Smart Air Quality Monitoring System

Zain Ali Bhinder[1]
Dept. of Electrical and Computer Engineering,
Concordia University,
Montreal, Canada
z_bhinde@encs.concordia.ca[1]

***Abstract*—In the contemporary era, air pollution has become a major cause of concern across the globe. The major reason for this pollution can be pollutants coming from energy production, households, industry, transportation, and so on. Therefore, sustainable development goals agenda aims to pay special attention to air quality to obtain the comfort of residents in their daily activities. In order to measure the pollution index of specific place, air quality measurement system need to be developed to obtain the pollutants level of that region. This project describes the development of a low-cost air quality measurement system for monitoring C02 emission in the environment using ESP 32 microcontroller, MQTT and Node-RED. This is system will utilize an MQTT mosquito broker which will run on local laptop. A MQ- 135 Gas sensor will be interfaced with the ESP 32 micro-controller to detect Co2 gas, with the laptop performing functions of MQTT broker to transfer sensor data information on Node-RED dashboard to display concentration of pollutant at any time. In addition, a buzzer will also be connected to the micro controller to alarm the danger in the event of gas exceeds its safety limits.**

Keywords—Carbon Dioxide (CO_2), MQ-135, Microcontroller, Message Queuing Telemetry Transport (MQTT), Buzzer, Node-Red.

I. Introduction

Pure air is considered as the most essential component for human life as it has direct impact on the human life and helps nourishes lungs, blood and other organs with oxygen. Air is considered to be pure when it is not contaminated with any other harmful substances. However, finding clean air has become increasingly difficult specially in mega cities where air quality has exceeded the limit up to hazardous level. This climbing pollution does not have adverse effect on the environment and greenhouse effect but also being harmful to human health and increasing rate of morbidity and mortality. The reason for this rising pollution levels is the pollutant that can occur form natural source or because of waste product produced from different industries.

These pollutants can be classified into either primary or secondary. The former is usually produced from natural sources (lightning, volcanoes or wildfires), mobile sources (car, buses or airplanes), stationary sources (industries, power plants or sewerage treatment) in the form of carbon monoxide, carbon dioxide, ammonia, lead etc. The later cannot be emitted directly but formed due to interaction of primary pollutants such as nitric acid, trioxide or sulfur trioxide. Detecting these pollutants in the air is both complicated and complex as they are usually colorless and odorless.

One major type of greenhouse emission is carbon emissions due to release of carbon dioxide in the air following any human activity or process. According to the Environmental Protection Agency the six main sources of greenhouse gases are transportation, electricity production, industry, commercial and residential, agriculture and land use and forestry. The carbon emissions have been damaging the planet significantly causing global warming and climate change which results in extreme weather events such as tropical storms, wildfires, severe droughts and heat wave. These emissions are also very harmful to human health causing respiratory diseases and negatively impacting the cognitive performance.

It has become need of the hour to determine air quality to avoid high risk polluted places and enhance the life quality of human being. For this purpose, a low-cost air measurement system needs to be developed to obtain accurate levels of pollutants at a specific place. Furthermore, it is desirable to accommodate some communication technology through which the system can sent sensors readings from different location to a specific place through internet network. Therefore, this project focuses on the development of a low cost, high accuracy air quality system and presents a design of an ESP32 NodeMCU air quality system using Message Queuing Telemetry Transport (MQTT) and Node-RED. The smart home solution design utilizes an MQTT mosquito broker on laptop.

II. Literature review

As discussed, the environmental condition has become a grave predicament of this contemporary era. According to World Health Organization, approximately 90% people are exposed to pollution. This undesirable situation has put the life of human, animals and all the living species on planet earth in great danger. This concern can also be considered as one of the reasons of epidemic diseases that the world faces in recent few decades. These recent environmental changes have made the world to think in that way so that this environmental condition can be monitored and ultimately measures can be taken to control the hazardous emission. As a result, scientist and researchers have worked on developing a low-cost air quality monitoring system. Various technologies and approaches have been used by different researchers. Among these, Raspberry Pi, GSM based micro-controller and NodeMCU ESP32 are the popular one. In addition, the evolution of IoT has also made these applications more reliable, easy to access and secure. The implementation of IoT based system makes it easy to access data from any part of the world, provide security over the data accessibility and easy to maintain. All these technologies were reviewed through different papers and finally we decided to developed our IoT based air quality monitoring project using NodeMCU ESP32.

