LOVELY PROFESSIONAL UNIVERSITY
PHAGWARA, PUNJAB

REPORT
ON
AIR WATCH
THE AIR QUALITY TRACKER

Master Of Computer Application (MCA)

ECE662 - IOT WITH NODEMCU

Submitted By :-

[Raja Saha 12209471]

[Minakshi Sahu 12213937]

[Anurag Verma 12202995]

Submitted To :-

[Dr. Akhil Gupta 20239]

Table of Contents.

Project Title

Air Pollution Monitoring System

Abstract- One of the world's biggest problems today is reducing air pollution. Most of the time, organizations and educational Developing nations experience tainted environments due to inadequate planning and facilities. Teachers and students in a classroom may experience health problems as a result of extended exposure to a given setting. Numerous people experience asthma, and it gets quite challenging for an individual when having an episode of asthma. A cheap price A gadget for monitoring the environment is created to identify various pollutants, such as CO, CO2, noise, temperature, and relative humidity. A cheap price A gadget for monitoring the environment is created to identify various pollutants, such as CO, CO2, noise, temperature, and relative humidity. It has been noted that identical sensors sensing identical gasses provide varying values despite the fact that the sensors' sensitivity is suitable, thus it has additionally attempted to carry out calibration of the sensors by the use of machine learning. Additionally, it can be employed to identify pollutants within interior spaces and alert the person in advance to potential asthma attacks consumer.

1. Introduction.

Indoor pollution has a significant impact on city dwellers and cannot guarantee a healthy atmosphere. Several of the dangerous indoor contaminants consist of VOC, CO2, CO, and particulate particles. A human spends the majority of his time indoors, therefore the quality of Control is crucial in many ways. The bad air quality could cause health problems, a decrease in focus at work, even early demise, etc. It gets quite challenging for an individual to inhale contaminated air since he possesses the highest danger of suffering a severe asthma episode. Despite air, Although there are quality monitoring stations, these are extremely costly, non-portable, and tailored notably for those who suffer from asthma. As stated by the Ozone from the US Environmental Protection Agency asthma if the Quality Index (IQI) is 101 or above, it poses a serious risk to those with asthma, requiring them to modify their activities, medications, and even their lifestyle of existence. Additionally, symptoms get worse even when ozone levels between 50 and 100 are moderate. Asthma is a chronic illness with symptoms that intensify quickly ('attacks') that are erratic and could prove deadly. Asthma attack prediction models need high sensitivity to reduce the risk of death and high specificity to prevent the needless prescription of preventative drugs that have possible negative side effects happenings. Thus, a precise sensor prediction is needed when it comes to making it unique for those with asthma. To address This device, which is sensor-based, has been designed. The primary components are sensors. The primary components for detecting the surroundings are sensors. data. For data validation of these in real-time monitoring sensors and for determining the accuracy of the information The system's sensor calibration is very important. The info obtained by poor-quality sensors may have an impact on the caliber of the method of making decisions. A calibration procedure entails utilizing the measurement unit's known inputs and by doing so, the system's overall inaccuracy can be observed. Next, such mistakes are fixed by reducing the errors to none. To put it briefly, calibration entails comparing obtained and conventional measurements. Once the values have been obtained We feed this information into an algorithm for machine learning. which analyzes and forecasts various data sets likelihood of an asthma attack and provides advance notice as well as suggestions for users.

PROBLEM STATEMENT

High sensitivity is needed to reduce the risk of death from asthma episodes, and high specificity is needed to prevent the overprescription of prophylactic drugs that have a high chance of unfavorable side effects. In order to address this issue and customize it for asthma patients, an accurate sensor prediction is necessary. To this end, a sensor-based device has been designed. The primary components for gathering environmental data sensing are the sensors. The calibration of sensors is particularly important for real-time monitoring data validation of these sensors and for determining the precision of the data in the system. The quality of the decision-making process may be impacted by the data collected by subpar sensors.

It is feasible to construct an affordable air quality monitoring system using ESP32. Energy consumption: This Internet of Things circuit uses less energy while operating at currents of less than 100 mA.

RELATED WORK

Authors: N Jahnavi Reddy, Nikitha R Sherkhane , Roja Pravallika , Mrs. J Ruby Dinakar.

In this study, they suggest a three-unit system as system hardware, which uses low-cost environment monitoring devices to detect different polluting gasses such as CO, CO2, VOC, and particulate matter. The experiment is carried out in a confined space. They used IOT Cloud (ThingSpeak) to store data and then a software component to calibrate the data.

Disadvantages

Air Quality Stations are both expensive and immobile.

Sensor calibration is required.

Devahema, P.V. Sai Surya Vamsi, Archit Garg, Abhinav Anand, and Desu Rajasekhar Gupta are the authors.

