# Air Quality Monitoring

## PROJECT DEFINITION

The project involves setting up IoT devices to measure air quality parameters and make the data publicly available for raising awareness about air quality and its impact on public health. The objective is to create a platform that provides real-time air quality information to the public. This project includes defining objectives, designing the IoT monitoring system, developing the datasharing platform, and integrating them using IoT technology and Python

## DESIGN THINKING

The project to set up IoT devices for measuring air quality parameters and creating a platform for real-time air quality information dissemination holds significant promise for addressing environmental and public health challenges. By following a design thinking approach, the project aims to create a user- centred and impactful solution

## SOME KEYS

- User – centric approach
- Data Accessibility
- Awareness and Education
- Continuous improvement
- Impact Assessment

## OBJECTIVES

- Promote public awareness
- Access to real time data
- Data transparency
- Data for policy improvement
- Community engagement
- Data driven solutions
- Long term Impact
- Scalability and ReplicabilityTIVES

## ABSTRACT

In this project we are going to make an IoT Based Air Pollution Monitoring System in which we will monitor the Air Quality over a webserver using internet and will trigger a alarm when the air quality goes down beyond a certain level, means when there are sufficient amount of harmful gases are present in the air like CO2, smoke, alcohol, benzene and NH3. It will show the air quality in PPM on the LCD and as well as on webpage so that we can monitor it very easily.

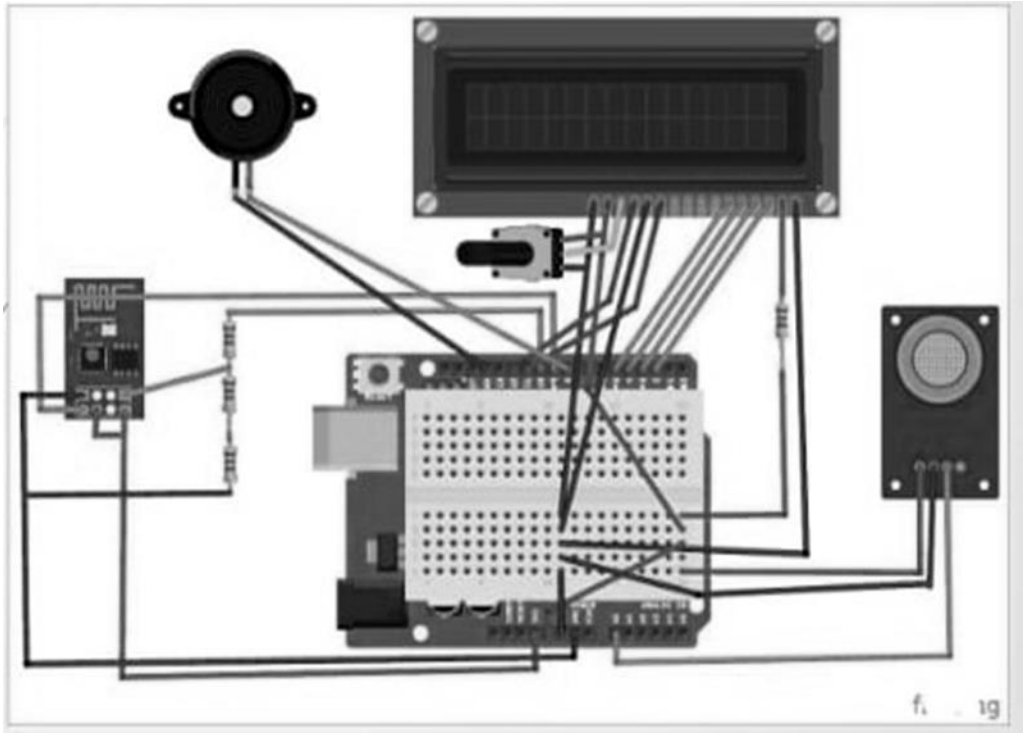
Previously many built the LPG detector using MQ6 sensor, Smoke detector using MQ2 sensor, and Air Quality Analyzer but this time we have used MQ135 sensor as the air quality sensor which is the best choice for monitoring Air Quality as it can detects most harmful gases and can measure their amount accurately. In this IOT project, you can monitor the pollution level from anywhere using your computer or mobile. We can install this system anywhere and can also

trigger some device when pollution goes beyond some level, like we can switch on the Exhaust fan or can send alert SMS/mail to the user.