A. *Comparison with different topologies*

Initially the proposed system was proposed to be implemented with Bluetooth module and GSM module to achieve wireless communication with the system. However, this scheme is rejected because of number of disadvantages. Although Bluetooth can reduce the system cost as most of smart devices has built in Bluetooth module but its major drawback is that such system can only function for limited range and GSM module even though can be very useful to notify user the sensors reading through SMS but the cost of SMS can make the system very expensive.

Furthermore NodeMCU ESP 32 controller is preferred over Arduino Mega 2560 microcontroller which is also widely employed for monitoring purpose is because it is cheaper and more importantly it is equipped with the built-in WIFI module as compared to the Bluetooth module of Arduino controller. Thus, enhancing the controlling range of the system.

This system is initially proposed to be implemented with the ESP 32 microcontroller and Raspberry Pi which has an ARMv6 700 MHz single core processor where the MQQT broker and Node Red will be placed on the Raspberry Pi. Since we were unable to borrow the Raspberry Pi from university due to its unavailability, the MQQT broker and Node Red is then placed on the laptop.

III. CARBON DIOXIDE IN THE ENVIRONMENT

It is considered as one of the toxic gases in the environment having properties of being colorless, non-flammable having sharp odor and sour taste. From global warming perspective, it is regarded as the most crucial greenhouse gases. It is a normal component of exhaled breath and formed form the combustion of carbon-containing materials, in fermentation and respiration by all animals, fungi and microorganisms. Studies suggests that human activities have raised atmospheric Co2 by 50% since the beginning of industrial times. Therefore, detecting Co2 concentration in a specific place has become very vital as increased C02 gas can cause headaches, sleepiness, drowsiness and may lead to suffocation, coma or death due to oxygen depleted air caused by extreme CO2 concentration.

A. *Acceptable CO_2 Levels*

ASHRAE association has sets some standards for the Co2 concentration in the indoor air. It has indicated a list of co2 levels and its impact on human health.

CO2 concentration	**Health impact**
< 400 ppm	Normal outdoor concentration levels
401 – 1000 ppm	Usual concentration level at indoor place with good air exchange
1001- 2000 ppm	Falls in levels of complaints for drowsiness and poor air
2001 – 5000 ppm	Complaints of headache, sleepiness Lack of concentration, increased hearth rate and nausea
>5000 ppm	Oxygen deprivation level leads to coma, suffocation or death

Table 1: Impacts of Carbon Dioxide Concentration relative to humanhealth

As clearly depicted from the above table that the recommended co2 levels that can be brought from outside in to indoor air environment should remain close to 700 ppm to 800 ppm above which can harmful impact on the human health.

IV. SOFTWARE DESIGN

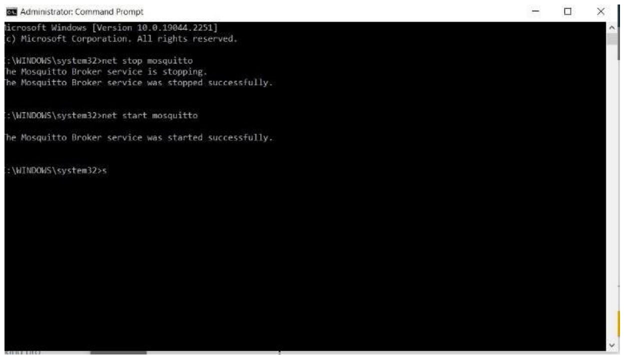
The following software have been used for the project:

A. *Arduino IDE*

Arduino IDE environment is used for developing code. The reason is because this software is open source and helps in code scripting which after completion can be uploaded to ESP 32 microcontroller. It can be run on windows, Linux and Mac OSX. This IDE facilitates communication of NodeMCU ESP 32 microcontroller and assist in viewing sensor data in serial monitor.

