/R0; // Get ratio RS_gas/RS_air
ppm_log = (log10(ratio)-b)/m; //Get ppm value in linear scale according to the ratio value
 ppm = pow(10, ppm_log); //Convert ppm value to log scale
Serial.println("Temperature: " + (String) t);
Serial.println("Humidity: " + (String) h);
Serial.println("Our desired PPM = "+ (String) ppm);
ThingSpeak.writeField(myChannelNumber, 1, t, myWriteAPIKey);
delay(20000);
ThingSpeak.writeField(myChannelNumber, 2, h, myWriteAPIKey);
delay(20000);
ThingSpeak.writeField(myChannelNumber, 3, ppm, myWriteAPIKey);
delay(20000);
 if(ppm<=100)
 {
digitalWrite(LED_GREEN,HIGH);
digitalWrite(LED_YELLOW,LOW);
digitalWrite(LED_RED,LOW);
 }
 else if(ppm<=200)
 {
digitalWrite(LED_GREEN,LOW);
digitalWrite(LED_YELLOW,HIGH);
digitalWrite(LED_RED,LOW);
 }
 else
 {
```

```
digitalWrite(LED_GREEN,LOW);
digitalWrite(LED_YELLOW,LOW);
digitalWrite(LED_RED,HIGH);
 }
delay(2000);}
```

CHAPTER 5

RESULTS

For the two sets of experiments that are outlined in the following sections, the functioning of the planned prototype has been examined.

EXPERIMENT 1

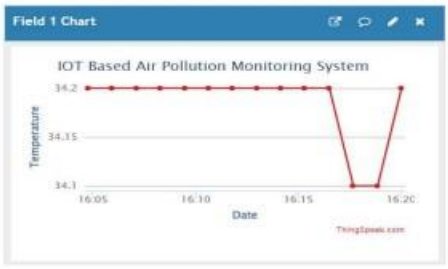
AIM:

To demonstrate the working of the system in a warm and humid outdoor atmosphere.

EXPERIMENTAL CONDITION:

The experiment was performed on a warm sunny day in a local outdoor area.

OBSEVATION IN THINGSPEAK CLOUD:



CONCLUSION:

To confirm the data, we used the Samsung mobile weather app as a source of information. With the temperature data, it matched with an error of +1.20, the humidity data with an error of +5, and the PPM data with an error of +0.11. Thus, we can say that the setup was successful in measuring the humidity and temperature in the vicinity of the setup area.

EXPERIMENT 2

AIM:

To demonstrate the working of the system in smoky conditions.

EXPERIMENTAL CONDITION:

The experiment was performed in the presence of smoke coming from an incense stick placed near the setup.

OBSEVATION IN THINGSPEAK CLOUD:

CONCLUSION:

The results show that the system is capable of quickly detecting smoke in the vicinity of the setup. To confirm the data, we used the Samsung mobile weather app as a source of information. The temperature data matched with an error of +1.80, the humidity data with an error of +4, and the PPM data with an error of -0.7. Thus, it can be said that this monitoring system allows us to identify the existence of smoke.

CHAPTER 6
CONCLUSION

This Internet of Things project measures and displays the Air Quality Index (AQI), as well as the temperature and humidity of the surrounding air. The project's data can be used to calculate the air quality in parts per million. The MQ135 sensor's drawback is that it cannot particularly detect the amount of carbon monoxide or carbon dioxide in the atmosphere; however, it does have the benefit of being able to detect dangerous gases such as smoke, CO, CO2, NH4, etc. Following a number of tests, it is clear that the apparatus can accurately measure the temperature in Celsius, the humidity in percentage, and the air quality in parts per million. Through the use of Google data, the trial outcomes are confirmed. Additionally, the led indicators assist us in determining the surrounding air quality. The project's inability to measure the part per million (ppm) levels of the various pollution components is a disadvantage, too. It would have been better if there had been gas monitors for various contaminants. However, with time, it would become more expensive to set up and unnecessary to have an air quality monitoring system. It's an IOT-based project, therefore uploading the data to the ThinkSpeak cloud will need a steady internet connection. It is therefore reasonable to draw the conclusion that the developed prototype can successfully monitor the air quality, humidity, and temperature of the surrounding atmosphere.