## REQUIRED COMPONENTS

- MQ135
- Gas sensor
- Arduino Uno
- Wi-Fi moduleESP8266
- 16X2 LCD
- Breadboard
- 10K potentiometer
- 1K ohm resistors
- 220 ohm resistor
- Buzzer

## CIRCUIT DAIGRAM



EXPLANATION

First of all we will connect the ESP8266 with the Arduino. ESP8266 runs on 3.3V and if you will give it 5V from the Arduino then it won't work properly and it may get damage. Connect the VCC and the CH_PD to the 3.3V pin of Arduino. The RX pin of ESP8266 works on 3.3V and it will not communicate with the Arduino when we will connect it directly to the Arduino. So, we will have to make a voltage divider for it which will convert the 5V into 3.3V. This can be done by connecting three resistors in series like we did in the circuit. Connect the TX pin of the ESP8266 to the pin 10 of the Arduino and the RX pin of the esp8266 to the pin 9 of Arduino through the resistors.

ESP8266 Wi-Fi module gives your projects access to Wi-Fi or internet. It is a very cheap device and make your projects very powerful. It can communicate with any microcontroller and it is the most leading devices in the IOT platform. Learn more about using ESP8266 with Arduino here.

Then we will connect the MQ135 sensor with the Arduino. Connect the VCC and the ground pin of the sensor to the 5V and ground of the Arduino and the Analog pin of sensor to the A0 of the Arduino.

Connect a buzzer to the pin 8 of the Arduino which will start to beep when the condition becomes true.

In last, we will connect LCD with the Arduino. The connections of the LCD are as follows Connect pin 1 (VEE) to the ground. Connect pin 2 (VDD or VCC) to the 5V.

Connect pin 3 (V0) to the middle pin of the 10K potentiometer and connect the other two ends of the potentiometer to the VCC and the GND. The potentiometer is used to control the screen contrast of the LCD. Potentiometer of values other than 10K will work too.

Connect pin 4 (RS) to the pin 12 of the Arduino.

Connect pin 5 (Read/Write) to the ground of Arduino. This pin is not often used so we will connect it to the ground.

Connect pin 6 (E) to the pin 11 of the Arduino. The RS and E pin are the control pins which are used to send data and characters.

The following four pins are data pins which are used to communicate with the Arduino.

1.Connect pin 11 (D4) to pin 5 of Arduino.

2.Connect pin 12 (D5) to pin 4 of Arduino.

3.Connect pin 13 (D6) to pin 3 of Arduino.

4.Connect pin 14 (D7) to pin 2 of Arduino.

Connect pin 15 to the VCC through the 220 ohm resistor The resistor will be used to set the back light brightness. Larger values will make the back light much more darker. Connect pin 16 to the Ground.

WORKING PRINCIPLE

The MQ135 sensor can sense NH3, NOx, alcohol, Benzene, smoke, CO2 and some other gases, so it is perfect gas sensor for our Air Quality Monitoring Project. When we will connect it to Arduino then it will sense the gases, and we will get the Pollution level in PPM (parts per million). MQ135 gas sensor gives the output in form of voltage levels and we need to convert it into PPM. So for converting the output in PPM, here we have used a library for MQ135 sensor, it is explained in detail in "Code Explanation" section below.

Sensor was giving us value of 90 when there was no gas near it and the safe level of air quality is 350 PPM and it should not exceed 1000 PPM. When it exceeds the limit of 1000 PPM, then it starts cause Headaches, sleepiness and stagnant, stale, stuffy air and if exceeds beyond 2000 PPM then it can cause increased heart rate and many other diseases.

When the value will be less than 1000 PPM, then the LCD and webpage will display "Fresh Air". Whenever the value will increase 1000 PPM, then the buzzer will start beeping and the LCD and webpage will display "Poor Air, Open Windows". If it will increase 2000 then the buzzer will keep beeping and the LCD and webpage will display "Danger! Move to fresh Air".

