Vehicular Pollution Monitoring System

A Mini Project Report

Submitted to the APJ Abdul Kalam Technological University

in partial fulfillment of requirements for the award of degree

Bachelor of Technology

in

Electronics and Communication Engineering

by

Adith T M(NSS20EC004)

M Arathi Krishnan(NSS20EC054)

Saranya Suresh(NSS20EC076)

Shazia Shanavas(NSS20EC077)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

NSS College of Engineering, Palakkad

KERALA

AUGUST 2023

DEPT. OF ELECTRONICS & COMMUNICATION ENGINEERING

NSS College of Engineering

Palakkad 2022-23

CERTIFICATE

This is to certify that the report entitled **VEHICULAR POLLUTION MONITORING SYSTEM** submitted by **Adith T M** (NSS20EC004), **M Arathi Krishnan** (NSS20EC054), **Saranya Suresh** (NSS20EC076) & **Shazia Shanavas** (NSS20EC077) to the APJ Abdul Kalam Technological University in partial fulfillment of the B.Tech. degree in Electronics and Communication Engineering is a bonafide record of the project work carried out by them under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Sruthi N
Assistant Professor
Dept. of ECE
NSS College of Engineering
Palakkad

Prof. Vinod G
Associate Professor and Head
Dept. of ECE
NSS College of Engineering
Palakkad

DEPT. OF ELECTRONICS & COMMUNICATION ENGINEERING
NSS COLLEGE OF ENGINEERING
PALAKKAD 2022-23

DECLARATION

We hereby declare that the project report **VEHICULAR POLLUTION MONITORING SYSTEM**, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Sruthi N, Assistant Professor

This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources.

We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Adith T M
M Arathi Krishnan
Saranya Suresh
Shazia Shanavas

Palakkad
03-08-2023

Acknowledgement

We take this opportunity to express our deepest sense of gratitude and sincere thanks to everyone who helped us to complete this work successfully. We express our sincere thanks to Prof. Vinod G, Head of Department, Electronics and Communication Engineering, NSS College of Engineering for providing us with all the necessary facilities and support.

We would like to place on record our sincere gratitude to our project guide Sruthi N, Assistant Professor, Department of Electronics and Communication Engineering, NSS College of Engineering for the guidance and mentor ship throughout this work.

Finally, we thank our families, and friends who contributed to the successful fulfillment of this project work.

Adith T M
M Arathi Krishnan
Saranya Suresh
Shazia Shanavas

Abstract

The escalating levels of Vehicular Pollution have emerged as a significant and pressing concern for both public health and environmental well-being. To tackle this issue, we propose a groundbreaking approach to monitor Vehicular Pollution utilizing a WiFi module. The primary objective of our system is to offer real-time monitoring and analysis of pollutant levels emitted by vehicles in urban areas. To achieve this, our system is designed and implemented using the Arduino Uno platform, incorporating a WiFi Module and a Gas Sensor. The Gas Sensor is capable of accurately measuring the concentration of critical pollutants, such as carbon monoxide (CO), nitrogen dioxide (NO2), and particulate matter (PM) emitted from vehicles.

The integration of the WiFi module enables seamless communication with an online server or cloud platform, facilitating real-time data transmission and analysis. As a result, vehicle owners can now be promptly alerted via email if their vehicle's pollution level exceeds a predetermined threshold value. By providing vehicle owners with timely and actionable information, our system empowers them to take proactive measures in reducing their vehicle's emissions and contributing to overall pollution reduction in urban environments. Moreover, the availability of real-time pollution data offers urban planners and policymakers valuable insights to develop targeted strategies and policies aimed at curbing Vehicular Pollution effectively.

This novel approach to Vehicular Pollution monitoring holds immense potential to revolutionize urban air quality management and enhance environmental sustainability. As a cost-effective and user-friendly solution, it has the capability to be widely adopted and make a significant impact in addressing the critical issue of Vehicular Pollution in urban areas.