In this work, they employed the most dynamic and efficient gas sensor (i.e. MQ135), which is capable of sensing various harmful gases such as NH3, NOx, alcohol, Benzene, smoke, and CO2 and displaying the results in PPM (Parts Per Million).As a result, it is a dynamic gas sensor for our air pollution monitoring system. When linked to an Arduino, it will detect all gases and report the pollution level in PPM (parts per million). The output of the MQ135 gas sensor will be in the form of voltage levels, which must be translated into PPM.

Disadvantages

If you want to see the current Air Quality Value in PPM, you must refresh the page several times.

A local server must be set up to demonstrate its functionality and to monitor the air quality from anywhere in the world; port 80 (used for HTTP or internet) is required.

In real-time, IoT-based air and noise pollution monitoring in urban and rural areas, as well as critical zones such as schools and hospitals

Mahantesh B Dalawai, Mr. Pradeep S, and Dr. Siva Yellampalli [3] are the authors.

To implement his project, Mahantesh B Dalawai employed a Renesas RL78 series R5F100LEA microcontroller with Flash ROM 64KB, RAM 4KB, and Data Flash 4KB, as well as a High speed on-chip oscillator, Self-reprogrammable under software control, 58 GPIO's, and so on. If you want to see the current Air Quality Value in PPM, you must refresh the page several times. A local server must be set up to demonstrate its functionality and to monitor the air q uality from anywhere in the world; port 80 (used for HTTP or internet) is required.The pollution level at each sensor node can be sent to a server via a GSM/GPRS system, or it can be displayed on a huge display next to a square. The basic goal is for individuals to have more than one option way to a destination. If a person is aware of pollution information ahead of time, he can take a safe way while also controlling pollution.

Disadvantages

Sensors for monitoring dust, noise, smoke, and other factors are not installed.

It takes time since the pollution level from each vehicle on the road must be read.

2. Modules/Components.

If you want to see the current Air Quality Value in PPM, you must refresh the page several times. A local server must be set up to demonstrate its functionality and to monitor the air q uality from anywhere in the world; port 80 (used for HTTP or internet) is required. The suggested system makes use of some hardware components. The following are the components that are used to construct the system.

Sr No.	Component's Name	Quantity's	Component's Price
1	ESP32	1	425
2	PM2.5 Sensor	1	350
3	MQ-7	1	180
4	MQ135	1	195
5	Buzzer	1	14
6	Bread Board	2	70
8	Wires	30	45
9	Lcd	1	125

Total Cost : 1404

2.1. ESP32

The ESP32 is a popular IoT learning tool. This provides a complete Linux system on a compact platform for a very low cost. The ESP32 uses GPIO to link device sensors and actuator pins. ESP32 and IoT combine to form a breakthrough technology for healthcare creativity. The ESP32 has been meticulously designed. Featuring integrated antenna switches, RF-balun, control amplification, low-noise amplifier, and filters, as well as power management modules. It is capable of functioning as a complete stand-alone system. scheme or as a slave to a host MCU to reduce overhead Interaction between the main application processor and the user. EPS32 is capable of conversing with others. communicate with other Wi-Fi and Bluetooth devices via its Interfaces such as SPI/SDIO or I2C/UART are available.

The app has the following features, indices of air quality for a specific city using real-time computation, air quality daily forecasts, etc.

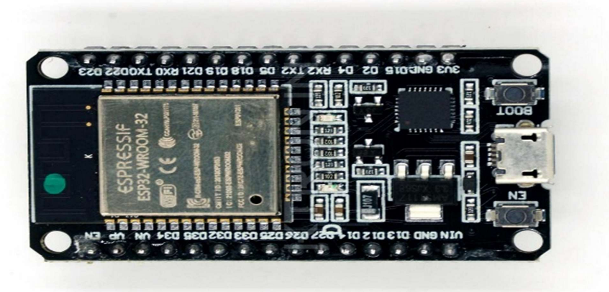


Figure 1.1 ESP32

2.2 PM2.5 Sensor

A PM2.5 sensor or Particulate Matter 2.5 sensor is a device used to measure air pollution levels by detecting and quantifying airborne particles up to 2.5 micrometers in diameter. These small particles can be harmful to human health because they can penetrate deep into the respiratory tract. PM2.5 sensors usually work on the principles of light scattering or light absorption. Here is a brief description of how they work and reduce pollution levels:

Figure 1.2 PM2.5

- **Method of light scattering:** In this method, a laser or LED light source sends a beam of light through an air sample. - Particles in the air scatter light in different directions. - The sensor contains a light sensor that measures the intensity of scattered light from different angles. - The amount of scattered light is proportional to the concentration of PM2.5 particles in the air. - The algorithm then calculates the PM2.5 concentration based on the light scattering data.