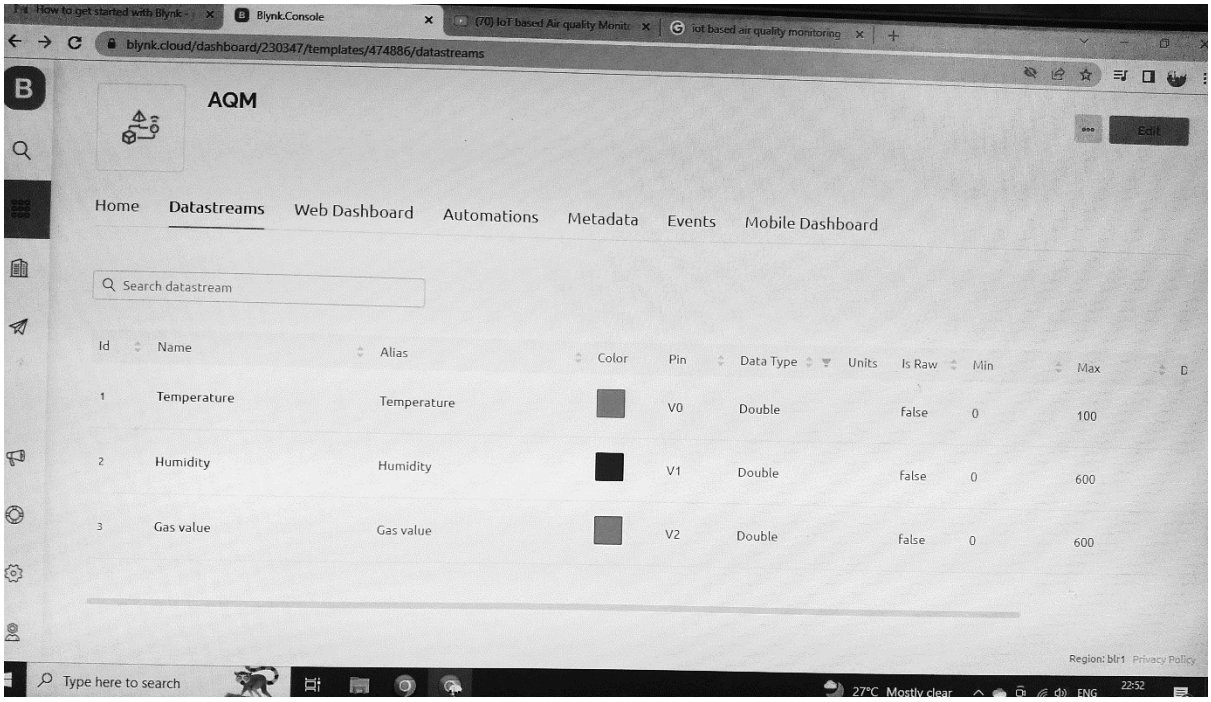
```
#define BLYNK_TEMPLATE_ID "TMPL3IQs5F73H"
#define BLYNK_TEMPLATE_NAME "AQM"
#define BLYNK_AUTH_TOKEN "xKg1ful6LQKU2cl4Xbg6t3rnQ1IE-u4z"
#include "MQ135.h"
#include <SoftwareSerial.h>
#define DEBUG true
SoftwareSerial esp8266(9,10); // This makes pin 9 of Arduino as RX pin and pin 10 of Arduino as the TX pin
const int sensorPin= 0;
int air_quality;
#include <LiquidCrystal.h>
LiquidCrystal lcd(12,11, 5, 4, 3, 2);

void setup() {
pinMode(8, OUTPUT);
lcd.begin(16,2);
lcd.setCursor (0,0);
lcd.print ("circuitdigest ");
lcd.setCursor (0,1);
lcd.print ("Sensor Warming ");
delay(1000);
Serial.begin(115200);
esp8266.begin(115200); // your esp's baud rate might be different
 sendData("AT+RST\r\n",2000,DEBUG); // reset module
 sendData("AT+CWMODE=2\r\n",1000,DEBUG); // configure as access point
 sendData("AT+CIFSR\r\n",1000,DEBUG); // get ip address
 sendData("AT+CIPMUair_quality=1\r\n",1000,DEBUG); // configure for multiple connections
 sendData("AT+CIPSERVER=1,80\r\n",1000,DEBUG); // turn on server on port 80
```