Contents

Chapter 1

Introduction

One of the most important ecological problems that directly affects both human health and ecological balance is air pollution. It is a significant factor in a great number of sudden deaths. In addition to harming human health, air pollution also has an adverse impact on the ecosystem, causing smog, acid rain, ozone layer damage, and global warming. In addition to its negative impacts on health, air pollution also causes significant economic losses, particularly in terms of the money needed to provide medical care to those who are harmed. Due to their lack of proper protection from air pollution, the poor are frequently the population group that is most impacted.

Vehicular Pollution has a negative impact on urban air quality, as a result of pollutants entering the atmosphere from vehicles. We must implement effective and dynamic pollution control measures in order to improve the environment and air quality. Thus, it is vital to screen and control the air contamination. The best way to deal with controlling air contamination is to monitor exceeded degrees of air pollutants and taking appropriate measures to control it.Even though there are systems available, it requires the user to go to a test centre for pollution testing making it hectic for people.

Chapter 2

Block Diagram and Description

2.1 Block Diagram

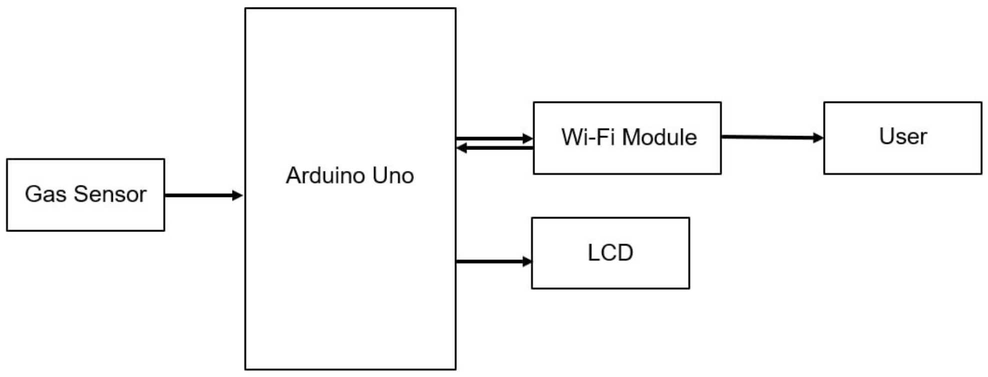


Figure 2.1: Block Diagram

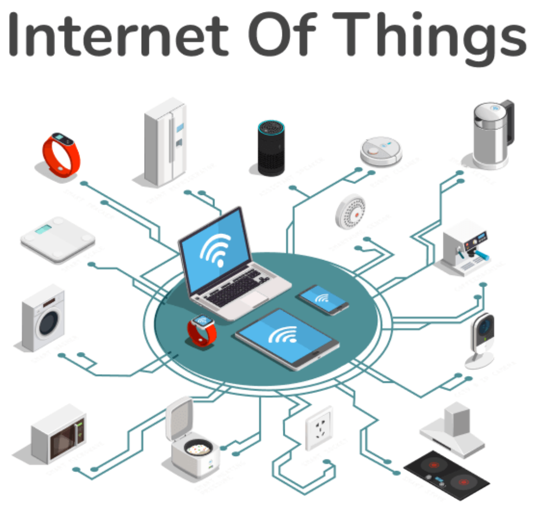
2.2 Block Diagram Description

The block diagram for Vehicular Pollution monitoring system consists of Arduino Uno which acts as the main control unit for data processing and decision making. A gas sensor MQ135 is connected to Arduino Uno to detect the pollution causing gases. The PPM is detected using the sensor interfaced with Arduino and displayed in a 16×2 LCD which is interfaced with the Arduino. A WiFi module is connected to the Arduino Uno which sends an Email alert to the user indicating the pollution levels.

Chapter 3

Technologies Used

3.1 Internet of Things (IoT)



The integration of sensors, Arduino, and WiFi modules in the Internet of Things (IoT) has led to a revolutionary paradigm of smart and connected devices. Sensors gather real-world data, Arduino acts as the intelligent processing unit, and the WiFi module enables seamless communication with the internet. This powerful combination empowers IoT devices to collect, analyze, and exchange information in real-time, unlocking a wide range of applications across industries. From smart homes and industrial automation to environmental monitoring and healthcare, this trifecta of technology promises to create a more efficient, convenient, and sustainable future.

