ABSTRACT

The ultimate goal of this project was on developing an Internet of Things (IoT) based air pollution monitoring system. This document gives stage by stage procedures that the developer undertook in coming up with the system. The developer undertook an assessment and the results showed that air pollution has alarming effects. Its effects cause a lot of human diseases which include respiratory and heart conditions amongst other body threats. The objectives of this project are to monitor air quality as the detects harmful gases like, carbon monoxide, Sulphur dioxide and nitrogen oxide, to trigger devices such as buzzer and a fan when the pollution levels go beyond and to show a pattern of the air quality components periodically on the web platform. Components used to come up with the system include Arduino Uno microcontroller, MQ135 gas sensor, 16*2 liquid crystal display (LCD), ESP8266 Wi-Fi module, 10 K potentiometer, buzzer and a fan. Key findings within the study were the ability of the system to sound an alarm using a buzzer when harmful gases are detected and ventilation fan will turn on to extract the air.

In the first chapter, the background study, system goals, methods and tools used to develop the system, delimitations, research limitations, and research significance were discussed. The second chapter assessed the existing work versus the earlier work. The chapter represented the existing system that surmount the earlier problems. In the methodology chapter, the components that were used to come up with the system were discussed. The design phase of the system illustrated how the system was designed. Lastly the implementation phase, system testing, installation and maintenance were conducted.

DECLARATION

I, TINASHE .E. KUNAKA (R154336Q) hereby declare that I am the sole author of this thesis. I authorize the Midlands State University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Signature: .... Date: ..../.........../...........

APPROVAL

This dissertation entitled “IoT based air quality monitoring system” by Tinashe E Kunaka meets the expectations governing the award of the degree the BSC in Computer Science Honors Degree of the Midlands State University, and is approved for its contribution to knowledge and literal presentation. The writer also acknowledges the full ownership by Midlands State University of the ideology used.

Supervisor.....................................

DEDICATION

This project is dedicated to the Information Technology and air pollution monitoring sectors to which the technology can become a major building block of at large and my Computer Sciences lecturers that pushed me thereby making it possible for me to ensure the completion of these educational endeavours by imparting knowledge that was implemented.

ACKNOWLEDGEMENTS

To begin with, l humbly thank and praise the Lord Almighty GOD for his guidance in the development of this project from planning stage up until implementation stage. Without the guidance and support of my project supervisor, Mr. M. Giyane, the development of this project would not have been a success. I thank him for his devotion and untiring mentorship throughout the project development process. Sincere gratitude goes to my family for the outstanding moral and financial aid they extended to me only to see me through to the completion of this project. All the support and encouragement from Witness Jailos, Tatenda Jailos and all my friends was not in vain.

Table of Contents

List of figures

List of Tables

List of Appendices

CHAPTER ONE: INTRODUCTION

According to Diakopoulos (2018), air pollution is where substances or excess energy of daily activities are introduced to the atmosphere. This can immediately or obliquely change the air quality or result in unwanted effects on vegetation, man and animals. Internet of Things (IoT) is a growing field which can also be used in managing and protecting the world we live in. The research in this chapter aims to propose an IoT air pollution monitoring system, henceforth cover the background of the study, system objectives, methods and tools used to develop the system. Furthermore, delimitations, research limitations and research significance will also be described.

1.1 Background of the study

Air pollution is every nation's biggest problem, whether it is being developed or developed Health problems have increased more rapidly, particularly is urban areas of developing countries where industrialization and an increasing number of vehicles lead to release of many gaseous pollutants. Air pollution pollutants mainly come from production and energy use. To a larger degree, carbon dioxide is believed to be pollutant when planes, power plants, cars and human exercises which include burning of fossil fuels are involved. Fossils fuels include gasoline, natural gas, coal, petroleum among other pollutants. Other causes of air pollution include factory combustibles, sulphur dioxide which is released in the process of burning fossil fuels. According to Belloc, H. (2008), carbon monoxide is also another great cause of air pollution. Carbon monoxide mainly come from inappropriate combustion or vehicle emissions. Nitrogen oxides is another major pollutant, it is mostly produced by artificial and natural activities.

Different air quality monitoring has been taken. Sustainable shale energy resource environment development requires a strong understanding of potential emissions and the most advantageous air quality monitoring methods. Collecting measurements near the point of leakage is relatively easy, then identifying and characterizing such leaks.

Air quality monitoring is usually done by using Summa canisters or bags where an evacuated canister draws in the gas phase sample or a pump is used to fill the bag. It is then sealed and delivered for analysis to the laboratory. For the sampling of a known point source, summa canisters are suitable. Each sample provides each collection with a single data point. Although this method is rough and allows a sample to be kept up for 30 days, it is an inefficient method for continuous monitoring of potentially leaky infrastructure.

1.2 Problem definition

Over the past decades, there has been a huge growth in polluting industries as a result of civilisation and urbanization, burning of leaves and wastes openly, massive quantities of building waste leading to hazardous pollution. Air pollution has alarming effects. Its effects cause a lot of human diseases which include respiratory and heart conditions amongst other body threats. Pneumonia and asthma are also other diseases which attacks children born in areas that are exposed air pollutants. These harmful gases that are released into the atmosphere are also likely to cause acid rain. These pollutants will combine with water droplets and the rain becomes acidic. Acid rain mainly affect human, animals and crops. Therefore, the hazardous impacts of air pollution must be regularly monitored and reported. In order to monitor air quality, a new framework of an Internet of Things air pollution monitoring system is proposed to monitor the environmental parameters around us such as carbon monoxide, nitrogen oxide, benzene and alcohol using a sensor.

In the background of the study, the researcher discussed about other air quality monitoring systems which were implemented before and they have some drawbacks which are surpassed by the proposed system. Quantification which was discussed in the above section is generally hindered in many cases due to limited accessibility. For example, sources such as most gas pipelines can be located underground, and most natural gases infrastructure in sot readily accessible to researchers for safety and security reasons. The reality, is therefore, that fugitive emissions are detected by measuring above ground air concentrations and often at significant distances from the underground infrastructure.

Looking at summa canisters, they can provide only a snapshot of air quality when collecting samples, which could be greatly affected by changes in weather conditions, such as wind, and various human induced activities at the site. Their use can therefore provide an incomplete picture of the total air quality.

1.3 Aim

The main objective of this study is to develop an Internet of Things (IoT) based monitoring system for air pollution using Arduino Uno microcontroller and a webserver. A trigger will go off if the air quality on the liquid crystal display (LCD) exceeds a certain level measured in parts per million (PPM).

1.5 Objectives

- ❖ To monitor air quality as the system detects the most harmful gases like carbon dioxide, carbon monoxide, Sulphur dioxides and nitrogen oxides.
- ❖ To trigger devices such as the buzzer and a fan when the pollution goes beyond normal levels.
- ❖ To show a pattern of the air quality components periodically on the web platform**.**

1.6 Instruments and methods

1.6.1 Methods

A system model will be developed to give views to users and other students who may be willing to conduct a research in this area. The working and the system's significance will be shown by the small electrical components before the actual system is developed. The instruments and components will be briefly discussed below:

1.6.2 Instruments

<u>Programming language</u>

C- this is an extremely popular, simple and flexible programming language for general purposes. It is machine independent, structured language of programming that is widely used in different applications

1.6.3 Required Components

- ❖ Arduino Uno
- ❖ Gas sensor – MQ135
- ❖ Breadboard
- ❖ ESP8266 Wi-Fi module
- ❖ 16 X 2 Liquid Crystal Display
- ❖ Buzzer
- ❖ 220-ohm resistor

- 1K ohm resistors
- 10K potentiometer

1.8 Justification and rationale

In order to save lives of people and plants, it is good to control the level of air pollution. In developing countries, there’s already a low life expectancy by reducing the levels of air pollution more lives will be saved and hence increasing a life expectancy ratio. According to Europe statistics, there are more than 20 000 premature deaths in a year caused by air pollution. The decline in greenhouse gasses presented by climate change policies would result in a decline fall in air pollutants from fossil fuel combustion (most notably nitrogen oxides and Sulphur dioxide), and their associated health effects.

1.9 Conclusion

The developer is going to use the webpage the user interface. Among other designs, the developer will also design the circuits and the Arduino uno will work according to the commands. In the next chapter, the developer is going to talk about the literature review of the study where the proposed system will be compared to other existing systems.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

This chapter's main goal is to assess the existing work versus the earlier work. It accurately represents the existing system that surmount the earlier problems of the project and its restrictions. System overview, related works, other air quality monitoring systems and system evaluation will be discussed in this chapter.

2.2 Related Work

Air pollution has a major control over the concentration of atmospheric components resulting in results such as global warming and acid rain. An air pollution system is of the utmost importance to invalidate such antagonistic imbalances in nature. The research seeks to analyse past literature on air pollution monitoring systems and identify gaps which the researcher will use to implement the proposed system.

2.2.1 Wireless Sensor Network (WSN)

Wireless sensor network is a network consisting of a large number of mobile and static sensors, (Ghayvhat,2017). These sensors are used to record and monitor humidity, sound intensity, temperature, air quality, vibration intensity, pressure and speed and wind direction. Each and every sensor has important feature to compute, save and broadcast data.

Wireless sensor networks use a quite number of technologies which are localization, time synchronization, security administration, network protocol, power executives and data aggregation. This technology has some restrictions in storage, bandwidth energy and processing power and this makes it difficult to provide security to the network, (Weik, 2015).

2.2.2 Geographical Information System (GIS)

According to Panigraph (2016), this is a system which was implemented to measure air pollution of any area. GIS systems are made up of a microcontroller, a momentary memory buffer, mobile unit, gas sensors and a web server. Data is collected from different locations and posted to the web server through internet. The system's data for a precise location is averaged in a closed time and space. The system consists of a global positioning system (GPS) which provides accurate illustration of pollution sources in a certain location. A general packet radio service (GPRS) is used to transfer the recorded readings to a server. Finally, the data will be displayed on a website and the approved users with be able to access this data as shown by fig 2.2.