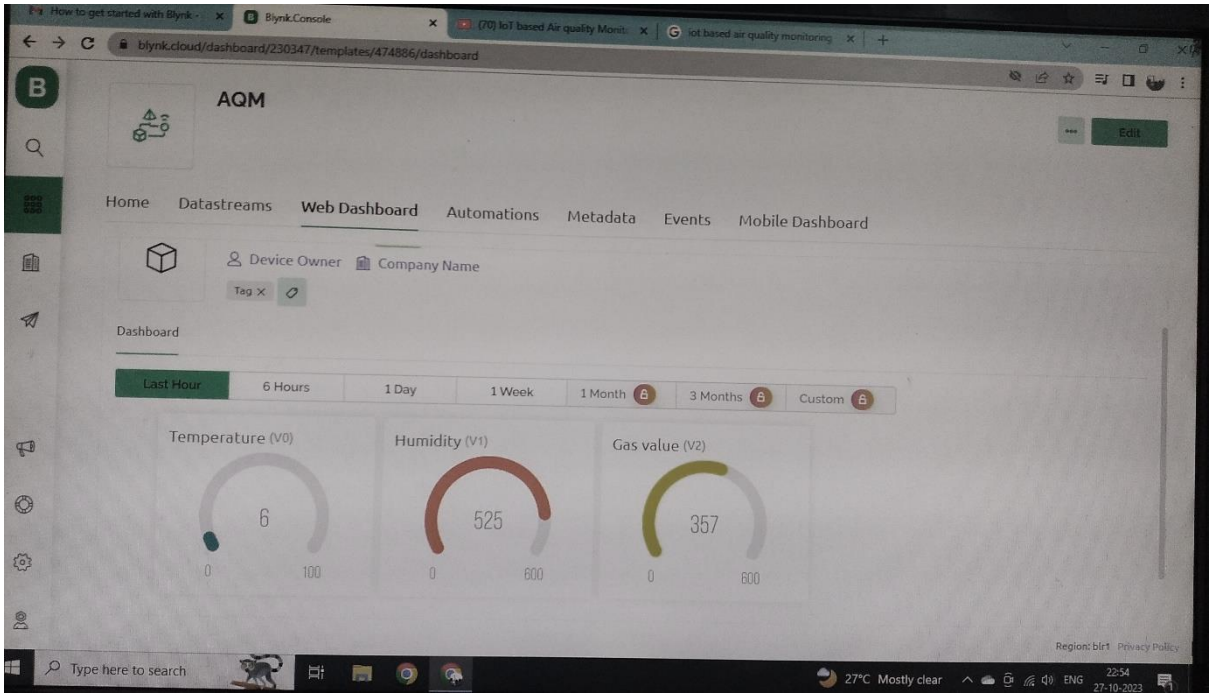
```
pinMode(sensorPin, INPUT);      //Gas sensor will be an input to the arduino
lcd.clear();
}

void loop() {
MQ135 gasSensor = MQ135(A0);
float air_quality = gasSensor.getPPM();
if(esp8266.available()) // check if the esp is sending a message
  {
  if(esp8266.find("+IPD,"))
  {
    delay(1000);
    int connectionId = esp8266.read()-48; /* We are subtracting 48 from the output because the read() function returns the ASCII decimal value and the first decimal number which is 0 starts at 48*/
    String webpage = "<h1>IOT Air Pollution Monitoring System</h1>";
      webpage += "<p><h2>";
       webpage+= " Air Quality is ";
      webpage+= air_quality;
      webpage+=" PPM";
      webpage += "<p>";
    if (air_quality<=1000)
{
  webpage+= "Fresh Air";
}
else if(air_quality<=2000 && air_quality>=1000)
{
  webpage+= "Poor Air";
}
else if (air_quality>=2000 )
```