- **Light absorption method:** In this method, the light source sends a beam of light through the air sample. - Particles in the air absorb some light. - The sensor measures the decrease in light intensity after it has passed through the air sample. - The amount of absorbed light is directly proportional to the concentration of PM2.5 particles. - An algorithm based on light absorption data is used to calculate the concentration of PM2.5. The formula for calculating the pollution level (PM2.5 concentration) may vary slightly depending on the specific sensor and its calibration. It usually includes some adjustment factors and mathematical algorithms to convert the raw data (measurements of light scattering or absorption) into relevant PM2.5 concentrations in micrograms per cubic meter ($\mu g/m^3$). The calibration process involves comparing the sensor and measurements to reference instruments to ensure accuracy. Different sensor manufacturers may have their own algorithms and calibration methods. Users of PM2.5 sensors usually receive calibrated data and can monitor air quality in real-time or over a period of time. It is important to note that the accuracy and precision of PM2.5 sensors can vary, so it is important to use well-calibrated and reliable sensors to accurately monitor air quality.

2.3. MQ-7

The MQ-7 sensor is a type of gas sensor commonly used to detect carbon monoxide (CO) in the air. It is widely used in applications where the monitoring of CO levels is important for safety, such as in homes with gas heaters or in industrial environments with potential CO emissions. Here is a brief description of how the MQ-7 sensor works and how it calculates CO concentration:

Figure 1.3 MQ7

- **Operation of the MQ-7 sensor:** Gas Detection: The MQ-7 sensor contains a semiconductor material (typically SnO2 - tin dioxide) that is sensitive to the presence of carbon monoxide. When CO gas is present, it interacts with the surface of the sensor material, causing its electrical resistance to change.

- **Heating element:** The sensor has a built-in heater that keeps the sensor material at a constant temperature. This ensures that the sensor remains stable and responds to changes in CO concentration.

- **Measurement:** The resistance of the sensor material is measured and this resistance value changes when the CO concentration in the air changes. A higher concentration of CO decreases the resistance of the sensor and a lower concentration of CO increases the resistance of the sensor.

- **Calculation of CO concentration:** The MQ-7 sensor does not directly provide CO concentration in ppm (parts per million). Instead, it provides a voltage or resistance value that corresponds to the ambient CO level. A calibration process is required to convert that voltage or resistance reading to CO concentration. Calibration involves determining the relationship between sensor and output and actual CO concentrations using reference gases. Once the sensor is calibrated, you can use the formula to estimate the CO concentration: CO concentration (ppm) = (Sensor output − Ro) /sensitivity

- **Sensor Output:** The actual voltage or resistance reading from the MQ-7 sensor. Ro (clean air resistance): The resistance of the sensor when exposed to clean air, usually measured in a controlled environment. Sensitivity: This is the sensor's sensitivity factor, which indicates how much the sensor's resistance changes in response to a known CO concentration. This is determined during the calibration process. It is important to note that proper calibration and maintenance of the MQ-7 sensor is critical to accurately measuring CO concentration. The sensor may also require periodic recalibration to account for changes in its sensitivity. In addition, the sensor is specific for the detection of carbon monoxide and its formula is not suitable for measuring other gases.

2.4. MQ135

The MQ135 sensor is a popular gas sensor used to detect various air pollutants such as ammonia (NH3), carbon dioxide (CO2), methane (CH4), benzene and other volatile organic compounds (VOC). It is often used in applications where it is necessary to monitor indoor air quality or detect pollutants in the environment. Here is a brief description of how the MQ135 sensor works and how it calculates the contamination level:

Figure 1.4 MQ135

MQ135 sensor function:

- **Gas detection:** The MQ135 sensor contains a semiconductor that is sensitive to various gases. When these gases are in the environment, they interact with the sensor material, causing changes in its electrical conductivity. **Heating element:** Like many gas sensors, the MQ135 has a built-in heater to keep the sensor material at a constant temperature. This ensures the stability and sensitivity of the sensor to changes in gas concentration.

- **Measurement:** The sensor measures changes in its electrical resistance due to the presence of certain gases. The degree of resistance change is proportional to the target gas concentration.

- **Calculation of pollution levels:** The MQ135 sensor provides a voltage or resistance value corresponding to the detected gas concentration. However, it does not give the gas concentration directly in parts per million (ppm). To obtain gas concentration values, the sensor must be calibrated and a conversion formula used. The calibration process involves exposing the sensor to known concentrations of the target gas in a controlled environment. Once the sensor is calibrated, you can use the formula to estimate the gas concentration: Gas concentration (ppm) = CO concentration (ppm) = (Sensor output − Ro) /sensitivity

- **Sensor Output:** The actual voltage or resistance read by the MQ135 sensor. Ro (Clean Air Resistance): The resistance of the sensor when exposed to clean air (usually measured in a controlled environment). Sensitivity: This is the sensitivity coefficient of the sensor, which indicates how much the resistance of the sensor changes in response to a known target gas concentration. Sensitivity values are determined during the calibration and may vary for different gases. The exact calibration and sensitivity factors depend on the gas you want to detect. In addition, some MQ135 sensors are pre-calibrated for certain gases, such as CO2, but the sensor and formula vary depending on the target gas. Keep in mind that periodic recalibration may be necessary to maintain the accuracy of the MQ135 sensor over time. The MQ135 is versatile but is typically used to detect one specific gas at a time. If you need to measure several gases at the same time, several sensors or more complex equipment may be required.