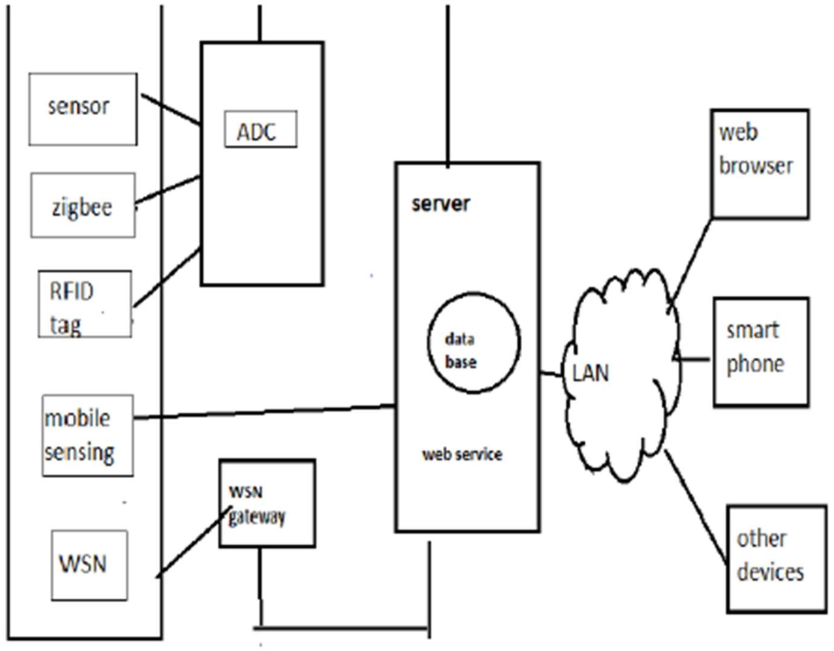


Fig 2.2: GPS module

2.2.2 Air Quality Monitoring Systems

Already used monitoring tools include mass spectrometers, gas chromatographs and Fourier transform infrared instruments and they output fairly correct air quality values. Other technologies that can be used to monitor air quality are electrochemical, catalytic bead and infrared. For the most part, they use smart transducer interface module with semiconductor gas sensors that use the quality of 1451.2. Smart interface transducer module has been established for an efficient an efficient monitoring system but for the power prerequisite and the ability to improve for large implementation.

A compact, robust gas sensor with flexible applications and low cost could be a reciprocal equally effective. A pollution control system depending on geo sensor network with control action and appropriate measurements rates proposed in cannot be an enormous deployment because of high costs, (Bekey, 2014).

2.3 Evaluation

Bearing in mind all the research that the developer undertook and several skills that have been applied, communication speed and components required to make up the proposed system needs to be taken into consideration. Since Wi-Fi is the pillar of all IoT projects, but precisely in Zimbabwe there are many reasons which are leading to the failure of these projects. The reasons include:

- **Internet access and speeds**

  Zimbabwean networks are much slower compared to other countries' networks, according to Powertel results in 2012. In rural areas, there is low network coverage, and in smaller towns, connections are often spotty or unreliable. Related difficulties occur in cities, where signals can be blocked and different signals strengths depending on the neighbourhood. And without consistent connection, real-time monitoring is essentially useless.

- **Hardware components costs**

  In Zimbabwe, currently we don't manufacture these Internet of Things components. These components have to be imported whenever we want to use them. This is the main challenge since it requires foreign currency to import the components.

- **Load shedding**

  According to Mahajan (2015), load shedding is a situation where available electricity is insufficient to meet all the customers' requirements, and an electricity utility will disrupt energy supply to certain areas. In Zimbabwe there is a high rate of load shedding and this affects the success of Internet of Things projects since they require electricity.

2.4 Foreknowledge to be conducted

2.4.1 Smart SIM cards over Wi-Fi

Since there are connectivity problems in most countries like Zimbabwe, we can choose to meet these challenges with an equally rigorous solution. Since connectivity is in many ways key to air quality monitoring, we can outfit our sensors with Smart SIM cards, (Ezhilarasi, 2018). These Smart SIM cards can automatically switch between networks to capture the best signal. This can be applied in large rural sites where signals can be weak and varied.

2.4.2 Power Saving

Cost saving measures will be important for the long-term viability of air quality control. It will be more cost effective for these projects to run on solar power. However, large projects that are run on solar power often necessitate an enormous number of solar cells, which can be prohibitively expensive. Devices that are able to operate on Low Power Wide Area (LPWA) networks have a distinct advantage, (Jimenez, 2016). LPWA is perfectly suited to handling the type of data coming from each individual air monitor and helps reduce the battery power used during transmission. This reduces cost, making the widespread adoption of IoT monitors more economically feasible, (Armada, 2007).

2.5 Conclusion

As a result of an increment in the use of Wi-Fi, air monitoring was made easier and that Internet of things devices can communicate with networked devices. In this chapter, the developer briefly discussed about other air pollution monitoring system that are currently used, the evaluation of the system and foreknowledge to be conducted to cater for these problems. In the next chapter, we are going to discuss about the components that are used to make up the proposed system.

CHAPTER THREE: METHODOLOGY

3.1 Introduction

This chapter analyses the base that is used to come up with the proposed system using Arduino. The components which are used are described in this chapter. To have a clear picture on how these components are going to be connected together, appropriate information about how the components basically work in needed.

3.2 Components Description

The main components and their proposed use are talked about below:

3.2.1 Control Unit: Arduino Uno R3

This is the primary component of the system. According to Diakopoulos, D. (2018), this microcontroller is a flexible hardware platform which can be used and it can be computed according to what it is to be used. Additionally, this device is open source which means that its code is available freely and it works perfectly with other components. It has 14 digital output/input pins, an on-board resonator, a reset button, 6 analog inputs and holes for mounting pin headers for related modules. A USB cable is required to connect the Arduino uno board directly to the computer. To deal with the microcontrollers, it requires at least 2 KB of the RAM, and 1 KB from the electrically erasable programmable read-only memory, 32 KB from the flash memory and lastly 0.5 KB which is required by the bootloader.

Fig 3.1: Arduino Uno R3

Source: Circuit Digest

3.2.2 Sensors: MQ135 Gas sensor

According to K SrinivasaSricharan (2014), the delicate material in this sensor is stannic oxide (Sn02). Its conductivity is extremely low in clean air and it rises as the concentration of the pollutant increases. It can monitor various types of toxic gases for instance ammonia gas, sulphide, carbon dioxide and benzene with a detection range of 10 – 10 000 ppm.

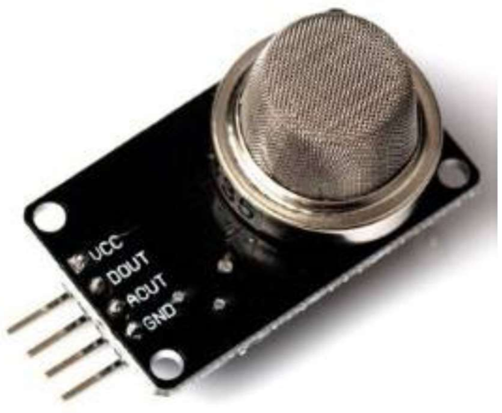


Fig 3.2: MQ135 gas sensor

Source: Circuit digest

3.2.3 Communication device: Wi-Fi Module ESP8266

According to Aubepart (2017), this is a self-contained system on a chip (SOC) with an incorporated IP protocol stack that is capable of giving Wi-Fi network access to any microcontroller. It is capable of either offloading all Wi-Fi networking functions from another application processor or hosting an application. This module comprises of numerous features although in this project we are only going to use the transmission control protocol/ internet protocol. This device is powered by 3.3 volts.

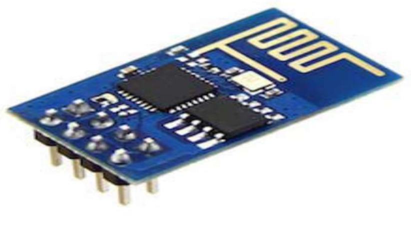


Fig 3.3: Wi-Fi module ESP8266

Source: Circuit digest

3.2.4 Display device: 16*2 Liquid crystal display (LCD)

This is a display device which will be used to output information in some sort of a visual form. 16*2 liquid crystal display simply means that the screen will output 16 characters in a line, Kushagra (2017). It can also output every character in a 5 * 7 picture element matrix when it is displaying alphanumeric characters and the alphanumeric display consists of two registers which are knowledge and command. The air quality will be displayed on the LCD in this project.

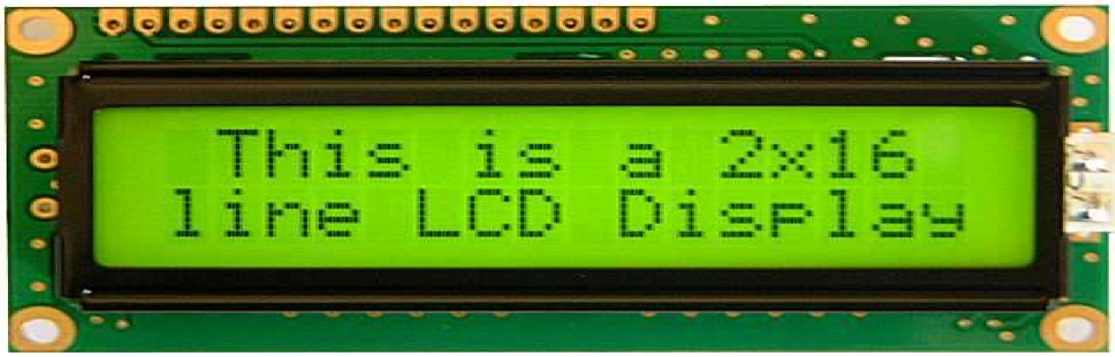


Fig 3.4: 16*2 LCD

Source: Circuit digest

3.2.5 Monitoring system

The air quality will be monitored over a webserver using internet. According to Forouzan (2009), Hyper Text Transfer Protocol (HTTP) is a correspondence convention which utilizes the Transmission Control Protocol (TCP) to transmit hypertext demands and information between web browsers and web servers. To monitor the air quality much easily, it will be displayed on the LCD and on the web pages. The air quality can be monitored from anywhere using a computer or a mobile as long as there is an internet connection.