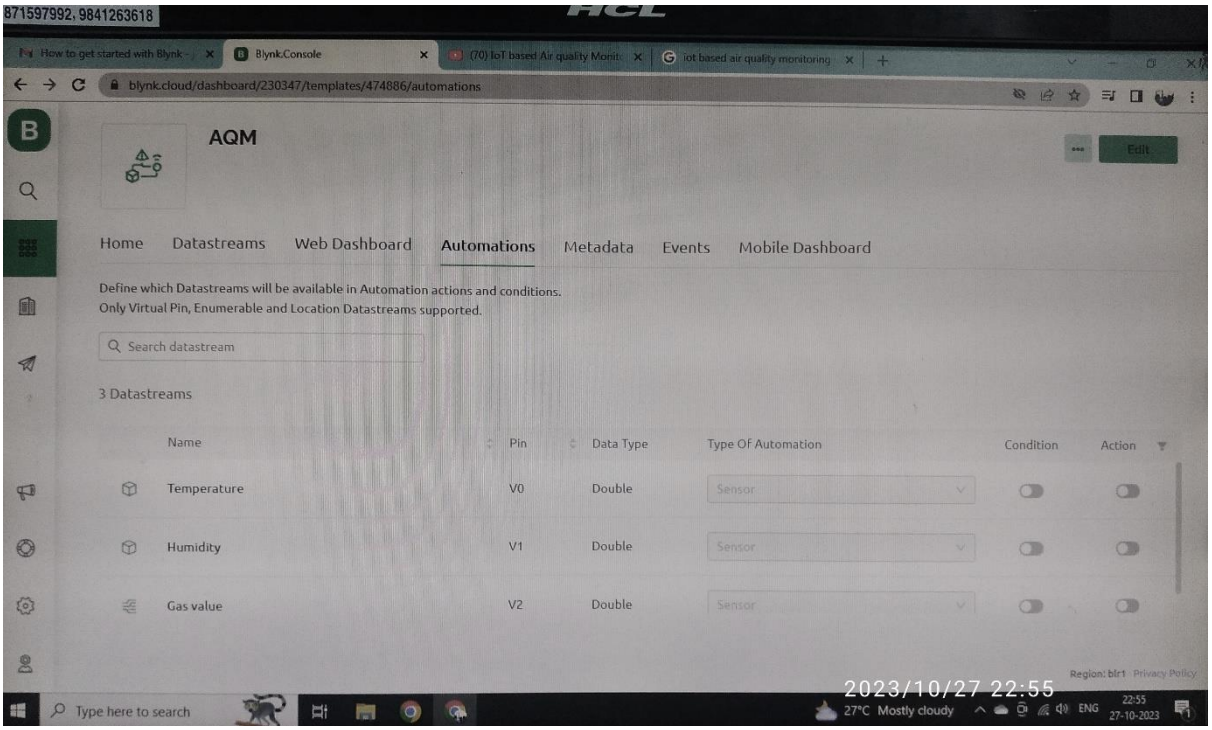
```
{
webpage+= "Danger! Move to Fresh Air";
}
webpage += "</h2></p></body>";
   String cipSend = "AT+CIPSEND=";
    cipSend += connectionId;
    cipSend += ",";
    cipSend +=webpage.length();
    cipSend +="\r\n";
    sendData(cipSend,1000,DEBUG);
    sendData(webpage,1000,DEBUG);
     cipSend = "AT+CIPSEND=";
    cipSend += connectionId;
    cipSend += ",";
    cipSend +=webpage.length();
    cipSend +="\r\n";
   String closeCommand = "AT+CIPCLOSE=";
    closeCommand+=connectionId; // append connection id
    closeCommand+="\r\n";
    sendData(closeCommand,3000,DEBUG);
   }
 }
lcd.setCursor (0, 0);
lcd.print ("Air Quality is ");
lcd.print (air_quality);
lcd.print (" PPM ");
lcd.setCursor (0,1);
if (air_quality<=1000)
{
```

```
lcd.print("Fresh Air");
digitalWrite(8, LOW);
}
else if( air_quality>=1000 && air_quality<=2000 )
{
lcd.print("Poor Air, Open Windows");
digitalWrite(8, HIGH );
}
else if (air_quality>=2000 )
{
lcd.print("Danger! Move to Fresh Air");
digitalWrite(8, HIGH);   // turn the LED on
}
lcd.scrollDisplayLeft();
delay(1000);
}
String sendData(String command, const int timeout, boolean debug)
{
   String response = "";
   esp8266.print(command); // send the read character to the esp8266
   long int time = millis();
   while( (time+timeout) > millis())
  {
    while(esp8266.available())
     {
      // The esp has data so display its output to the serial window
        char c = esp8266.read(); // read the next character.
      response+=c;
    }
```

```
    }
   if(debug)
   {
     Serial.print(response);
   }
   return response;
}
```






## CONCLUSION

Ultimately, the project aspires to make a meaningful contribution to public health and environmental well-being by providing accessible, real-time air quality information. By fostering a sense of responsibility and engagement among the public, it aims to create a healthier and more environmentally conscious community. This project stands as a testament to the potential of technology and design thinking to address pressing societal issues.