2.5. LCD(Liquid Crystal Display)

LCD, or liquid crystal display, is a flat-screen technology commonly used in various electronic devices such as televisions, computer monitors, smartphones, calculators, and other devices. LCDs are preferred for their thin and energy-efficient design, which makes them ideal for portable and battery-powered devices. Here is a brief explanation of how an LCD screen works:

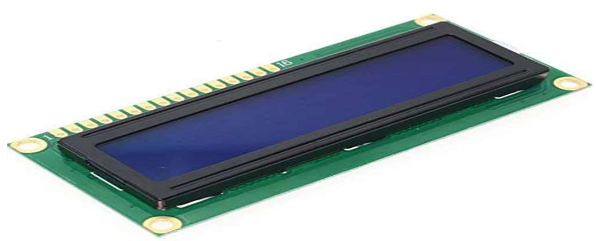


Figure 1.5 LCD

The working principle of the LCD screen:

- **Liquid Crystals:** At the heart of an LCD screen is a layer of liquid crystals, which are organic molecules that have the unique ability to change direction when an electric field is applied. These liquid crystals are usually placed between two transparent electrodes and two layers of polarizing material.

- **Polarization:** There are polarizers on the top and bottom of the LCD panel. Polarizers are filters that let light pass through only in a certain direction. The upper polarizer is oriented vertically, while the lower one is oriented horizontally. These polarizers help guide the light and its passage.

- **Electric field:** When an electric voltage is applied to the electrodes, the liquid crystals between the electrodes align with the direction of the applied electric field. This changes the polarization of the light passing through them. Rotation of light: Liquid crystals can rotate the plane of polarization of incoming light. The amount of rotation can be adjusted by adjusting the supply voltage. This rotation allows the LCD to control the amount of light passing through each pixel.

- **Color filters:** Most LCD monitors use color filters to display colors. Each pixel is divided into red, green, and blue subpixels. By precisely adjusting the light passing through these subpixels and changing their intensity, the entire range of colors can be displayed.

- **Liquid Crystal States**: Depending on the liquid crystal orientation of a pixel, it can be in one of two states: andquot;onandquot; (light transmission) or andquot;offandquot; (blocking light). These modes are controlled by the pixel voltage.

- **Pixel Grid:** An LCD panel consists of a grid of thousands or millions of pixels. Each pixel can be controlled individually to display specific colors and brightness levels. Backlight: Most LCD monitors have a backlight behind the LCD panel. This backlight provides the necessary illumination for the screen. The liquid crystals modulate the light from the backlight to create the image.

- **Image creation:** by selectively adjusting the orientation of the liquid crystals of each pixel, the screen creates an overall image. Combining different colors and brightness levels in different pixels results in text, images, videos, and graphics. In short, the LCD monitor controls the orientation of the liquid crystals in each pixel, which in turn modulates the light flow. This modulation creates the images and colors you see on the screen. Thanks to polarizers, color filters, and precise voltage regulation, LCD screens can produce sharp, colorful, and energy-efficient displays in many electronic devices.

2.6. Piezo buzzer:

A piezo buzzer or piezoelectric buzzer is a simple acoustic component used to generate sound and is commonly found in various electronic devices and applications. It produces sound through the piezoelectric effect, which is the ability of certain materials to generate an electrical charge when subjected to mechanical stress. Here is a brief description of how the piezo buzzer works:

Figure 1.6 Buzzer

Operating principle of the

- **Piezoelectric element:** At the core of the piezo buzzer is a piezoelectric ceramic or crystal element. This element can warp or deform when electrical stress is applied to it, and conversely, it can develop electrical stress when subjected to mechanical stress.

- **Signal input:** When AC voltage is applied to the piezoelectric element, it rapidly switches between positive and negative voltage. This voltage causes the piezo cell to oscillate at the same frequency as the AC voltage.

- **Mechanical Vibration:** When a piezo element vibrates, it creates mechanical pressure waves or vibrations in the surrounding air. Those vibrations create sound waves that travel outward from the buzzer.