3.2.6 Microcontroller programming tool: Arduino Integrated development environment (IDE)

According to Arduino Reference Guide (2019), this is a software development platform for the microcontroller hardware. This code editor and compiler is used to write C/C++ programming languages. To govern the functions of the Arduino board and how it is going to control other devices that are connected to it, it has to be programmed using this IDE. The IDE's main functions is to check for programmatic errors and then compile the code before it uploads the code to the Arduino board. The code is then uploaded using interfaces such as RS232 port or a USB cable depending on the version of the microcontroller.

3.2.7 Programming Languages: C/C++

According to C++ Institute (2019), these are high level programming languages that were established by an organization called Bell laboratories. At first, they established it as C for general purpose language which supports structured programming. Later on, object-oriented programming (OOP) features were added to it and C++ was established as the extension so that it can develop high-level code for controlling machinery.

3.3 Information gathering

The student used a number of techniques to collect the data. Below are the information gathering techniques used:

3.3.1 Sampling

The researcher did a sampling from 3 sites in Harare. The sampling was conducted in those areas with high industrial activities. Table 3.1 distinguish and describe these areas.

Table 3.1: Sampling Sites in Harare

Site	Site Description
Town House	This site is highly contaminated with emissions from motor vehicles since it is the city center of Harare.
Southerton	This site is encompassed by industries. Pollutants are transported by winds to the sampling area from Mbare and Southerton industries.
Mbare	This site is highly polluted by open burning of refuse which is done in the area and pollutants which come from the Graniteside industrial area. There is also a big bus terminus where air pollution is also coming from these vehicles.

3.3.2 Sampling results

- **Sulphur dioxide** – for Sulphur dioxide, the pollution levels was high in Southerton and low in Town house. Since southerton is surrounded by industries and some coal emissions from hospitals, the results obtained were. These industries also use fired coal which contains 2-3% Sulphur in Zimbabwe, Verma SK (2014. Vehicle emissions also contributed to high pollution levels in these areas particularly diesel vehicles.

In all the above sites, Sulphur dioxide concentration surpassed the World Health Organization daily minimum value of pollution.

- **Nitrogen oxide**

  For all the sampling sites there was a high nitrogen oxide concentration. Nitrogen, Sulphur dioxide among other pollutants were the major pollutants in Harare. All these pollutants surpassed the World Health Organization air quality daily minimum value.

3.4 Conclusion

It is very important to depict each and every component which is to be used in the proposed system. This will make it easy to work out the project since all the components are detailed. This chapter gives all inside and out knowledge on how to these electronic components work. The next chapter will be the design phase, the student will be talking about how these components are interconnected and the system design.

CHAPTER 4: DESIGN PHASE

4.1 Introduction

In this chapter, the researcher is going to discuss about how the proposed system is going to be designed. The system's circuit, architectural, physical and interface design are outlined in this chapter. The interaction of hardware and software is described.

4.2 Architectural design

This is a conceptual model that is used to determine the system's structure and behavior of the system (Orphanos, 2000). An architectural design describes and illustrate the system. The system's architecture is described in this project using a block diagram which shows how the hardware and software are related. Below is a block diagram which shows the interconnections of the components:

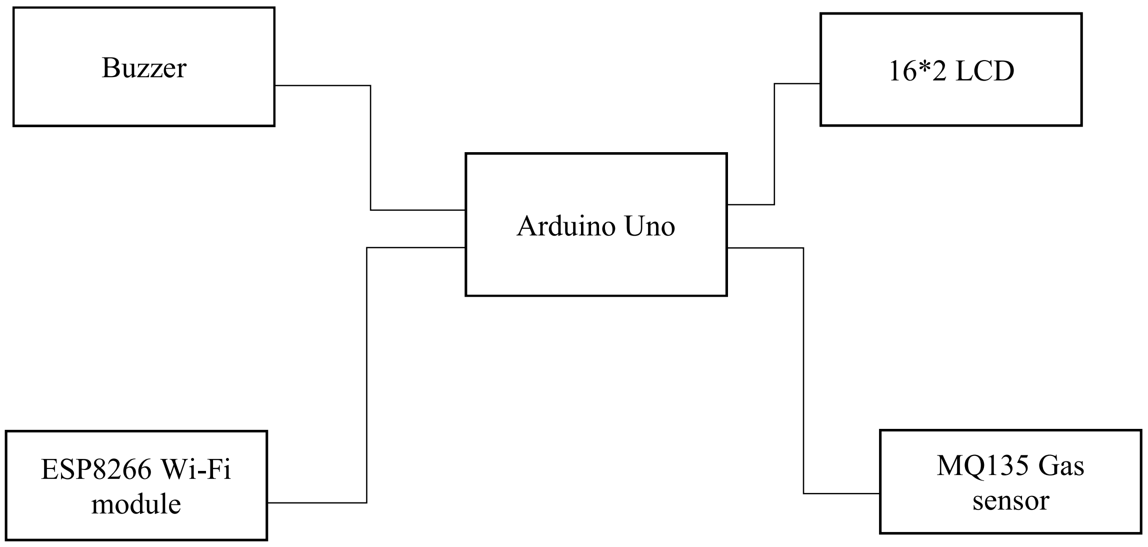


Fig 4.1: Block diagram

4.3 System design

The design of the proposed system is represented by a flow chart. A flow chart is an illustration that portrays the processes of a system step by step. Fig 4.2 is a flow chart which portrays the proposed system.

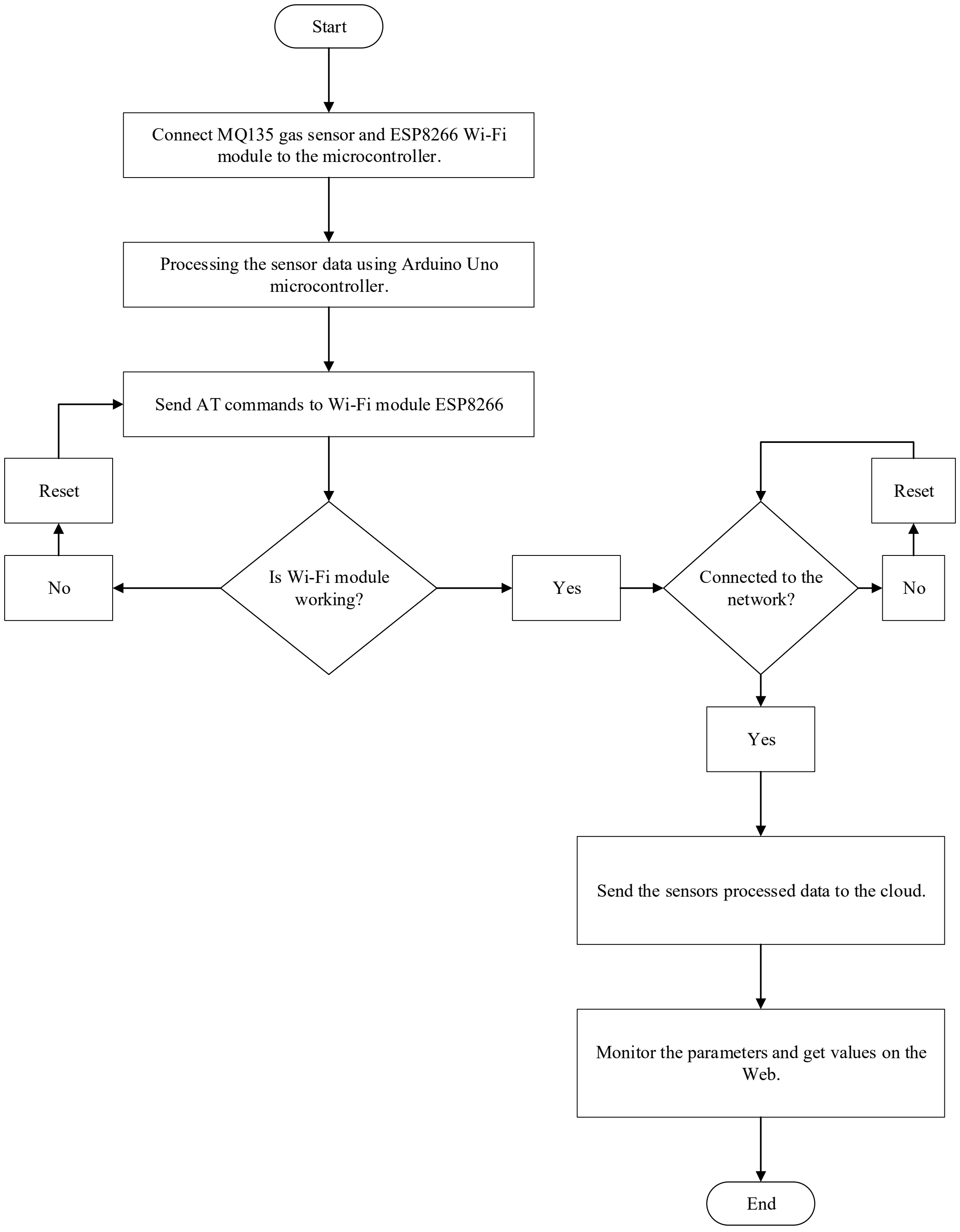

Start
Connect MQ135 gas sensor and ESP8266 Wi-Fi module to the microcontroller.
Processing the sensor data using Arduino Uno microcontroller.
Send AT commands to Wi-Fi module ESP8266
Reset
No
Is Wi-Fi module working?
Yes
Connected to the network?
No
Reset
Yes
Send the sensors processed data to the cloud.
Monitor the parameters and get values on the Web.
End

Fig 4.2: System flow chart

4.4 Circuit design

A circuit design is an important aspect in coming up with the system. It consists of all the components that were discussed in the previous chapter.