Chapter 4

Components Used And Description

4.1 Arduino Uno

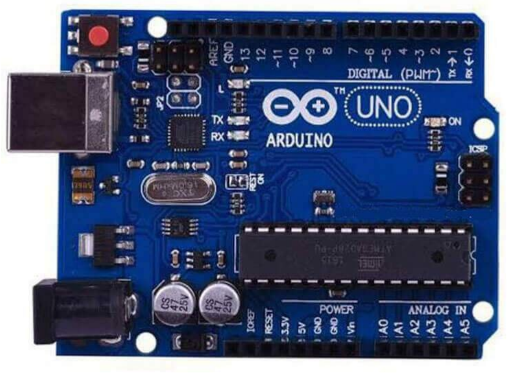


Figure 4.1: Arduino Uno

The Arduino Uno is a popular and versatile micro controller board that is widely used in various electronics and prototyping projects. Developed by the Arduino team, the Uno is based on the ATMega328P micro controller, which is the brain of the board. It offers a simple and user-friendly platform for beginners and advanced users alike to develop interactive and creative projects. The Arduino Uno board features a set of digital input/output pins, analog input pins, PWM (Pulse Width Modulation) pins, and various communication interfaces such as UART, I2C, and SPI. These pins and interfaces allow users to connect various sensors, actuators, displays, and other components, making it an excellent choice for prototyping and experimentation. Programming the Arduino Uno is done using the Arduino Integrated Development Environment (IDE), which is based on the C/C++ language. The IDE provides a straightforward and intuitive interface for writing, uploading, and debugging code on the board.

4.1.1 Power

USB or an external power supply is used for powering the Arduino. Power pins are as follows:

- **Pin 3.3V & 5V**
  The micro controller and other components used on the Arduino mega board are powered by this regulated power source. It can be acquired from the board's Vin pin or another regulated voltage source, such as a USB cable, or another regulated voltage source, such as 3.3V0-pin. This may draw a maximum of 50mA in power.

- **GND Pin**
  The 5-GND pins on the Arduino mega board are available for usage anytime the programme calls for it.

4.1.2 Reset (RST) Pin

This board's RST pin may be used to rearrange the board. This pin can be turned low to reorganise the board.

4.1.3 Vin Pin

The board may be supplied with the input voltage in the 7 to 20 volt range. The voltage provided by the power jack can be accessed using this pin. However, the board will by default output 5V through this pin.

4.1.4 Serial Communication

To send and receive serial data, this board's serial pins TXD and RXD are used. Information is transmitted when a Tx is used, and data is received when an RX is used. There are four configurations for the serial pins on this board. It comprises Tx (1) and Rx (0) for serial 0, Tx (18) and Rx (19) for serial 1, Tx (16) and Rx (17)for serial 2 and Tx (14), Rx(15) for serial 3.

4.1.5 LED

The LED on this Arduino board is connected to pin-13, sometimes known as digital pin 13. This LED may be regulated based on the high and low values of the pin. You will be able to alter your programming abilities in real time as a result.

4.1.6 AREF

Analog Reference Voltage, often known as AREF, is a reference voltage for analogue inputs.

4.1.7 Analog Pins

The board contains 16 analog pins with the designations A0-A15. The fact that all of the analog pins on this board can be used as digital I/O pins is crucial information. Each analog pin can be accessed with a 10-bit resolution that measures voltages between GND and 5 volts. However, the AREF pin and the analog Reference function both allow for the higher value to be changed ().

4.1.8 I2C

The Serial Data Line (SDA), which is used to store data, and the Serial Clock Line (SCL), which is primarily used to provide data synchronisation among the devices, are the two pins that may allow I2C communication.

4.1.9 SPI Communication

The data transmission between the controller and other components is done via a serial peripheral interface, or SPI. SPI uses four pins for communication: MISO (50), MOSI (51), SCK (52), and SS (53).