B. *MQTT Broker*

Message Queue Telemetry Transport, also known as MQTT, is a messaging protocol which is configured for such devices that has low bandwidth constraints. It is a light weight system that is used by client for publishing and subscribing to messages. The MQTT broker (i.e. Mosquito) is responsible for getting the messages, then filtering those messages, deciding in which messages the client might interested and then the messages are published to all the clients that are subscribed to the broker. In our project of air quality monitoring system, MQTT broker was installed on the laptop. For connecting Broker net start mosquito command is used on cmd.

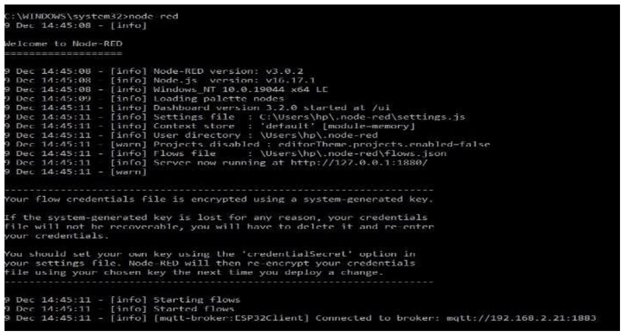


```
icrosoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

:\WINDOWS\system32>net stop mosquitto
he Mosquitto Broker service is stopping.
he Mosquitto Broker service was stopped successfully.


:\WINDOWS\system32>net start mosquitto

he Mosquitto Broker service was started successfully.


:\WINDOWS\system32>s
```

Figure 2 Mosquito broker

C. *Node-Red*

.Node-RED is basically a development environment based on JavaScript and uses Node.js. Primarily, it is used with IoT based system by calling different APIs together, utilizing hardware devices and online services. Node-RED was developed by IBM. It facilitates the user with interactive dashboard for the system containing various control buttons, graphs, charts or gauges etc. It is initialized through node-red command on cmd.

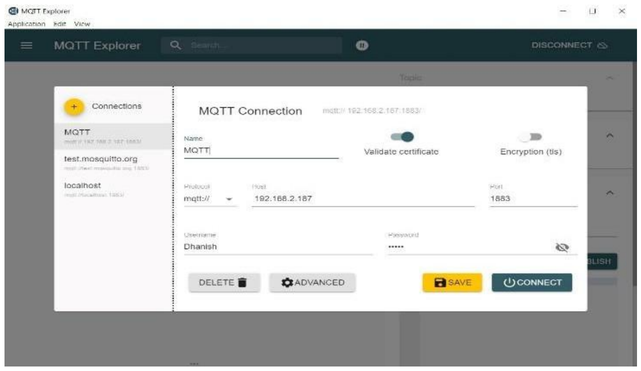


```
C:\WINDOWS\system32>node red
9 Dec 14:45:08 - [info]

Welcome to Node-RED
===================

9 Dec 14:45:08 - [info] Node-RED version: v3.0.2
9 Dec 14:45:08 - [info] Node.js  version: v16.17.1
9 Dec 14:45:08 - [info] Windows_NT 10.0.19044 x64 LE
9 Dec 14:45:09 - [info] Loading palette nodes
9 Dec 14:45:11 - [info] Dashboard version 3.2.0 started at /ui
9 Dec 14:45:11 - [info] Settings file  : C:\Users\hp\.node-red\settings.js
9 Dec 14:45:11 - [info] Context store  : 'default' [module=memory]
9 Dec 14:45:11 - [info] User directory : \Users\hp\.node-red
9 Dec 14:45:11 - [warn] Projects disabled : editorTheme.projects.enabled=false
9 Dec 14:45:11 - [info] Flows file     : \Users\hp\.node-red\flows.json
9 Dec 14:45:11 - [info] Server now running at http://127.0.0.1:1880/
9 Dec 14:45:11 - [warn]

---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

9 Dec 14:45:11 - [info] Starting flows
9 Dec 14:45:11 - [info] Started flows
9 Dec 14:45:11 - [info] [mqtt-broker:ESP32Client] Connected to broker: mqtt://192.168.2.21:1883
```

Figure 3 Node red

D. *MQTT Explorer*

MQTT Explorer is a cross platform application that makes it simpler to create MQQT clients. It enables to create clients to publish or subscribe to topics for testing MQQT devices and broker



mqtt:// 192.168.2.187:1883/

Name: MQTT
Validate certificate
Encryption (tls)
Protocol: mqtt://
Host: 192.168.2.187
Port: 1883
Username: Dhanish
Password: •••••

Figure 4 MQTT Explorer

E. *Fritzing*

Fritzing is an open-source initiative to develop amateur or hobby CAD software for the design of electronics hardware, intended to allow designers and artists to build more permanent circuits from prototypes.