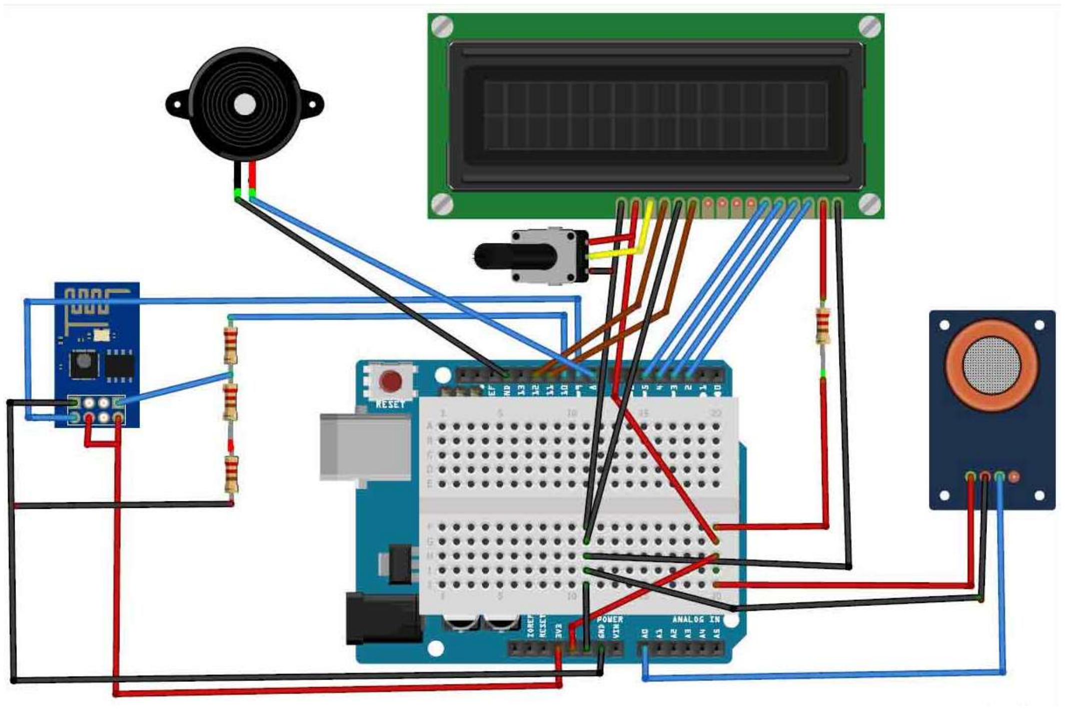


Fig 4.3: Circuit diagram

Source: Circuit digest

The circuits on fig 4.3 consist a MQ135 gas sensor, Arduino Uno, Wi-Fi module ESP8266, 16*2 Liquid Crystal Display (LCD), breadboard and a buzzer and these components were explained in the previous chapter. Table 4.1 gives details on how MQ135 gas sensor is connected to Arduino Uno.

Table 4.1 Arduino Uno – MQ135 Interfacing

MQ135 Pin	**Arduino Uno pin hole**
VCC	5V
GND	GND
A0	A0

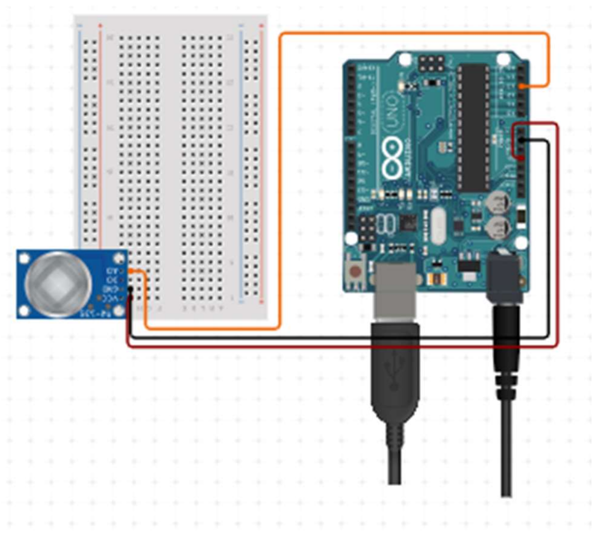


Fig 4.4: Arduino – MQ135 Interfacing

Source: Circuit digest

Table 4.2 details how a Wi-Fi module ESP8266 is connected to Arduino Uno.

Table 4.2 Arduino Uno – ESP 8266 Interfacing

ESP8266 Pin	**Arduino Uno pin hole**
VCC	3V
CH_PD	3V
GND	GND
RX	RX
TX	TX

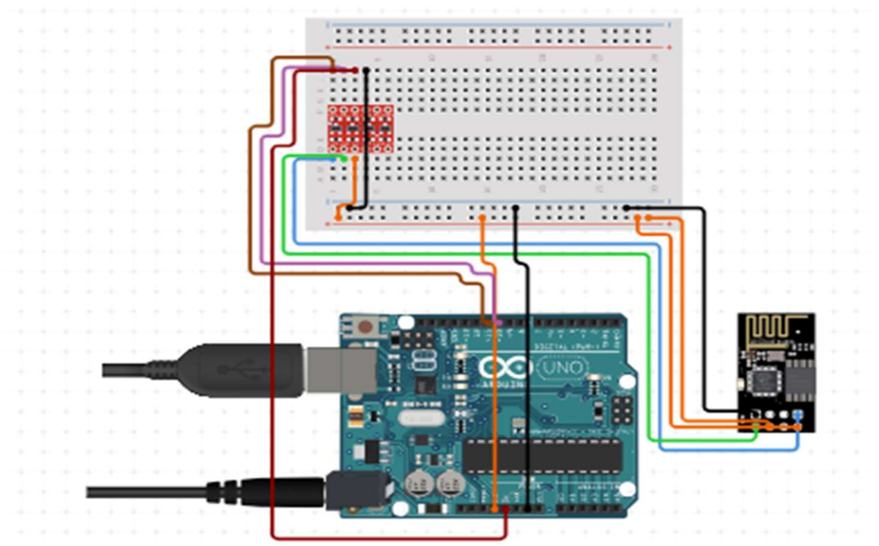


Fig 4.5: Arduino Uno – ESP8266 Interfacing

Source: Circuit digest

Table 4.3 details how a 16*2 LCD is connected to Arduino Uno.

Table 4.3 Arduino Uno – 16*2 LCD interfacing

16*2 LCD Pin	**Arduino Uno pin hole**
VEE	GND
VDD/ VSS	5V
RS	7
Read/ Write	GND
E	6
DB4	2
DB5	3
DB6	4
DB7	5
K	GND

A	220 ohm resistor

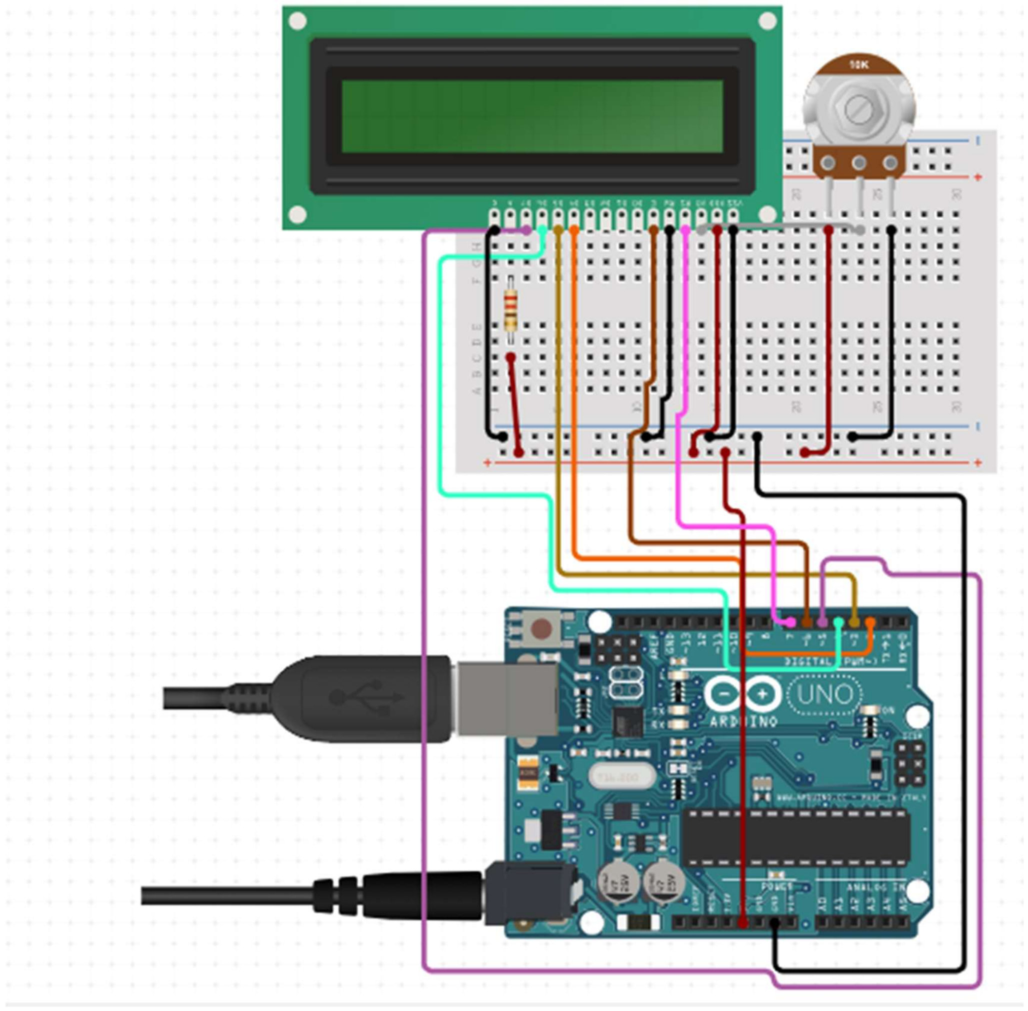


Fig 4.6: Arduino Uno – 16*2 LCD interfacing

Source: Circuit digest

Table 4.4 details how a 10K potentiometer is connected to Arduino Uno and 16 *2 LCD.