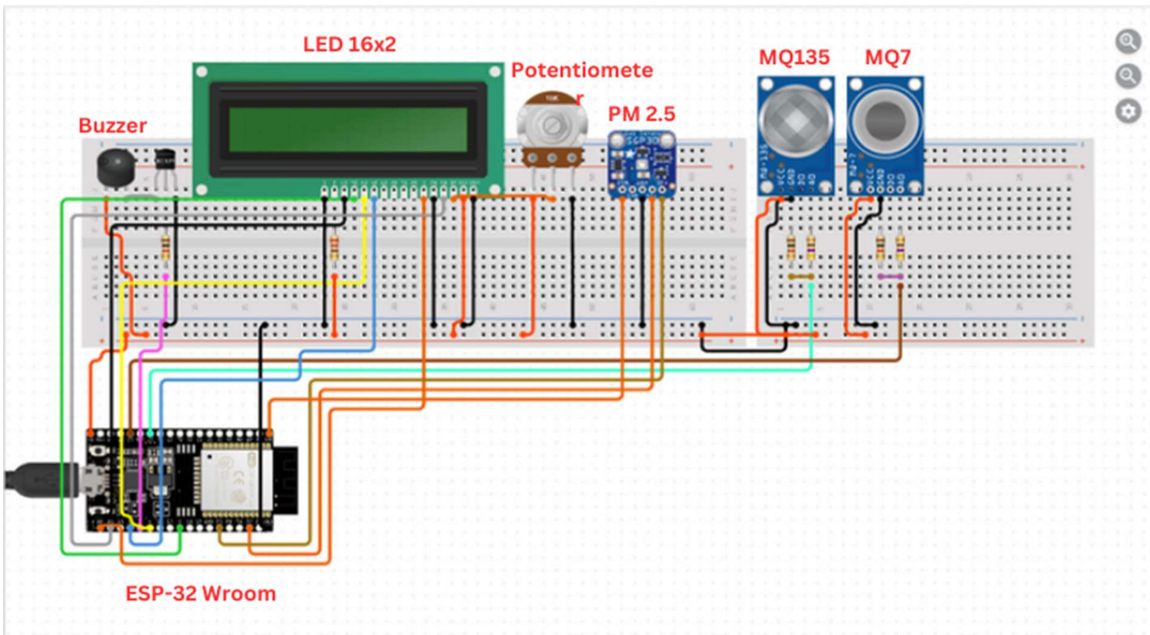
- **Sound output:** Rapid vibration of the piezo cell produces an audible sound or tone, usually in the form of a simple beep or beep. The frequency of the sound corresponds to the frequency of the alternating voltage applied to the piezo element. Volume control: The volume or loudness of the sound produced by the piezo buzzer can be adjusted by adjusting the amplitude (voltage level) of the AC signal applied to the piezo element. A higher amplitude results in a louder sound, while a lower amplitude results in a quieter sound.

Piezo composites are often used in a variety of applications, including:

- **Alarms and Timers**: These provide audio notifications of various events and alarms. Electronic Devices: Used to indicate button presses, low battery alarms, and other user feedback. Industrial devices: These act as warning or status indicators. Security systems: These are used to report intrusions or system problems. Car apps: These provide seat belt reminders and other vehicle alerts. Piezo buzzers are preferred for their reliability, simplicity, and compact size. They are generally considered a cost-effective solution for creating beeps and notifications in many electronic devices.

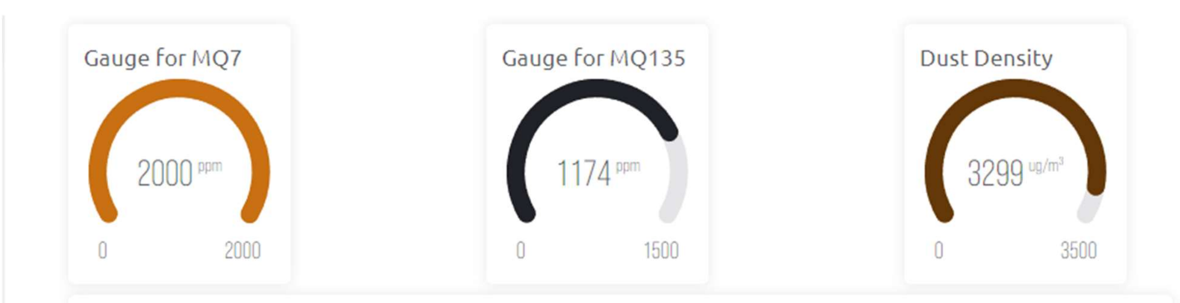
3. Working on the project

Block Diagram.