V. Architecture and Component of System

The high-level architecture of the system can be seen in the figure below. The proposed system is composed of 4 major steps namely sensing, controlling, communicating and output display. This architecture presents merits of being low-cost, affordable, low resource needs and open source. Each of these steps are briefly discussed below.

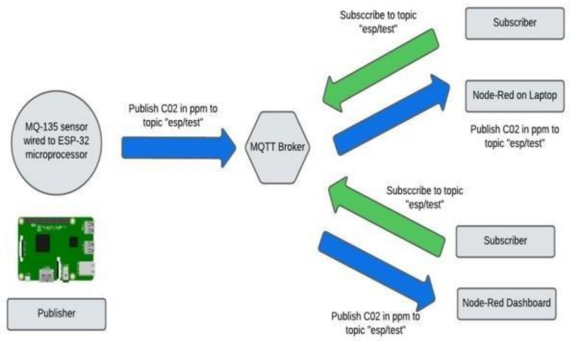

MQ-135 sensor wired to ESP-32 microprocessor
Publisher
Publish C02 in ppm to topic "esp/test"
MQTT Broker
Subsccribe to topic "esp/test"
Subscriber
Node-Red on Laptop
Publish C02 in ppm to topic "esp/test"
Subsccribe to topic "esp/test"
Subscriber
Node-Red Dashboard
Publish C02 in ppm to topic "esp/test"

Figure 5:High Level Architecture

A. *Real Time Sensing Unit*

Sensor can be defined as the device used for detection of specific parameters and convert the parameter form in to electrical signals or other required forms based on certain principle and process the output information for storage, display or control requirements. In this air quality measurement system, MQ-135 gas sensor is employed for sensing real time Co2 gas in the environment. MQ-135 gas quality sensor module operates at 5V and normally used to detect and measure variety of gases such as NH3, NOx, alcohol, benzene, smoke and CO2. The senor has a digital and analog output pin. The digital pin gives high output when the level of these gases goes beyond a threshold output. However, it requires some preheating before it give accurate results.

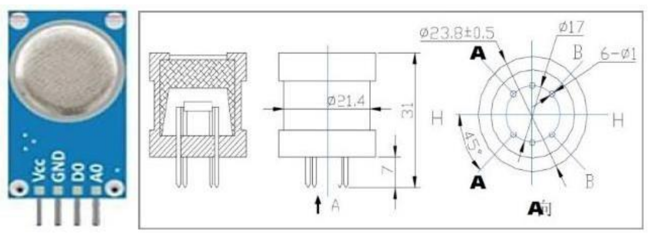


Figure 6: MQ-135 Sensor Pin Configuration

This sensor shows excellent sensing characteristics for variety of different gases such as carbon dioxide, Phenol, Toluene, Benzene and air. MQ135 sensor characteristics for various gases can be seen in the plot below where it can clearly observe that four gases including co2 has shown impressive response. Thus, MQ135 is selected to detect carbon dioxide with high precision of calibration.