Table 4.4: Potentiometer- Arduino and LCD interfacing

10K potentiometer Pin	**Arduino Uno and 16 * 2 LCD pin hole**
0	GND (Arduino)

Buzzer Pin	**Arduino Uno pin hole**
Positive	5V
Negative	GND
SIG	V0 (LCD)
Vin	5V (Arduino)

Potentiometer connections are also shown in Fig 4.5.

Table 4.5 details how a buzzer is connected to Arduino Uno.

Table 4.5: Buzzer – Arduino Uno interfacing

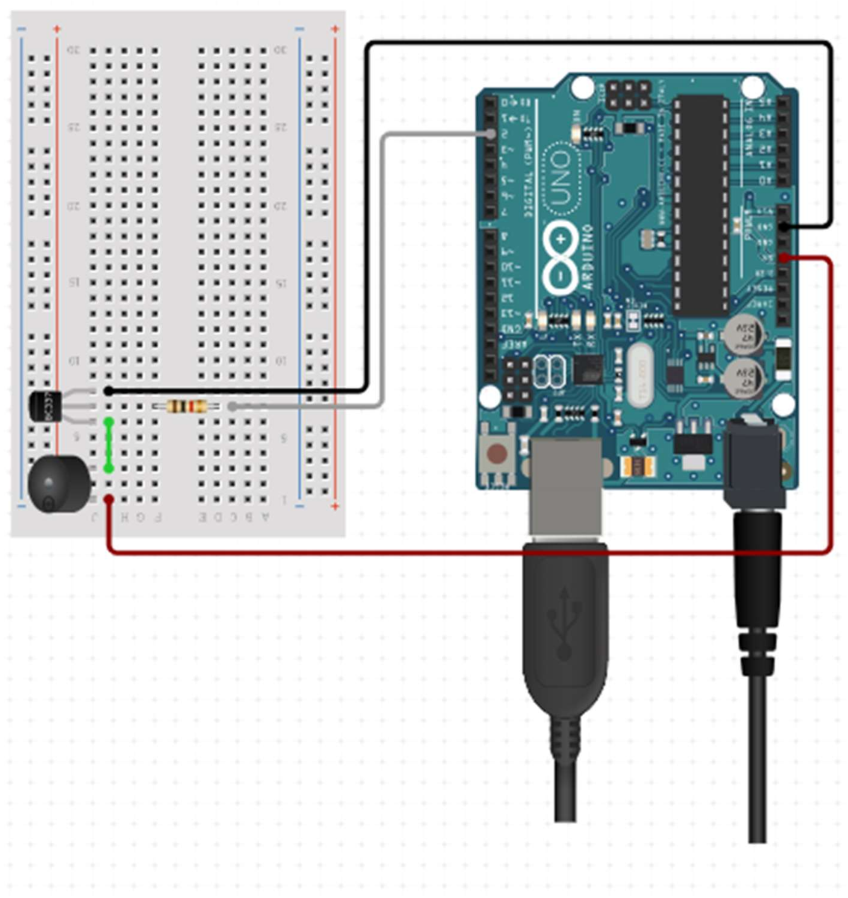


Fig 4.7: Arduino Uno - Buzzer Interfacing

Source: Circuit digest

4.5 Physical Design

The physical design of a system mainly focuses on the concrete input and output procedures of the system. The main focus is on how the system accepts the data, verifies the data, process and displays the output.

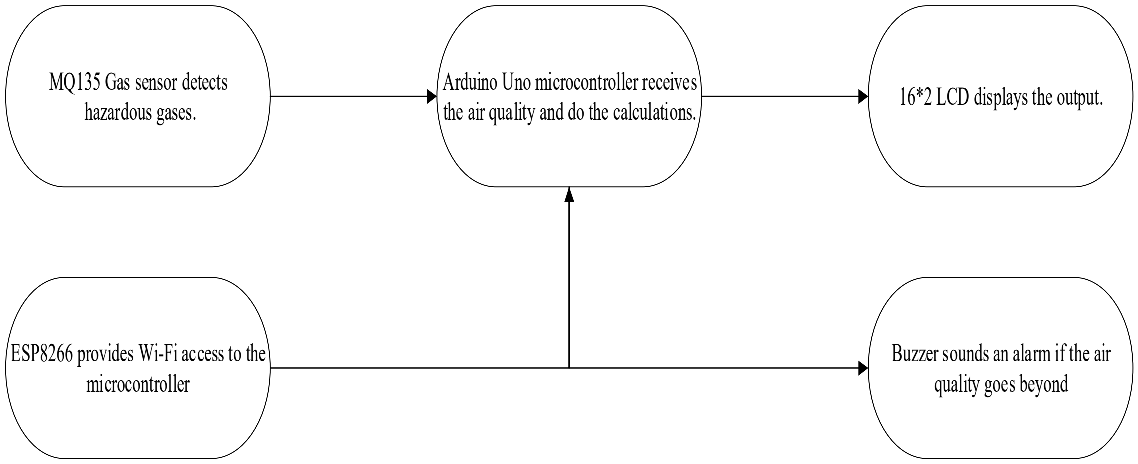

MQ135 Gas sensor detects hazardous gases.
Arduino Uno microcontroller receives the air quality and do the calculations.
16*2 LCD displays the output.
ESP8266 provides Wi-Fi access to the microcontroller
Buzzer sounds an alarm if the air quality goes beyond

Fig 4.8: Physical design of the system

4.6 Pseudo code

Pseudo code is a term often used in fields based on algorithms and programming. It is a methodology that enables the programmer to represent an algorithm's implementation. It is written in plain English as annotations and informative text. It does not have any syntax like any of the programming language and therefore the computer cannot compile or interpret it.

4.6.1 Pseudo code for MQ135

```
Initialize gas sensor
Initialize pins
If gas sensor connected
Print true
Else
Print false
```

4.6.2 Pseudo code for Wi-Fi setup

```
Send baud initialization
If
ESP8266 connected
Return ok
```

4.6.3 Pseudo code for Wi-Fi connection

```
Connection request from ESP8266 to Wi-Fi network
If connected
Print ok
Return true
Else
Return false
```

4.6.4 Pseudo code for turning buzzer on

```
Initialize buzzer
Initialize pins
If buzzer connected
Print true
Else
Print false
```

4.7 Conclusion

In this chapter, the developer was talking about the important aspects of the system design which are the system design, circuit design, architectural design and the physical design. These aspects essentially contribute in understanding the flow of the project. In the next chapter, the researcher will be discussing about the implementation of the system. The coding, testing, installation and maintenance of the system will be discussed in the next chapter.

CHAPTER 5: IMPLEMENTATION

5.1 Introduction

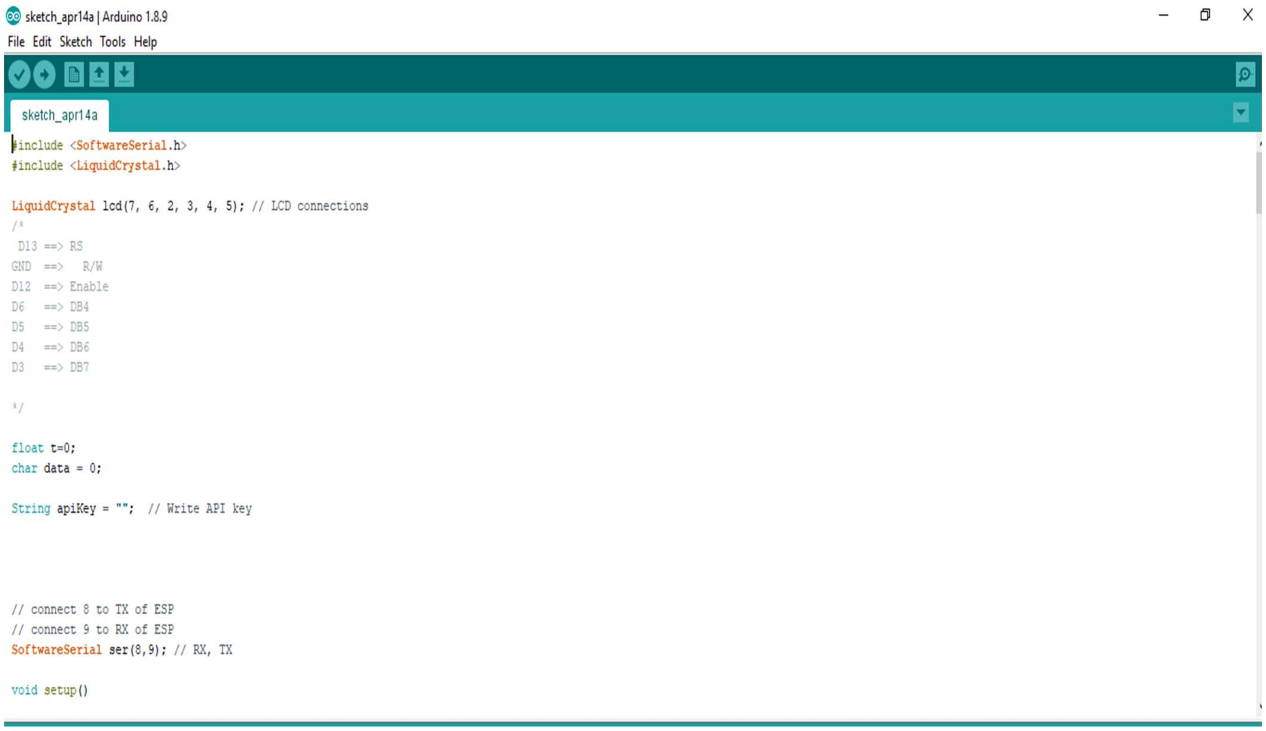
This is the final phase of the system development.in this phase testing, installation and maintenance of the new system is taken into consideration. under testing the users will be testing whether the developer have met the objectives as desired. Testing is done by the clients in the presents of the developer and under this stage loopholes in the project are detected and the developer will have to take corrective action .in addition, the maintenance of the project is analysed in the sense that the developer will have to see which maintenance strategy is best for the project.