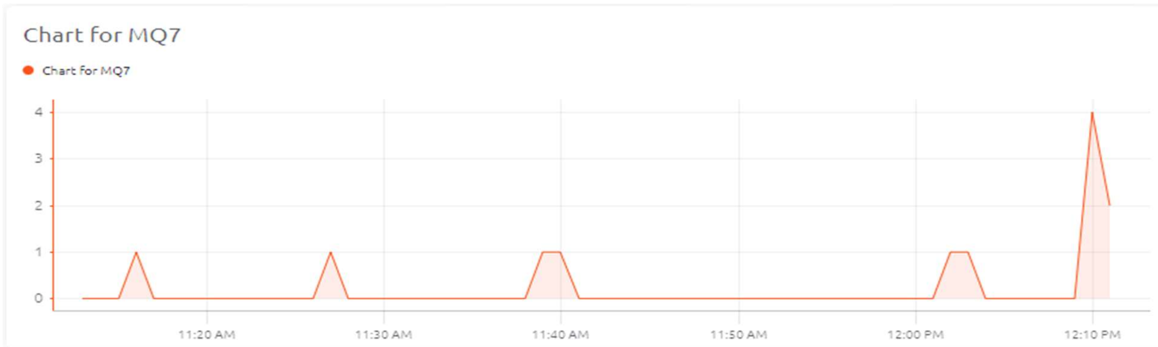


4. Result and Discussion.

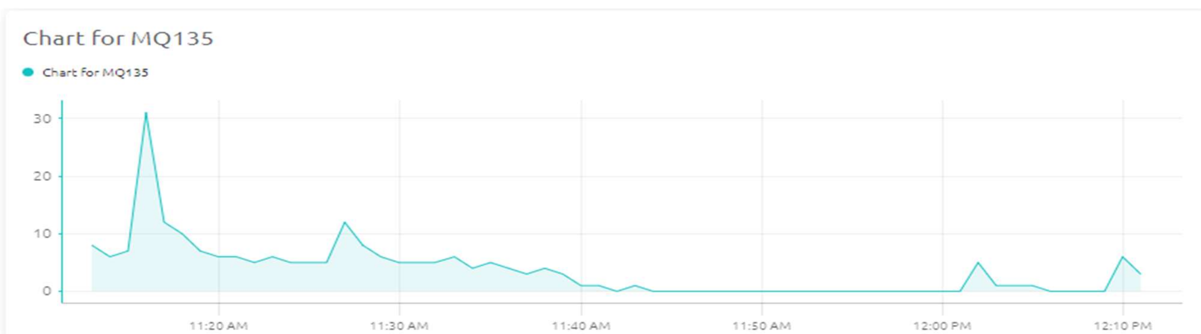
- **Gauge for MQ7/MQ135/Dust Density.**



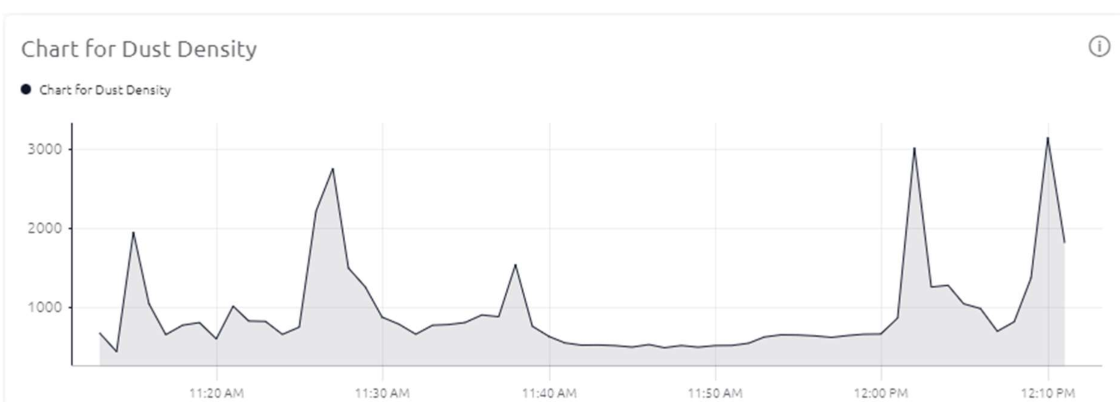
- **Chart for MQ7**



- **Chart for MQ135**



- **Chart for dust Density.**



5. Future Scope

- **Enhanced Data Analytics and Machine Learning:** In order to successfully reduce air quality issues, future air pollution monitoring systems will be able to predict pollution patterns, analyze the effects of different sources, and provide early warnings. This will be accomplished by utilizing sophisticated data analytics and machine learning algorithms.

- **Integration with Smart City Initiatives:** Real-time air quality data can be utilized in smart city projects to optimize energy use, traffic management, and urban planning, resulting in more livable and sustainable cities. These systems can play a key role in these initiatives

- **IoT-Based Distributed Networks:** By utilizing the Internet of Things (IoT), air quality sensor networks can be expanded to give wide coverage, even in isolated locations. Partnerships between public and commercial organizations can create a strong and extensive surveillance network.