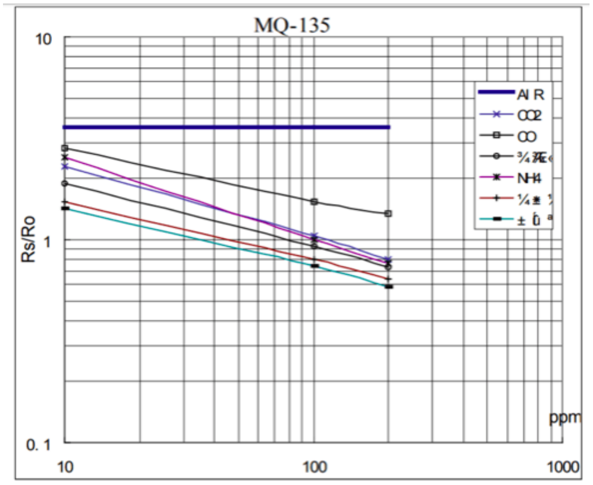


Figure 7: Sensitivity Characteristics for different cases

B. Controlling Unit

For better reliability and stability of the system, we use ESP 32 microcontroller in the control unit. The reason of choosing NodeMCU ESP 32 controller over Arduino is that it is cheaper and Arduino mega uses Bluetooth module which is unsuitable for most IOT application due to its limited features. On the other hand, NodeMCU ESP 32 microcontroller is equipped with WIFI module which is essential for the communication purpose and helps in displaying real time sensor values to remote location. Due to their excellent performance of supporting real-time data processing, they are widely employed for internet-of-things based application. The output of gas sensor is connected to the input of microcontroller. The processor identifies the analog values of the sensors that are interfaced with the analog pins of microcontroller and are process through a program written in Arduino IDE. The code work on the algorithm of signal receives and store mechanism. A buzzer is also attached in the microprocessor and processor will sent an active signal to buzzer whenever value exceeds to a desired value.

C. Communication

In order to facilitate communication between sensor installed in the micro controller and Node red dashboard, MQQT is employed which is designed for lightweight publish/subscribe messaging transport ideal for connecting remote devices with a small code footprint and minimal network bandwidth. The communication methodology is based pub sub method in which a MQQT broker is served as a medium to facilitate communication between publisher and subscriber. In this case, ESP 32 controller act as a publisher which will sent the sensor readings to the broker using a specific topic "ESP/test" and the subscriber will subscribe to this topic to receive the sensor readings.

D. Output Unit

The output unit consist of a buzzer module which is connected with the micro controller. When the value of C02 exceeds the predefined safe limit in the code, buzzer will receive a signal from the controller which will be turned on and indicate the alarming gas level situation.

E. Flowchart

The flowchart of the proposed system can be shown in the figure below which will indicate the sequential activity of the whole process.

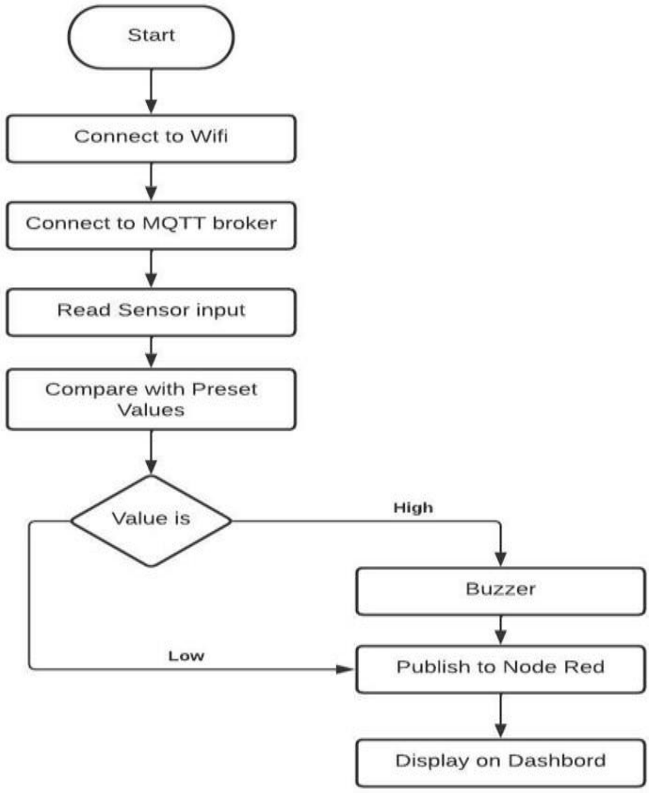


Figure 8: System Flowchart

VI. SCHEMATIC AND WORKING

The circuit schematic diagram displays how to interface the ESP32 microcontroller with piezo buzzer and MQ135 air Quality sensor. The MQ135 sensor has four pins namely GND, VCC, A0, and D0. The GND and VCC pin of sensor is connected with Ground pin and VCC pin is connected with 3.3V pin of the microcontroller respectively. A0 pin which is used for analog output is connected with the GPIO 32 of the sensor which has ADC to convert the analog data received from the sensor.