5.2 Coding

Coding refers to a technical procedure that combines indeterminate data into easily enterable short letters or digits. The developer will achieve efficiency as one of the system goals by using coding. The primary objectives of coding are classification of information, tracking things to request appropriate actions, dissemination of information and masking of information, (Marcas, 2012). The developer used C programming language to program the Arduino microcontroller. Arduino IDE was also used as the integrated development environment.

5.2.1 Microcontroller source code

The capability for the air quality monitoring system to communicate with the sensors, send and receive data is done through the Arduino Uno microcontroller and it is the core of the system. C programming language is used to develop the code. The code is then uploaded to the microcontroller using the IDE. The code snippets of the system are illustrated below:

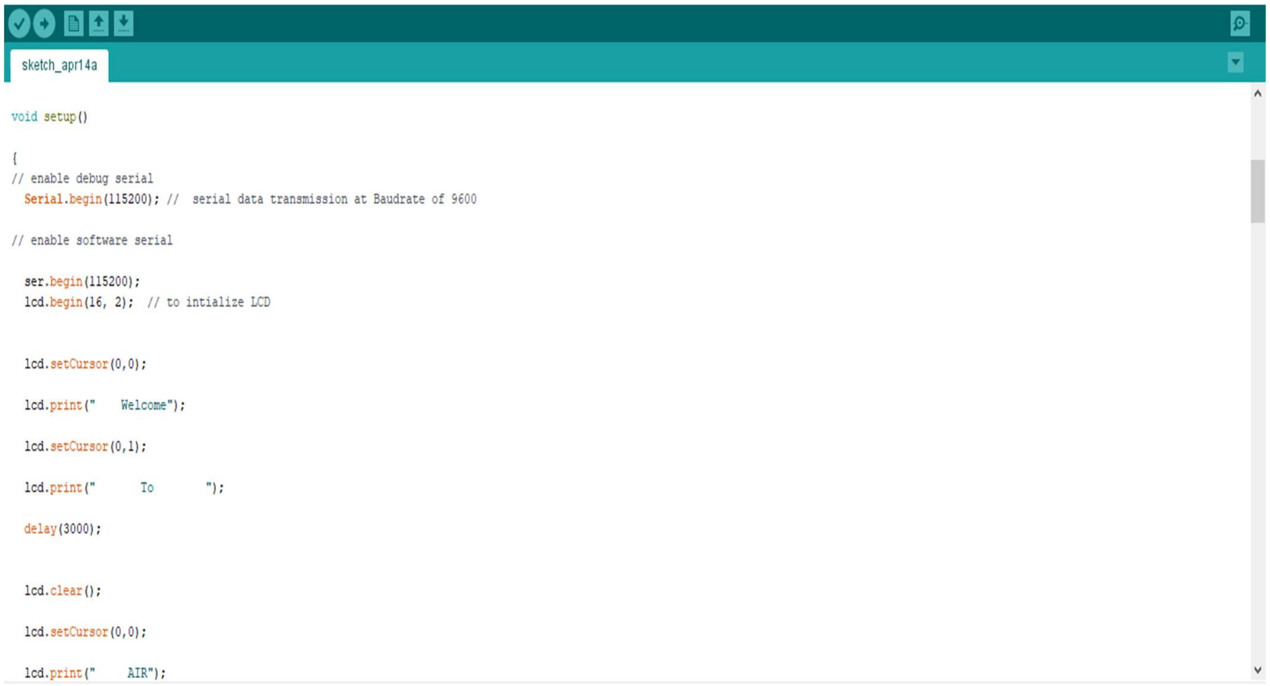

sketch_apr14a | Arduino 1.8.9

File Edit Sketch Tools Help

sketch_apr14a

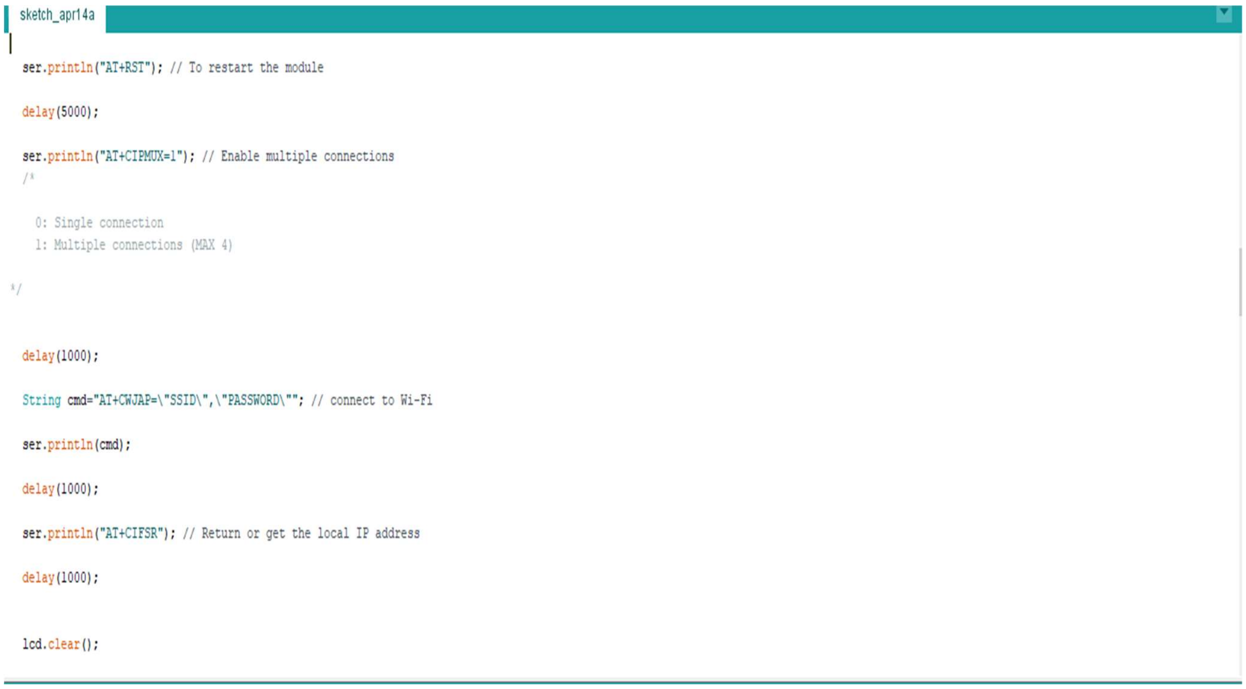
```
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(7, 6, 2, 3, 4, 5); // LCD connections
/*
 D13 ==> RS
GND  ==>   R/W
D12  ==> Enable
D6   ==> DB4
D5   ==> DB5
D4   ==> DB6
D3   ==> DB7

*/

float t=0;
char data = 0;

String apiKey = "";  // Write API key




// connect 8 to TX of ESP
// connect 9 to RX of ESP
SoftwareSerial ser(8,9); // RX, TX

void setup()
```

Fig 5.1: Microcontroller source code


sketch_apr14a

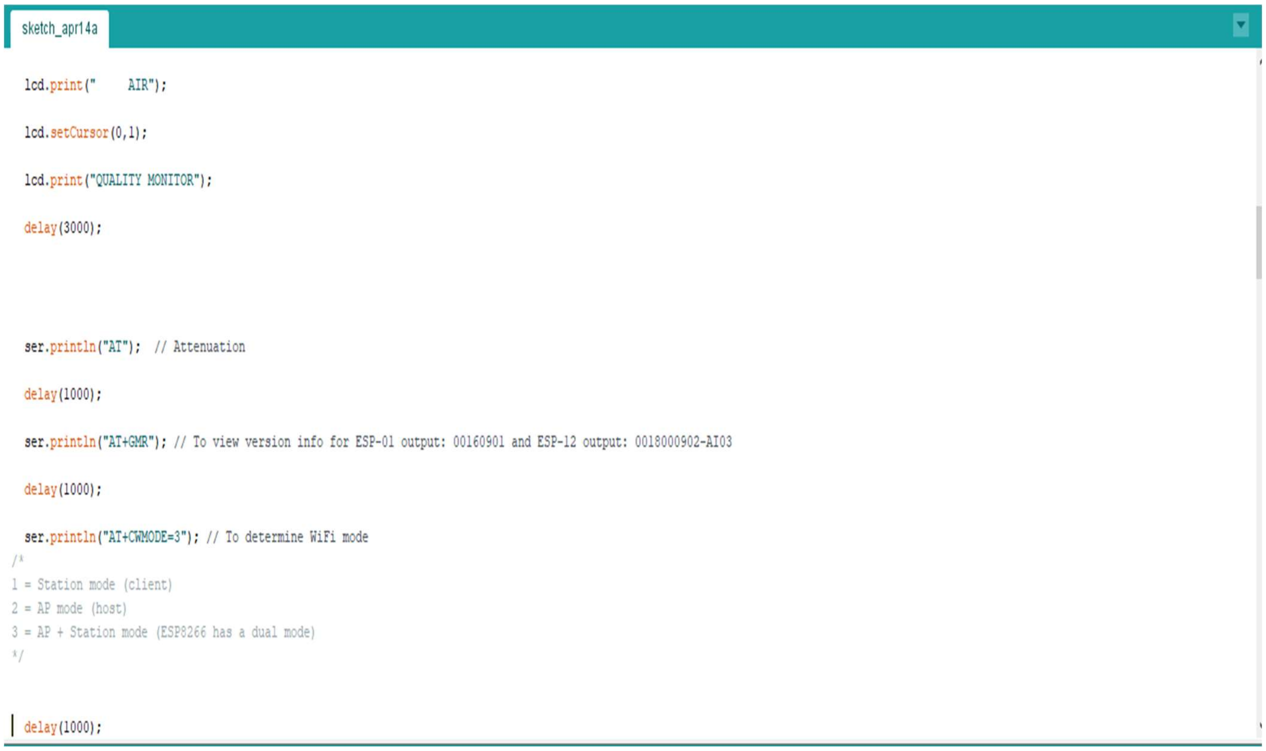
```
void setup()

{
// enable debug serial
  Serial.begin(115200); //  serial data transmission at Baudrate of 9600

// enable software serial

  ser.begin(115200);
  lcd.begin(16, 2);  // to intialize LCD


  lcd.setCursor(0,0);

  lcd.print("    Welcome");

  lcd.setCursor(0,1);

  lcd.print("       To        ");

  delay(3000);


  lcd.clear();

  lcd.setCursor(0,0);

  lcd.print("     AIR");
```