4.1.10 Dimensions

The dimensions of the Arduino Uno board are approximately 68.6mm (length) x 53.4mm (width). These dimensions are for the standard Arduino Uno board, and there might be slight variations in size for different versions or clones of the board.

4.1.11 Programming

An Arduino Uno can be programmed using an IDE, which supports the C programming language. In this instance, the code is written within the application and transmitted via a USB connection to the Arduino board to produce the sketch.

An Arduino Uno board's boot loader eliminates the requirement to burn the program code onto the Arduino board using an external burner. Here, the boot loader's communication can be done using the STK500 protocol. When using the Arduino board for your project, the power supply can be supplied via a power jack or the board's Vin pin.

4.2 ESP8266 Wi-Fi Module

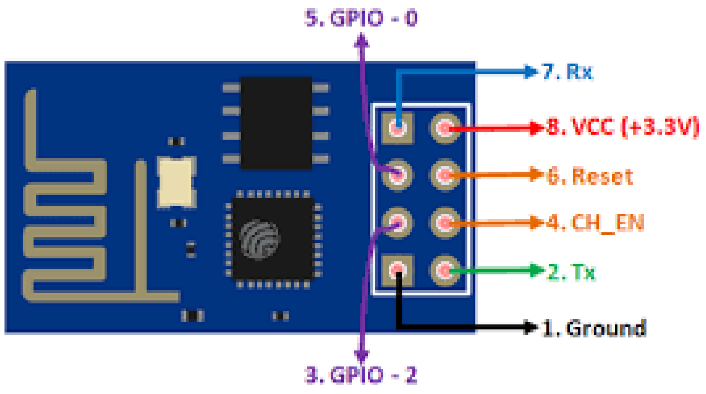


Figure 4.2: ESP8266 Wi-Fi Module

The ESP8266 is a popular Wi-Fi module that has gained widespread use in the Internet of Things (IoT) and home automation projects due to its low cost, ease of use, and extensive community support. The most common version of the ESP8266 module has 8 pins, which are as follows:
1. VCC: Connect this pin to a 3.3V power supply to power the module.
2. GND: Connect this pin to the ground of the power supply.
3. TXD: This is the Transmit Data pin and is used for sending data from the ESP8266 to another device (e.g., a micro controller or USB-to-serial converter).
4. RXD: This is the Receive Data pin and is used for receiving data from another device.
5. CH_PD: This is the chip power-down pin. Connect it to VCC to enable the module.
6. RST: This is the Reset pin. Connect it to GND to reset the module.
7. GPIO0: This is a General Purpose Input/Output pin. Its state during bootup determines whether the module enters programming mode or normal operation mode.
8. GPIO2:This is another General Purpose Input/Output pin. It can be used for various purposes based on your project requirements.

The ESP8266 module can be programmed using the Arduino IDE or other development environments, making it accessible to a wide range of users, from beginners to experienced developers. It provides a cost-effective and straightforward solution for adding Wi-Fi capabilities to electronic projects, enabling them to communicate with other devices and the internet.

