BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

Submitted to:
Project Teacher: Deeksha Kumari

Submitted By:

NAME: AKASH DATTA
UID: 17BCS4510

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Chandigarh University, Gharuan

1. INTRODUCTION

1.1 PROJECT DEFINITION

IOT Based Air Pollution Monitoring System is used to monitor the Air Quality over a web server using Internet. It will trigger an alarm when the air quality goes down beyond a certain level, means when there are sufficient amount of harmful gases present in the air like CO2, smoke, alcohol, benzene, NH3 and NOx.

1.2 BENEFITS OF OUR PROJECT

Effective Air Monitoring

In closed spaces, the advanced sensor devices get installed in the desired areas, which work automatically. They instantly detect the presence of particulate matter or air pollutants in the air and trigger notifications on smartphones at once. This enables the authorities to analyze the situation and act accordingly. Instant notifications allow the managers to monitor the air quality and effectively handle the situation. The notifications are sent through SMSs alerts or push messages on their smart devices so that they don't miss out on anything and keep the surroundings fresh enough for people to survive.

•Toxic Gas Detection

The IoT-powered air quality monitoring system is well-equipped with advanced sensor devices and gateway connectivity that is functioned to detect the presence of toxic gases within the premises. The solution is interlinked with the smart device of the user and allows him to take immediate actions in case of disasters. Thus, an IoT-powered system is an essential add-on in the industry to reduce the chances of gas explosions, fire hazards, and other naturally occurring calamities. The industry managers have complete control over the IoT-solution and can take appropriate actions as soon as they receive emergency triggers on their devices, thus reducing the maintenance and operational costs of the industry.

•Temperature and Humidity Measurement

IoT technology is an effective concept that contributes to measuring the temperature and humidity ratio within any industrial premises. It helps the authorities in maintaining a proper ambiance required for the workers to work under certain environmental conditions by keeping real-time control on the IoT-powered solution. The temperature and humidity monitoring helps analyze the situation and maintains a favorable environment as required. Moreover, in the case of mining industries, an IoT-powered air quality monitoring system detects the oxygen levels, thereby keeping the employees safe in case of any decrease in the levels. It allows them to evacuate the premises as soon as there is any decrease in the oxygen levels.

•Human Health

Whether closed or open premises, human health gets affected by the increase in air pollutants or particulate matter. Thus, installing an air quality monitoring system helps monitor the presence of pollutants, resulting in better environmental conditions for humans to reside. This also impacts their health and reduces the chances of occurring any health issues by maintaining a moderate ambiance or as required. Hence, IoT-powered solutions provide better services to the industrialists, which in turn, provides better services to their customers. It creates a positive impact on human health by eliminating unwanted air pollutants and particulate matter by allowing the authorities to take decisions according to the situation.

An IoT-powered **air quality monitoring system** consists of sensor devices and gateway connectivity that offers a data-driven approach to monitor air quality. The solution is fully equipped with necessary devices and advanced techniques to ensure fresher surroundings free from air pollutants and other toxic components spread in the air. It can be easily installed in appropriate places from where it triggers the authorities to take necessary actions whenever required. It is a must-have for the industrial premises to enhance the safety of the employees and keep them healthy under different working conditions.

1.3 FIELD OF PROJECT

They instantly detect the presence of particulate matter or air pollutants in the air and trigger notifications on smartphones at once. This enables the authorities to analyze the situation and act accordingly. Instant notifications allow the managers to monitor the air quality and effectively handle the situation.