- **Health Impact Assessment:** In order to shield populations from pollution-related health problems, future systems can create prediction models that connect data on air quality to public health outcomes. This will allow for early intervention and preventative measures.

- **Source Identification of Pollution:** State-of-the-art systems are able to incorporate source apportionment methods in order to pinpoint the precise sources of pollution, which helps the regulatory bodies target the polluters with specific actions.

- **Constant Sensor Improvements**: As sensors continue to advance, more accurate and reasonably priced devices will be produced, which will facilitate the use of smaller, more portable sensors and improve data accuracy.

- **Development of Policy and Regulation:** Evidence-based policymaking can be aided by air pollution monitoring systems, leading to the creation of more stringent rules and financial incentives for decreasing pollution from a variety of sources.

- **Integration of Remote Sensing and Satellite Data:** By combining satellite images and remote sensing technologies with ground-based data, it is possible to obtain a more thorough understanding of air quality on a regional and global scale, which can aid in the management of pollution beyond national borders. Air Quality Forecasting: By using advanced forecasting models, it is possible to anticipate changes in air quality, which enables the public and government to plan activities and make educated decisions.

- **Initiatives for Climate Change and Emission Reduction:** These systems can help monitor greenhouse gas emissions, support international agreements like the Paris Agreement, and contribute to attempts to mitigate the effects of climate change. Future air pollution monitoring systems have a wide range of potential applications that might greatly enhance environmental preservation, public health, and general quality of life. In order to fulfill the needs of a healthier and more sustainable world, these systems will continue to adapt as technology advances and public awareness of the necessity of clean air grows.

6. Reference's

- https://github.com/swatish17/MQ7-Library (For MQ7 library)
- https://github.com/GeorgK/MQ135 (For MQ135 Library)
- https://github.com/mickey9801/GP2Y1010AU0F (For sensor Library)
- https://github.com/arduino-libraries/LiquidCrystal (For LiquidCrystal Library)