4.3 MQ135 Gas Sensor

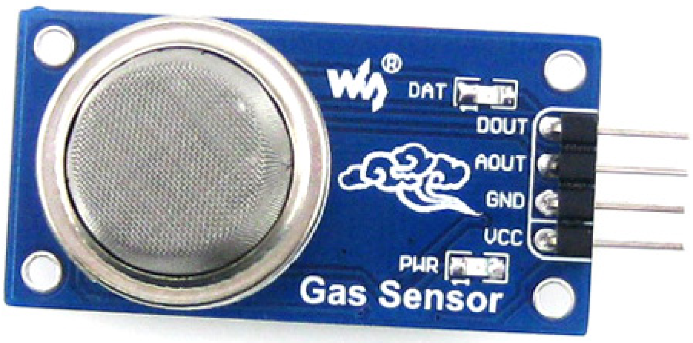


Figure 4.3: MQ135 Gas Sensor

The MQ135 gas sensor is a semiconductor-based sensor commonly used to detect various gases in the air. It works on the principle of gas absorption, which changes the sensor's resistance when exposed to specific gases. The sensor's resistance is measured, and based on the change, gas concentration can be estimated. The MQ135 sensor is sensitive to a range of gases, including ammonia (NH3), nitrogen oxides (NOx), benzene (C6H6), alcohol, carbon monoxide (CO), and smoke. It is widely used in air quality monitoring devices, gas leak detection systems, air purifiers, and environmental monitoring projects. To ensure accurate measurements, the MQ135 sensor may require calibration. Calibration involves exposing the sensor to known gas concentrations and creating a relationship between gas concentration and the sensor's resistance. This process improves the accuracy of the sensor's readings.When using the MQ135 sensor, it's essential to consider its limitations. High humidity and certain contaminants can affect its sensitivity, so it should be used and placed carefully in the target environment. The sensor typically operates at 5V DC, making it compatible with micro controllers like Arduino and Raspberry Pi.

4.4 Jumper Wires

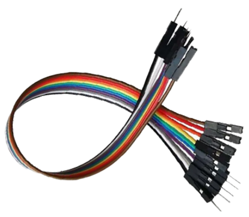


Figure 4.4: Jumper Wires

Electrical lines having connection pins at either end are known as jumper wires. They are employed to connect two circuit locations without the usage of solder. There are three types of jumper wires :

- Male-to-male jumper
- Male-to-female jumper
- Female-to-female jumper

While female ends do not have a projecting pin yet may also plug into objects, male ends do.

4.5 16x2 LCD

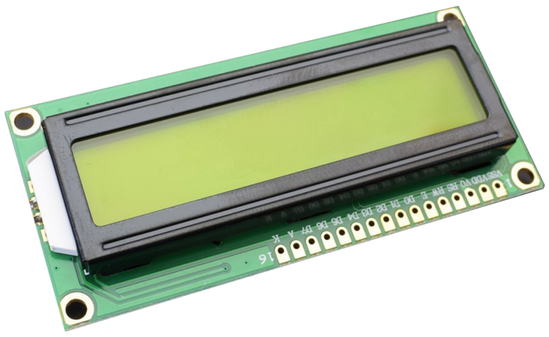


Figure 4.5: 16x2 LCD

A 16x2 LCD (Liquid Crystal Display) is a small electronic screen commonly used in various devices, such as digital clocks, calculators, and small embedded systems. It consists of 16 characters in each of its two lines, allowing it to display alphanumeric characters, symbols, and basic graphics. The LCD operates on low power and provides a simple interface for displaying essential information, making it a popular choice in many electronic projects and devices.

Chapter 5

System Design and Working

5.1 Detection of Pollution Causing Gases using MQ135 Gas Sensor

The MQ135 sensor can detect NH3, NOx, alcohol, Benzene, smoke, CO2 and a few other gases, so it is used for our Air Quality Monitoring Project. At the point when we connect it to Arduino then it will detect the gases, and we will get the Pollution level in PPM (parts per million). The MQ135 gas sensor outputs voltage levels, which must be converted into PPM. So for converting the result in PPM, here we have used a library for MQ135 sensor. Sensor was giving us value of 90 when there was no gas close to it and the safe degree of air quality is 350 PPM and it shouldn't surpass 1000 PPM. When it goes above the limit of 1000 PPM, it starts to cause headaches, sleepiness, and air that is stagnant, stale, and stuffy. If it goes above 2000 PPM, it can increased heart rate and many other diseases.

5.2 Displaying PPM on LCD

At the point when the value is less than 1000 PPM, the LCD and page will show "Fresh Air". The LCD and web page will both display the message "Poor Air, Open Windows" whenever the value rises by 1000 PPM. If it goes above 2000, the buzzer will continue to beep, and the LCD and web page will show "Danger! Move to fresh air".