GPIO pins are rated for 3.3V so any voltage higher than 3.3 V will fry the microcontroller board. So, to avoid this from happening, a voltage divider circuit is connected at the A0 pin to reduce the output voltage. Piezo Buzzer is a low-level trigger which means that it is turned on when control signal is set as low and turned off when control signal is set as high. The buzzer is connected with the GPIO 12 of the ESP32. Using Arduino IDE, sketch is uploaded into the microcontroller board.

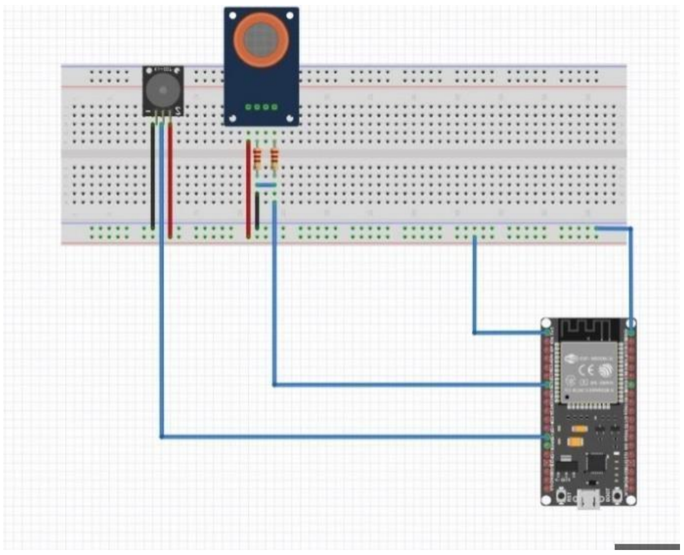


Figure 9: Circuit Schematic

VII. Hardware Implementation and results

The figure below shows the hardware implementation of the air quality monitoring system synchronized with the MQTT broker and Node-RED where MQ 135 gas senor is connected with ESP32 microcontroller. And the buzzer is connected to the output to provide the sensors reading and perform the functionality described in the above flow chart.

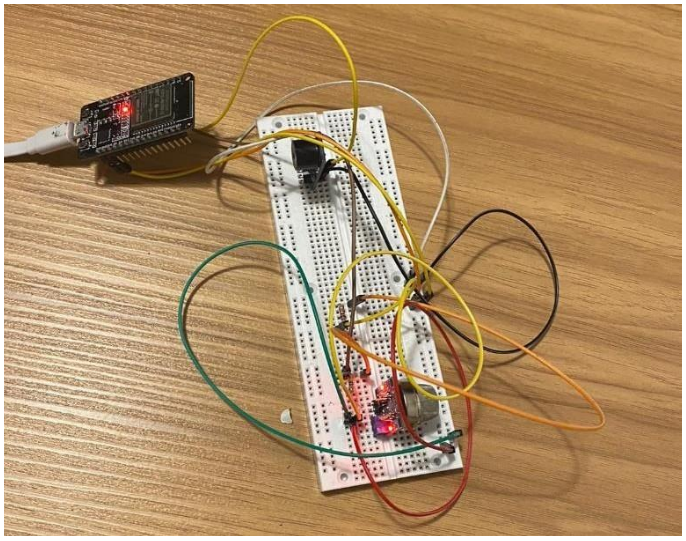


Figure 10: Hardware Prototype

A. *Node-Red Flows*

Initially, the sensor publishes reading to the MQTT broker through micro controller using topic "ESP/test" after establishing WIFI connection. Then the Node Red flow helps in receiving the data through wiring MQTT IN and debug component which will subscribe to the same topic "ESP/test". For the data visualization, sensor data is displayed on the Node-Red web dashboard on real time basis through the gauge function. The Node-Red flow of the system can be observed in the figure below.