2. FEASIBILITY STUDY

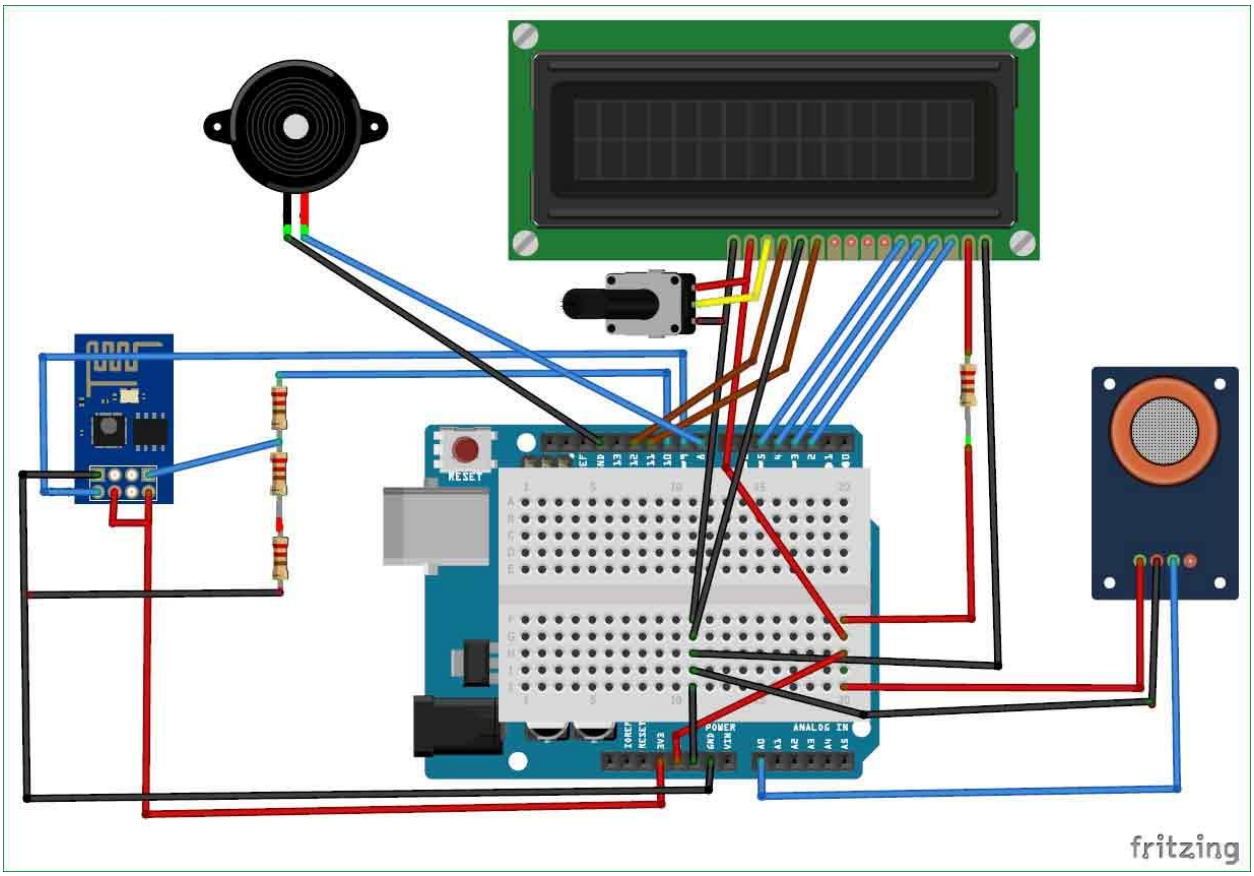
This project proposes an idea to install monitoring applications on smartphones. It is innovative because it provides easy access to the public to monitor real time air quality in their area. It uses low cost and readily available devices such as a dust sensor, carbon monoxide gas sensor, carbon dioxide gas sensor, and nitrogen dioxide gas sensor. For controlling these sensors, microcontrollers are used and the microcontrollers also act as transmitter to transmit the data to the cloud database. The information on air quality can be accessed through a smartphone app in real time.

3. LITERATURE REVIEW

The level of pollution has increased with times by lot of factors like the increase in population, increased vehicle use, industrialization and urbanization which results in harmful effects on human wellbeing by directly affecting health of population exposed to it. In order to monitor 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 web server 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. In this IOT project, you can monitor the pollution level from anywhere using your computer or mobile.

4. METHODOLOGY

We used Thingspeak IoT platform and we clearly defined the derivations that mentions the correct ppm on the screen with correct calibration. We have implemented it with less cost i.e., when we are pushing the data to the cloud, no need to see the output on LCD which adds more cost to the project [1]. When we are targeting IoT as a platform, our intension should be to present the idea on internet using the platforms like thinger.io or thingspeak or Cayenne website which are beautifully designed to present the output and even able to download the dataset. When doing an experiment air quality monitoring, no need to use LPG or methane detecting sensors as it is used for Home/office safety. We have used WiFi to push the data onto the cloud rather using GSM or GPRS module [2]. The problem in another paper that cited at [3] hasn't calibrated the sensor and not even converted the sensor output value into PPM. As per the guidelines by UN Data, 0-50 PPM is SAFE value, 51-100 is moderate as shown in figure 1. Delhi is the most polluted city in the world recorded around 250PPM. As we are using two sensors, both of them have internal heat element, it draws more power(P= V*I), so though the both sensors are turned ON, its output voltage levels varies and shows unpredictable values due to insufficient power drive. So we used a 9V battery and a 7805 family LM7805 Regulator for the CO sensor MQ7. We have used Arduino Uno Development kit that comes with ATMega328P microcontroller. In order to provide WiFi Support for it, we have used cost effective ESP-01 WiFi module which helps us to connect to the ThingSpeak Platform. The connections between them is mentioned in the connections diagram.