5.3 Sending messages using Wi-Fi Module

To send a message using ESP8266 and Arduino, connect the ESP8266 module to the Arduino, initialize Wi-Fi, establish communication (HTTP, MQTT, etc.), craft and send the message, verify using the serial monitor, upload the code, and ensure successful message transmission. Consider security measures if needed.

Chapter 6

Result and Conclusion

6.1 Result

The proposed air pollution detection system helps users to keep vehicle in good condition. It alerts the user about the detected pollution level through Email. This is achieved by integrating a Wi-fi module to the circuit.

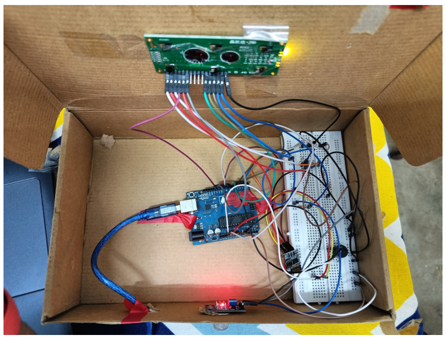
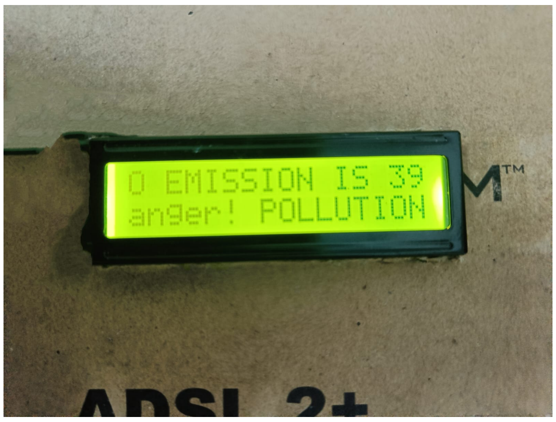


Figure 6.1: Prototype

6.1.1 Output



```
O EMISSION IS 39
anger! POLLUTION
```

Figure 6.2: Output

6.1.2 Simulation

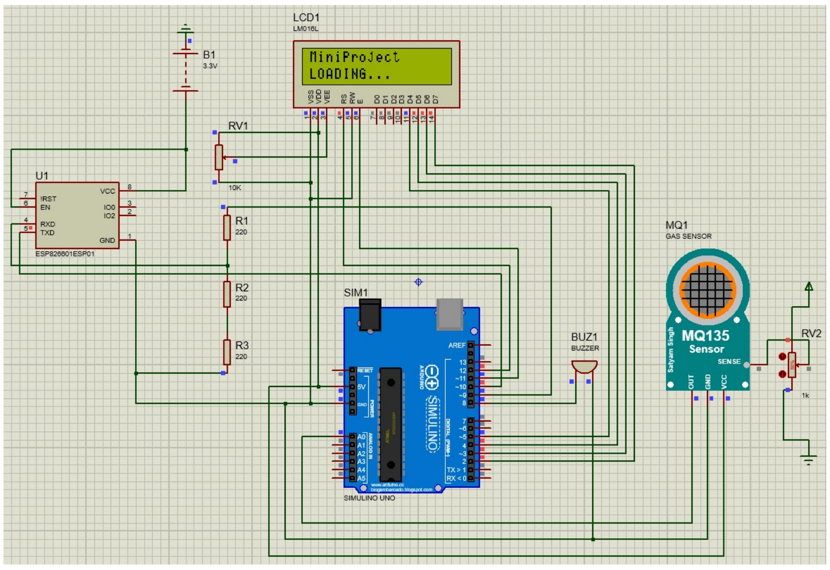


Figure 6.3: Output

6.2 Conclusion

In conclusion, the creation and use of a system for monitoring automotive pollution that combines an Arduino Uno, a WiFi module, and a gas sensor has shown to be an essential step in measuring and reducing air pollution brought on by vehicular emissions. The system's real-time monitoring capabilities enable quick detection of pollution occurrences and deliver essential information for analysis and decision-making. The capability of remote accessibility guarantees openness and public awareness while providing a practical means of keeping track of air quality. This system promotes environmentally friendly transportation habits and protects both the public's health and the environment by helping to advance environmental protection efforts.