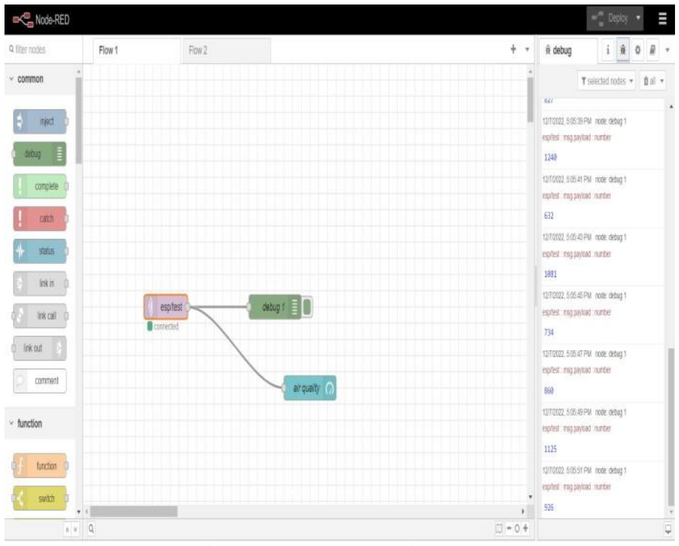
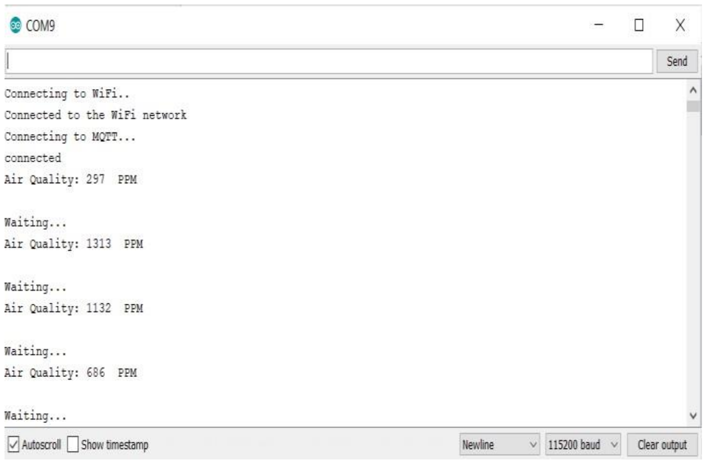


Figure 11: Node-Red Flow

B. *Arduino Serial Monitor Results*

The readings received from the NodeMCU ESP32 controller are displayed on the Arduino environment where it can be clearly observed that the controller has first established its connection with the WIFI and try to be connected with the MQQT broker. After successful connection the readings receiving in serial monitor from sensors can be observed which depicts that real time serial monitoring of carbon concentration with low latency.



```
Connecting to WiFi..
Connected to the WiFi network
Connecting to MQTT...
connected
Air Quality: 297  PPM

Waiting...
Air Quality: 1313  PPM

Waiting...
Air Quality: 1132  PPM

Waiting...
Air Quality: 686  PPM

Waiting...
```

Figure 12: Arduino IDE Serial Monitor for monitoring real time readings

C. *Node-Red Dashboard*

After forming connection with MQQT broker, broker will send the sensor information on the topic “ESP/test”. So firstly, a subscriber is formed on Node-Red by the same topic “ESP/test” which will start to received sensor reading in serial fashion. Then this reading is displayed in a visual manner through gauge at Node Red Dashboard after subscribing at the above-mentioned topic. The Node-Red dashboard displaying the real time sensor readings can be observed in the fig below.