7. Complete Program/Script

```
#define BLYNK_TEMPLATE_ID "TMPL3vv_4dy5R"
#define BLYNK_TEMPLATE_NAME "test"
#define BLYNK_AUTH_TOKEN "y_Kiuc-w0r1ihY7s-TvEyeu9kHtiK6wq"
#define BLYNK_PRINT Serial


#include <LiquidCrystal.h>
#include <WiFi.h>
#include "MQ7.h"
#include "MQ135.h"
#include <GP2Y1010AU0F.h>
#include <BlynkSimpleEsp32.h>

// For LED
LiquidCrystal lcd(5, 18, 19, 21, 22, 23); // RS, E, D4, D5, D6, D7
const int lcdContrast = 13;
```

```
// For Dust Sensor
int measurePin = 35;   // Connect dust sensor analog measure pin to Arduino A0
pin
int ledPin = 4;        // Connect dust sensor LED pin to Arduino pin 2
GP2Y1010AU0F dustSensor(ledPin, measurePin); // Construct dust sensor global
object
float dustDensity = 0;

// For MQ7 Sensor
MQ7 mq7(32, 5.0);
int mq7Value = 0;

// For MQ135 Sensor
MQ135 mq135(33);
int mq135Value = 0;

// For WiFi
const char ssid[] = "Heo";      // Your WiFi SSID
const char password[] = "0123456789"; // Your WiFi password
const char auth[] = BLYNK_AUTH_TOKEN;


//For pizzo
const int buzzerPin = 12; // Change this to the actual pin where your buzzer
is connected


void setup() {
  Serial.begin(9600);
  dustSensor.begin();
  lcd.begin(16, 2);
  pinMode(lcdContrast, OUTPUT);
  analogWrite(lcdContrast, 85);
  playOpening();
  welcome();
  delay(500);
  connectToWiFi();
  Blynk.begin(auth, ssid, password);
}

void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("WiFi not connected");
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(String(ssid) + " Wifi Lost");
    delay(2000); // Adjust the delay to control how often you check the WiFi
status
```

```
    connectToWiFi(); // Attempt to reconnect
  } else {
    // Example: Reading sensors and displaying data
    float dustDensity = dustSensor.read();
    int mq7Value = mq7.getPPM();
    int mq135Value = mq135.getPPM();
    sendBlink(mq7Value,mq135Value,dustDensity);
    // Display sensor data on the LCD
    lcd.clear();
    printLed(0, 0, "Dust Density:");
    printLed(0, 1, String(dustDensity, 2) + " ug/m3");
    delay(2000);
    lcd.clear();
    printLed(0, 0, "MQ7: " + String(mq7Value) + " ppm");
    printLed(0, 1, "MQ135: " + String(mq135Value) + " ppm");

    Serial.println("Dust Density: " + String(dustDensity) + " ug/m3");
    Serial.println("MQ7: " + String(mq7Value) + " ppm");
    Serial.println("MQ135: " + String(mq135Value) + " ppm");


    Blynk.run();


    delay(2000); // Adjust the delay between sensor readings

  }

}

void printLed(int col, int row, String str) {
  lcd.setCursor(col, row);
  if (str.length() <= 16) {
    lcd.print(str); // Print the text as is if it's 16 characters or less
  } else {
    // Scroll the text for strings longer than 16 characters
    int textLength = str.length();
    for (int i = 0; i <= textLength - 16; i++) {
      lcd.clear(); // Clear the LCD screen before printing the next portion
      lcd.setCursor(0, row);
      lcd.print(str.substring(i, i + 16));
      delay(500); // Adjust the delay to control the scrolling speed
    }
  }
}

void welcome() {
  String message = "Air Pollution Monitoring System";
```

```
  int messageLength = message.length();

  for (int i = 0; i <= messageLength - 16; i++) {
    printLed(0, 0, "Welcome to ...");
    printLed(0, 1, message.substring(i, i + 16));
    delay(500);  // Adjust the delay to control the scrolling speed
    lcd.clear();
  }
}

void animateWiFi() {
  const char wifiSymbols[] = { // Define WiFi animation symbols
    ' ', '.', ':', '|', ':', '.', ' '
  };

  static int frame = 0; // Variable to track the frame of the animation
  lcd.setCursor(15, 1); // Position the cursor at the end of the second row
  lcd.print(wifiSymbols[frame]);

  frame = (frame + 1) % (sizeof(wifiSymbols) / sizeof(wifiSymbols[0])); //
Cycle through the animation frames
  delay(200); // Adjust the delay to control the animation speed
  lcd.setCursor(15, 1);
  lcd.print(" "); // Clear the previous symbol
}

void connectToWiFi() {
  WiFi.begin(ssid, password);
  lcd.clear();
  lcd.print("Connecting WiFi");
  lcd.setCursor(0, 1);
  lcd.print("Please wait...");
  Serial.println("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    // delay(100);
    Serial.print(".");
    animateWiFi(); // Call the WiFi animation function while connecting
  }
  delay(1000);
  // Turn off the WiFi animation after a successful connection
  lcd.setCursor(15, 1);
  lcd.print(" "); // Clear the animation symbol

  Serial.println("\nConnected to WiFi");
  Serial.println("");

Serial.println("╫╫╫╫╫╫┌──┐╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫┌┐╫╫╫╫╫┌┐\n╫╫╫╫╫╫│┌┘╫╫╫╫╫╫╫╫╫╫╫
```

```
╪╪╪╤┘╘╪╪╪││\n┌┐┌┐┌┬┬┘╘┐┌─────┬┐┌┐┌───┴┬┴───┘│\n│└┘└┘╪┐┼┤│┌┤┌┐│┌┤
┌┤│├┤┌┤│││├┤┌┐│\n╘┐┌┐┌┤││││││└┤└┘│││││││├┤└┤└┤│├┤└┘│\n┼└┘└┘└┘└┘└┘└──┴───┘
└┴┘└┴──┴───┴──┴───┘\n");
  Serial.println("");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  lcd.clear();
  // String conn = String(ssid) + " Connected";
  printLed(0, 0, String(ssid) + " Connected");
  printLed(0, 1, "Successfully ...");
  delay(4000);
}


void playOpening() {
  // Define the melody notes and durations with their corresponding
frequencies
  int melody[] = {
    659, 659, 0, 659, 0, 523, 659, 0,
    784, 0, 0, 0, 392, 0, 0, 0,
    523, 0, 0, 392, 0, 0, 329, 0,
    0, 440, 0, 493, 0, 440, 392
  };

  int noteDurations[] = {
    8, 8, 4, 8, 4, 4, 8, 4,
    4, 4, 4, 4, 4, 4, 4, 4,
    4, 4, 4, 4, 4, 4, 8, 4,
    4, 8, 4, 8, 4, 8, 4
  };

  // Set the buzzer pin as an output
  pinMode(buzzerPin, OUTPUT);

  // Play the melody
  for (int i = 0; i < sizeof(melody) / sizeof(melody[0]); i++) {
    int noteDuration = 500 / noteDurations[i];
    tone(buzzerPin, melody[i], noteDuration);
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
    noTone(buzzerPin);
  }

  // Wait before playing the melody again
  delay(2000);
}
```

```
void sendBlink(int mq7o,int mq135o,int dustdeno){
  Blynk.virtualWrite(V0, mq7o);
  Blynk.virtualWrite(V1, mq135o);
  Blynk.virtualWrite(V2, dustdeno);
}
```