5. MODULE & TEAM MEMBER WISE DISTRIBUTION OF WORK

```
MQ135 gasSensor = MQ135(A0);
float air_quality = gasSensor.getPPM();
```

But before that we need to **calibrate the MQ135 sensor**, for calibrating the sensor upload the below given code and let it run for 12 to 24 hours and then get the RZERO value.

```
#include "MQ135.h"
void setup (){
Serial.begin (9600);
}
void loop() {
MQ135 gasSensor = MQ135(A0); // Attach sensor to pin A0
float rzero = gasSensor.getRZero();
Serial.println (rzero);
delay(1000);
}
```

After getting the RZERO value. **Put the RZERO value in the library file** you downloaded "MQ135.h": #define RZERO 494.63

Now we can begin the actual code for our Air quality monitoring project.

In the code, first of all we have defined the libraries and the variables for the Gas sensor and the LCD. By using the Software Serial Library, we can make any digital pin as TX

and RX pin. In this code, we have made Pin 9 as the RX pin and the pin 10 as the TX pin for the ESP8266. Then we have included the library for the LCD and have defined the pins for the same. We have also defined two more variables: one for the sensor analog pin and other for storing air_quality value.

```
#include <SoftwareSerial.h>

#define DEBUG true

SoftwareSerial esp8266(9,10);

#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11, 5, 4, 3, 2);

const int sensorPin= 0;

int air_quality;
```

Then we will declare the pin 8 as the output pin where we have connected the buzzer. lcd.begin(16,2) command will start the LCD to receive data and then we will set the cursor to first line and will print the ‘circuitdigest’. Then we will set the cursor on the second line and will print ‘Sensor Warming’.

```
pinMode(8, OUTPUT);

lcd.begin(16,2);

lcd.setCursor (0,0);

lcd.print ("circuitdigest ");

lcd.setCursor (0,1);

lcd.print ("Sensor Warming ");

delay(1000);
```

Then we will set the baud rate for the serial communication. Different ESP’s have different baud rates so write it according to your ESP’s baud rate. Then we will send the commands to set the ESP to communicate with the Arduino and show the IP address on the serial monitor.

```
Serial.begin(115200);
```

```
esp8266.begin(115200);
    sendData("AT+RST\r\n",2000,DEBUG);
    sendData("AT+CWMODE=2\r\n",1000,DEBUG);
    sendData("AT+CIFSR\r\n",1000,DEBUG);
    sendData("AT+CIPMUair_quality=1\r\n",1000,DEBUG);
    sendData("AT+CIPSERVER=1,80\r\n",1000,DEBUG);
pinMode(sensorPin, INPUT);
lcd.clear();
```

For printing the output on the webpage in web browser, we will have to use **HTML programming**. So, we have created a string named webpage and stored the output in it. 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.

```
if(esp8266.available())
   {
      if(esp8266.find("+IPD,"))
      {
        delay(1000);
        int connectionId = esp8266.read()-48;
        String webpage = "<h1>IOT Air Pollution Monitoring System</h1>";
            webpage += "<p><h2>";
            webpage+= " Air Quality is ";
            webpage+= air_quality;
            webpage+=" PPM";
            webpage += "<p>";
```

The following code will call a function named sendData and will send the data & message strings to the webpage to show.

```
        sendData(cipSend,1000,DEBUG);
        sendData(webpage,1000,DEBUG);


        cipSend = "AT+CIPSEND=";
        cipSend += connectionId;
        cipSend += ",";
        cipSend +=webpage.length();
        cipSend +="\r\n";
```

The following code will print the data on the LCD. We have applied various conditions for checking air quality, and LCD will print the messages according to conditions and buzzer will also beep if the pollution goes beyond 1000 PPM.

```
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);
```

Finally the below function will send and show the data on the webpage. The data we stored in string named ‘webpage’ will be saved in string named ‘command’. The ESP will then read the character one by one from the ‘command’ and will print it on the webpage.

```
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;
}
```

7. REQUIREMENTS

PMS5003 PM2. 5 Particulate Matter Sensor.

- MQ-135 Air Quality Sensor.
- BME280 Barometric Pressure Sensor.
- IoT Based Air Pollution/Quality Monitoring with ESP8266.
- Setting up Thingspeak.
- Source Code/Program IoT based Air Quality/Pollution Monitoring.