Chapter 7

Merits, Limitations and Future Prospects

7.1 Merits

- Our project contains a notification mechanism that notifies the user the pollution levels of the vehicle, hence making it easier for the user to know the pollution levels of their vehicle.
- The user need not visit the Pollution Control Centre every now and then.
- We have devised an inexpensive project which is also compact.
- Our system no longer requires the use of a GSM module and SIM to deliver messages because we used online API services.
- The system is easy to operate and does not require any prior technical knowledge.

7.2 Limitations

- The system as of now does not have a feature of alerting the concerned authorities about the pollution causing vehicle.
- Moreover the vehicle's location could only be shared with the authorities by integrating a GPS module which haven't been installed yet in our project.
- Our project just aims in pointing out the problem and does not solve pollution issues.

7.3 Future Prospects

We plan on developing our project into a fully functional and compact product after integrating a GPS module to track the vehicle and also a way to alert the concerned authorities of the pollution causing vehicle.

References

[1] ’Monitoring Vehicular Pollution by using Embedded System’,International Journal of Trend in Research and Development, Volume 5(2), ISSN: 2394-9333

[2] ’Vehicular Pollution Monitoring System using IoT’, International Journal of Recent Technology and Engineering ISSN: 2277-3878(Online), Volume-9 Issue-1, May 2020

[3] ’The 8051 Microcontroller and Embedded Systems Using Assembly and C Second Edition’, by Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. McKinlay

[4] ’Gross Polluters and Vehicle Emission Reduction’ article on nature sustainability by Matteo Bohm, Micro Nanni and Luca Pappalardo (https://doi.org/10.1038/s41893-022-00903-x)

[5] ’Temporal and Spatial Impact of Lockdown during COVID-19 on Air Quality Index in Haryana, India’, article by Manjeet, Anurag Airon, Rahul Kumar and Ruksar Saifi published in www.nature.com/scientificreports

[6] www.proteus.com

[7] ’Development of IoT based Vehicular Pollution Monitoring System’, 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), IEEE-Noida, September 2015