Fig 5.2: Microcontroller source code



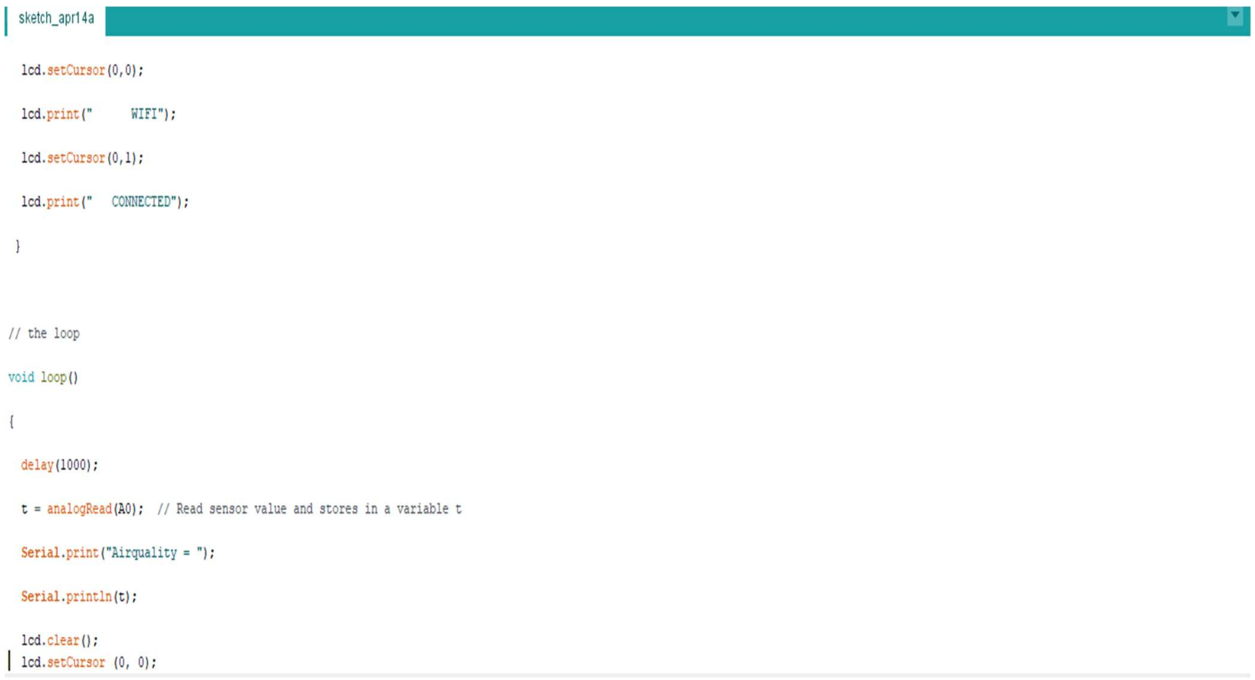
```
sketch_apr14a

ser.println("AT+RST"); // To restart the module

delay(5000);

ser.println("AT+CIPMUX=1"); // Enable multiple connections
/*

  0: Single connection
  1: Multiple connections (MAX 4)

*/


delay(1000);

String cmd="AT+CWJAP=\"SSID\",\"PASSWORD\""; // connect to Wi-Fi

ser.println(cmd);

delay(1000);

ser.println("AT+CIFSR"); // Return or get the local IP address

delay(1000);


lcd.clear();
```

Fig 5.3: Microcontroller source code



```
sketch_apr14a

lcd.print("    AIR");

lcd.setCursor(0,1);

lcd.print("QUALITY MONITOR");

delay(3000);



ser.println("AT");  // Attenuation

delay(1000);

ser.println("AT+GMR"); // To view version info for ESP-01 output: 00160901 and ESP-12 output: 0018000902-AI03

delay(1000);

ser.println("AT+CWMODE=3"); // To determine WiFi mode
/*
1 = Station mode (client)
2 = AP mode (host)
3 = AP + Station mode (ESP8266 has a dual mode)
*/


delay(1000);
```

Fig 5.4: Microcontroller source code



```
sketch_apr14a

  lcd.setCursor(0,0);

  lcd.print("     WIFI");

  lcd.setCursor(0,1);

  lcd.print("   CONNECTED");

 }


// the loop

void loop()

{

  delay(1000);

  t = analogRead(A0);  // Read sensor value and stores in a variable t

  Serial.print("Airquality = ");

  Serial.println(t);

  lcd.clear();
  lcd.setCursor (0, 0);
```

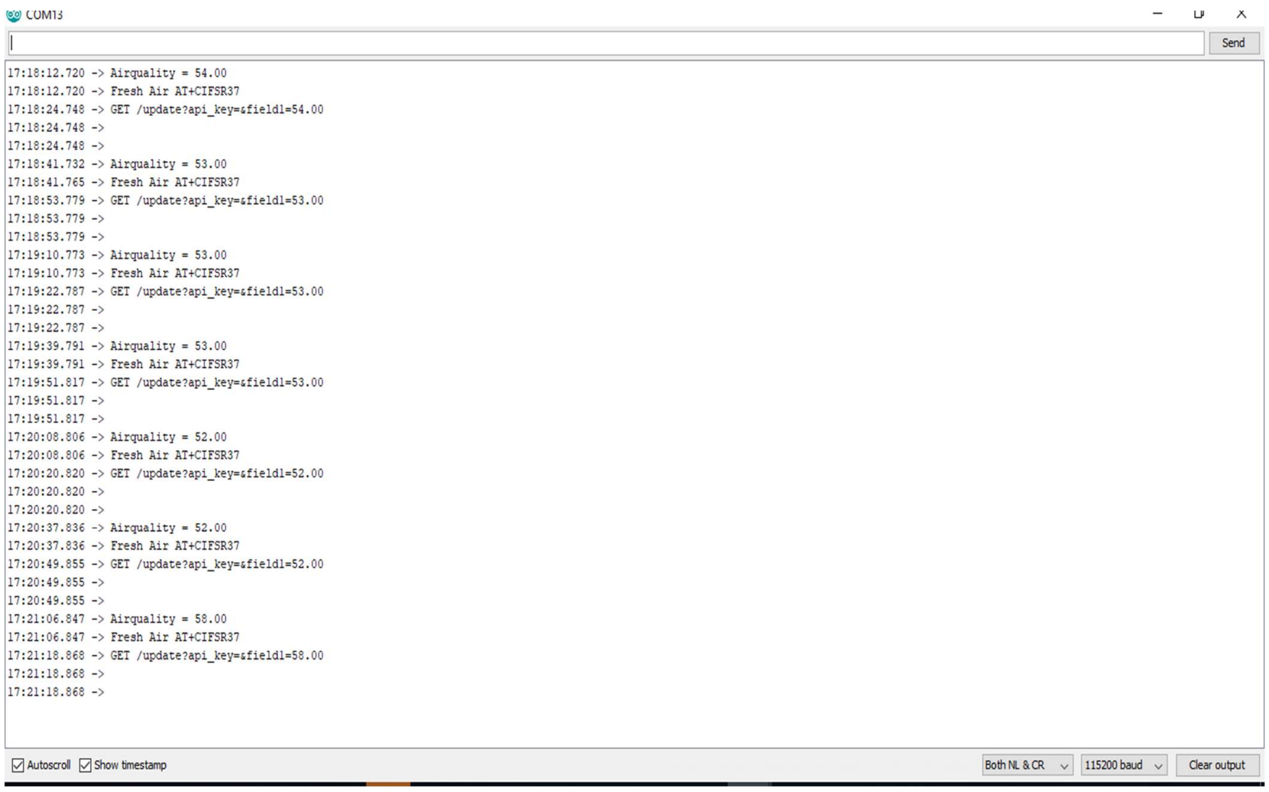
Fig 5.5: Microcontroller source code

5.3 Testing

This refers to a sequence of actions or steps that are taken to improve the software so that a very good quality software will be delivered, (Edwards, 2014). Software testing is done in the development of the system and in different modules of the software. In order to develop a functional software, the testing of the system interfaces between the sub-systems, the accuracy of output information and the importance of system documentation must be carried out in general. Testing should be done by well-trained personnel who will be able to learn the tasks of the system and come up with an end goal which is to come with an error free system without destroying the functionality of the system.