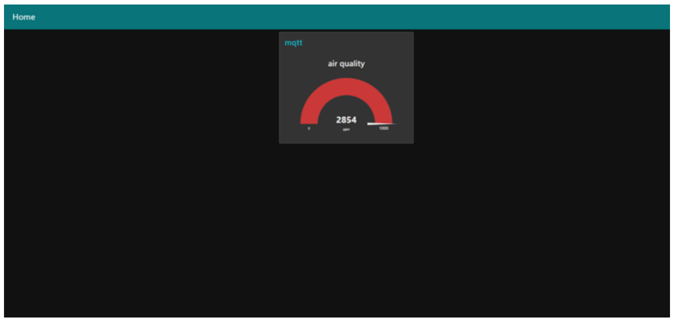


Figure 13: Node-Red Dashboard for displaying real time readings

VIII. Conclusion

This project provides the designing and hardware implementation of a portable, user-friendly and economical air quality monitoring system using IoT principles. This system can be implemented for various applications for the real tome monitoring of a Co2 concentration such as monitoring pollution along roadside or traffic spots, monitoring pollution level generated from different industries, getting pollution concentration at indoor or home application or obtaining pollution data for a specific location. For this purpose, MQ 135 gas sensor is connected to the NodeMCU ESP32 controller and a connection is formed with MQTT broker to publish sensor readings to the broker. The obtained data is then displayed on the Node red dashboard for monitoring purpose.

For future purpose, mobile phone notification system can also be implemented with this system through developing an application which will send these readings to the user to alert alarming air level in the environment even at remote place. Thereby, enhancing the security features of the system. Furthermore, the controller has room to accommodate different sensors so the system can be expanded with different sensors to enhance the reliability of the system. After slight modification, this system can be developed for future commercialization purpose.

References

[1] Carbon Dioxide and Carbon Monoxide Level Detector Ahmed Abdullah Ibrahim2018 21st International Conference of Computer and Information Technology (ICCIT)Year: 2018 | Conference Paper | Publisher: IEEE.

[2] Design and Implementation of A Low-Cost Air Quality Measurement Instrumentation Using Internet-of-Things Platform and Cloud-based Messaging Service Udin Komarudin; Heli Subarli; Kusnandar; Asep Najmurrokhman 2021 1st International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS)Year: 2021 | Conference Paper | Publisher: IEEE.

[3] Design of Indoor Air Quality Monitoring Systems Tigor Hamonangan Nasution; Ainul Hizriadi; Kasmir Tanjung; Fitra Nurmayadi2020 4rd International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM).

[4] IoT Driven Solution for Indoor Air Quality Monitoring System to Develop a Smart Healthcare Environment: A Review Based Study Nadia Shahrin Chandni; Mohammad Ismail; A. K. M. Muzahidul Islam 2022 International Conference on Decision Aid Sciences and Applications (DASA) Year: 2022 | Conference Paper | Publisher: IEEE.

APPENDIX A: SOURCE CODE

```
#include <MQ135.h>
#include "MQ135.h"
#include  <WiFi.h>
#include <PubSubClient.h>
#define sensor        32                     //pin for analog input
#define buzzer  12                           //pin for buzzer
const char* ssid = "BELL918";
const char* password = "2A326E47FD65";
const char* mqttServer = "192.168.2.21";
const int mqttPort = 1883;
const char* mqttUser = "Dhanish";
const char* mqttPassword = "12345";
WiFiClient espClient;
PubSubClient client(espClient);
void setup() {
 pinMode(sensor,INPUT);                    //MQ135 analog feed set for input
  //pinMode(digTrigger,INPUT);                //MQ135 digital feed set for input
  pinMode(buzzer,OUTPUT);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
  client.setServer(mqttServer, mqttPort);
  while (!client.connected()) {
    Serial.println("Connecting to MQTT...");
    if (client.connect("ESP32Client", mqttUser, mqttPassword )) {
      Serial.println("connected");
    } else {
      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(2000);
    }
  }
}
void loop() {
  client.loop();
  MQ135 gasSensor = MQ135(sensor);
    int air_quality = gasSensor.getPPM();
    Serial.print("Air Quality: ");
    Serial.print(air_quality);
    Serial.println(" PPM");
    Serial.println();
    char mqtt_payload[30] = "";
  snprintf (mqtt_payload,30, "%1d",  air_quality ) ;
  client.publish ( "esp/test", mqtt_payload ) ;
  if ( air_quality > 1000 ){
     digitalWrite( buzzer, LOW );
   }
  else {
     digitalWrite(buzzer,HIGH);
    }
    Serial.println("Waiting...");
    delay(2000);
}
```