Appendix A

Program Code

```
#include "MQ135.h"
#include <SoftwareSerial.h>
#include <LiquidCrystal.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
SoftwareSerial Serial1(9,10); // RX, TX
boolean No_IP=false;
String IP="";
const int sensorPin= 0;
int air_quality;
LiquidCrystal lcd(12,11, 5, 4, 3, 2);
void setup() {
pinMode(8, OUTPUT);
lcd.begin(16,2);
lcd.setCursor (0,0);
lcd.print ("MiniProject");
lcd.setCursor (0,1);
lcd.print ("LOADING...");
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<=000)
{
  webpage+= "Fresh Air"}
else
{
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 ("CO EMISSION IS ");
lcd.print (air_quality);
lcd.print (" PPM ");
lcd.setCursor (0,1);
if (air_quality<=1000)
{lcd.print("NO POLLUTION");
digitalWrite(8, LOW);
lcd.scrollDisplayLeft();}
else
{lcd.print("Danger! POLLUTION IS HIGH");
digitalWrite(8, HIGH);   // turn the LED on
```

```
lcd.scrollDisplayLeft();
delay(2000);}
delay(1000);
if(air_quality>1000)
{ Serial.begin(9600);
     lcd.begin(16,2);
     lcd.print("Sending Email by");
     lcd.setCursor(0,1);
     lcd.print(" Arduino & WIFI ");
     delay(2000);
     lcd.clear();
     lcd.print("  Search wifi  ");
     delay(2000);
     lcd.clear();
     lcd.print("Finding ESP8266");
     connect_wifi("AT",100);
     connect_wifi("ATE1",100);
     lcd.clear();
     lcd.print("Connected");
     delay(1000);
     connect_wifi("AT+CWMODE=3",100);
     connect_wifi("AT+CWQAP",100);
     connect_wifi("AT+RST",5000);
     lcd.clear();
     lcd.print("Connecting WiFi");
     check4IP(5000);
     if(!No_IP)
     {
        Serial.println("Connecting Wifi....");
        connect_wifi("AT+CWJAP=\"aswin\",\"123456787\"",7000);}
        //provide your WiFi username and password here
      else
        {}
      lcd.clear();
      lcd.print("WIFI Connected...");
      Serial.println("Wifi Connected");
      delay(1000);
      lcd.clear();
      lcd.print("Getting IP Add.");
      Serial.println("Getting IP Address....");
      get_ip();
      delay(1000);
      lcd.clear();
      lcd.print("IP:");
      lcd.print(IP);
      lcd.setCursor(0,1);
      lcd.print("PORT: 80");
```

```
    connect_wifi("AT+CIPMUX=1",100);
    connect_wifi("AT+CIPSERVER=1,80",100);
    delay(2000);
    lcd.clear();
    lcd.print("Configuring Email..");
Serial1.println("AT+CIPSTART=4,\"TCP\",\"mail.smtp2go.com\",2525");
delay(2000);
Serial1.println("AT+CIPSEND=4,20");
delay(2000);
Serial1.println("EHLO 192.168.1.123");
delay(2000);
Serial1.println("AT+CIPSEND=4,12");
delay(2000);
lcd.clear();
lcd.print("Try To Login.....");
Serial1.println("AUTH LOGIN");
delay(2000);
Serial1.println("AT+CIPSEND=4,30");
delay(2000);
Serial1.println("c2FkZGFtNDIwMUBnbWFpbC5jb20=");
//base64 encoded username
delay(2000);
Serial1.println("AT+CIPSEND=4,18");
delay(2000);
Serial1.println("Y2lyY3VpdDQyMDE=");
//base64 encoded password
lcd.clear();
lcd.print("Login Success");
delay(2000);
Serial1.println("AT+CIPSEND=4,34");
delay(2000);
Serial1.println("MAIL FROM:<adith7923@gmail.com>");
// use your email address
delay(2000);
Serial1.println("AT+CIPSEND=4,32");
delay(2000);
lcd.clear();
lcd.print("Seniding Email 2");
lcd.setCursor(0,1);
lcd.print("Saddam4201@ gmail");
Serial1.println("RCPT To:<saddam4201@ gmail.com>");
delay(2000);
Serial1.println("AT+CIPSEND=4,6");
delay(2000);
Serial1.println("DATA");
delay(2000);
Serial1.println("AT+CIPSEND=4,24");
```

```
  delay(2000);
  Serial1.println("Testing Success");
  delay(2000);
  Serial1.println("AT+CIPSEND=4,3");
  delay(2000);
  Serial1.println('.');
  delay(10000);
  Serial1.println("AT+CIPSEND=4,6");
  delay(2000);
  Serial1.println("QUIT");
  delay(2000);
  lcd.clear();
  lcd.print("Email Sent...");}}
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;}
void check4IP(int t1)
{ int t2=millis();
  Serial1.flush();
  while(t2+t1>millis())
  {while(Serial1.available()>0)
    {if(Serial1.find("WIFI GOT IP"))
      {No_IP=true;}}}}
void get_ip()
{IP="";
  char ch=0;
  while(1)
  { Serial1.println("AT+CIFSR");
    while(Serial1.available()>0)
    { if(Serial1.find("STAIP,"))
      { delay(1000);
        Serial.print("IP Address:");
        while(Serial1.available()>0)
        { ch=Serial1.read();
          if(ch=='+')
          break;
          IP+=ch;}}
      if(ch=='+')
      break;}
    if(ch=='+')
    break;
    delay(1000);}
  Serial.print(IP);
  Serial.print("Port:");
  Serial.println(80);}
void connect_wifi(String cmd, int t)
{ int temp=0,i=0;
  while(1)
  { Serial1.println(cmd);
    while(Serial1.available())
    { if(Serial1.find("OK"))
      i=8;}
    delay(t);
    if(i>5)
    break;
    i++;}
  if(i==8)
  Serial.println("OK");
  else
  Serial.println("Error");}
```