5.3.1 Microcontroller Testing

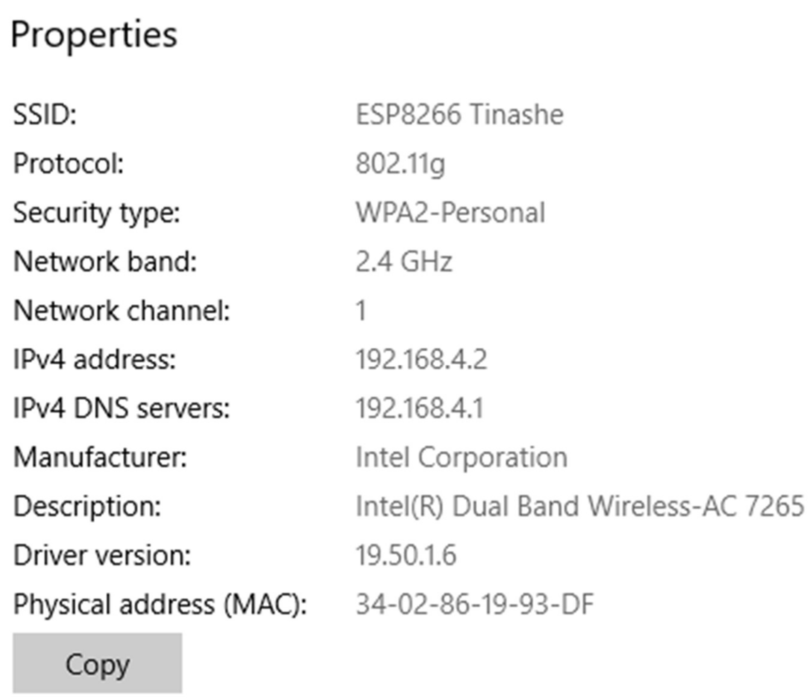
Microcontroller testing will focus on the sensors and ESP8266 Wi-Fi module testing. The Wi-Fi module will be tested to see if it is giving internet connection to the project and then send data to the server. MQ135 will be tested to see if its sensing air quality. Below is an image which illustrates the sensors air quality values printed to the serial monitor through Arduino.



```
17:18:12.720 -> Airquality = 54.00
17:18:12.720 -> Fresh Air AT+CIFSR37
17:18:24.748 -> GET /update?api_key=&field1=54.00
17:18:24.748 ->
17:18:24.748 ->
17:18:41.732 -> Airquality = 53.00
17:18:41.765 -> Fresh Air AT+CIFSR37
17:18:53.779 -> GET /update?api_key=&field1=53.00
17:18:53.779 ->
17:18:53.779 ->
17:19:10.773 -> Airquality = 53.00
17:19:10.773 -> Fresh Air AT+CIFSR37
17:19:22.787 -> GET /update?api_key=&field1=53.00
17:19:22.787 ->
17:19:22.787 ->
17:19:39.791 -> Airquality = 53.00
17:19:39.791 -> Fresh Air AT+CIFSR37
17:19:51.817 -> GET /update?api_key=&field1=53.00
17:19:51.817 ->
17:19:51.817 ->
17:20:08.806 -> Airquality = 52.00
17:20:08.806 -> Fresh Air AT+CIFSR37
17:20:20.820 -> GET /update?api_key=&field1=52.00
17:20:20.820 ->
17:20:20.820 ->
17:20:37.836 -> Airquality = 52.00
17:20:37.836 -> Fresh Air AT+CIFSR37
17:20:49.855 -> GET /update?api_key=&field1=52.00
17:20:49.855 ->
17:20:49.855 ->
17:21:06.847 -> Airquality = 58.00
17:21:06.847 -> Fresh Air AT+CIFSR37
17:21:18.868 -> GET /update?api_key=&field1=58.00
17:21:18.868 ->
17:21:18.868 ->
```

☑ Autoscroll ☑ Show timestamp   Both NL & CR   115200 baud   Clear output

Fig 5.6: Microcontroller testing



Properties

SSID:	ESP8266 Tinashe
Protocol:	802.11g
Security type:	WPA2-Personal
Network band:	2.4 GHz
Network channel:	1
IPv4 address:	192.168.4.2
IPv4 DNS servers:	192.168.4.1
Manufacturer:	Intel Corporation
Description:	Intel(R) Dual Band Wireless-AC 7265
Driver version:	19.50.1.6
Physical address (MAC):	34-02-86-19-93-DF

Fig 5.7: ESP8266 Internet connectivity

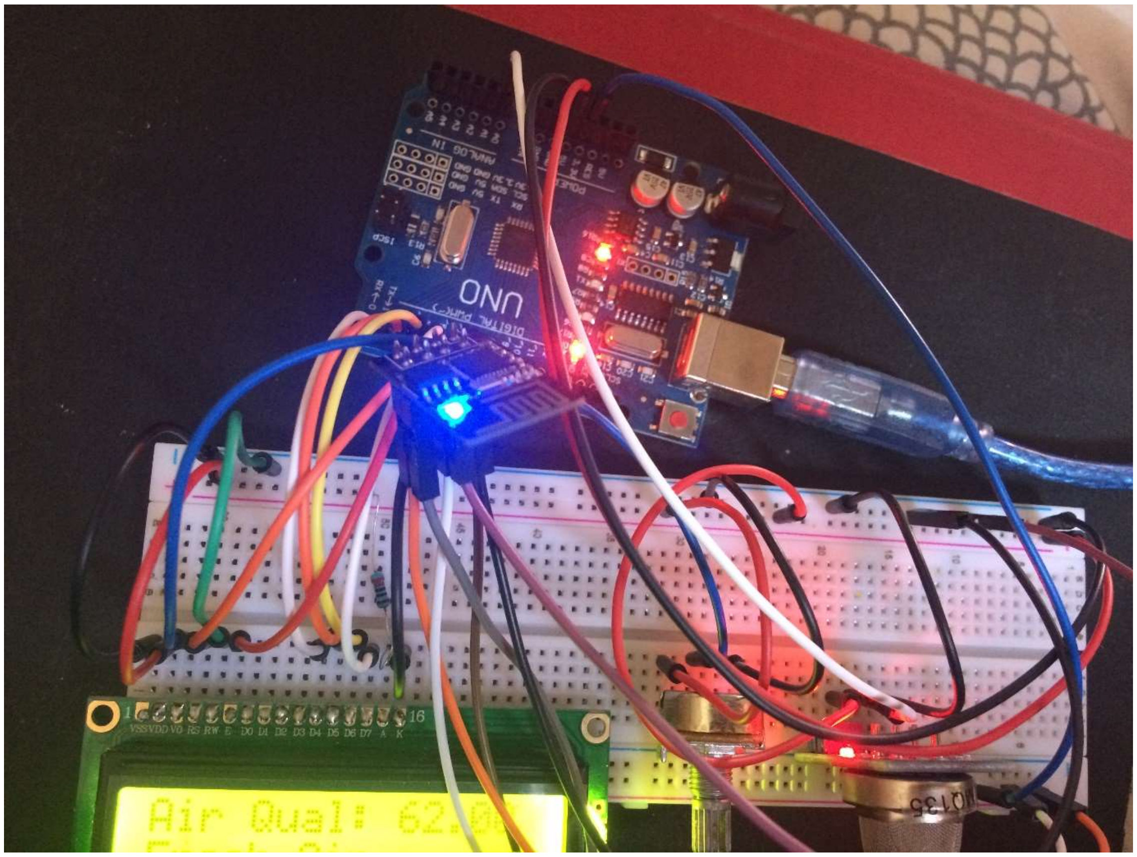


Fig 5.8: ESP8266 blinking a blue LED

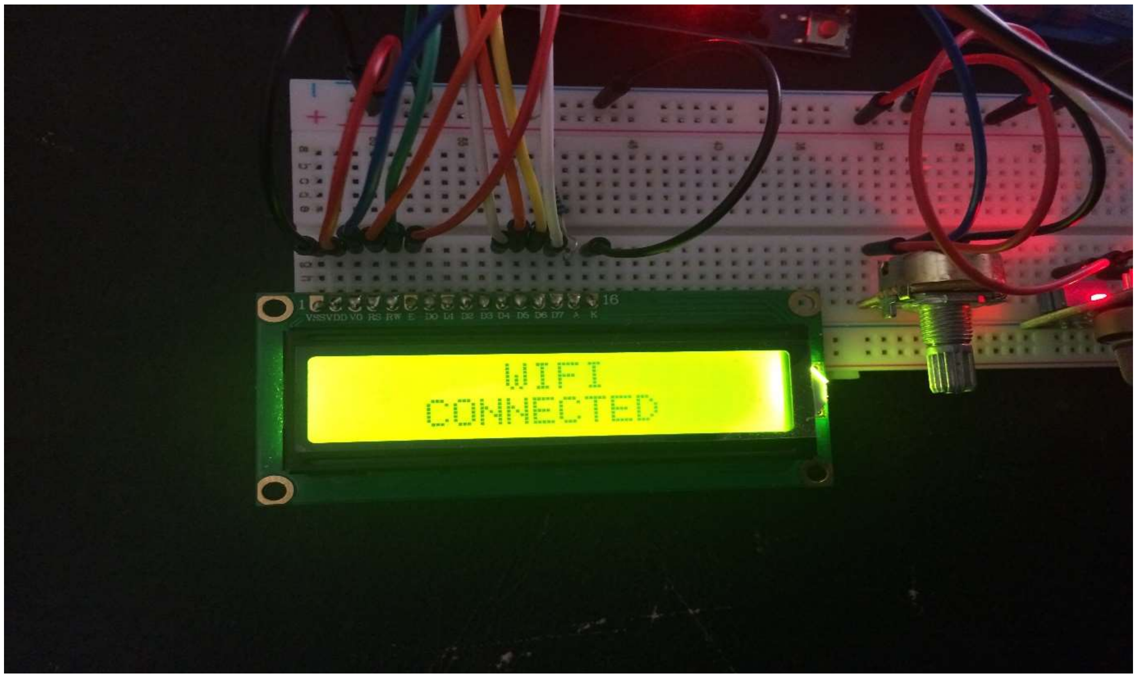


Fig 5.9: LCD displaying Wi-Fi connectivity.

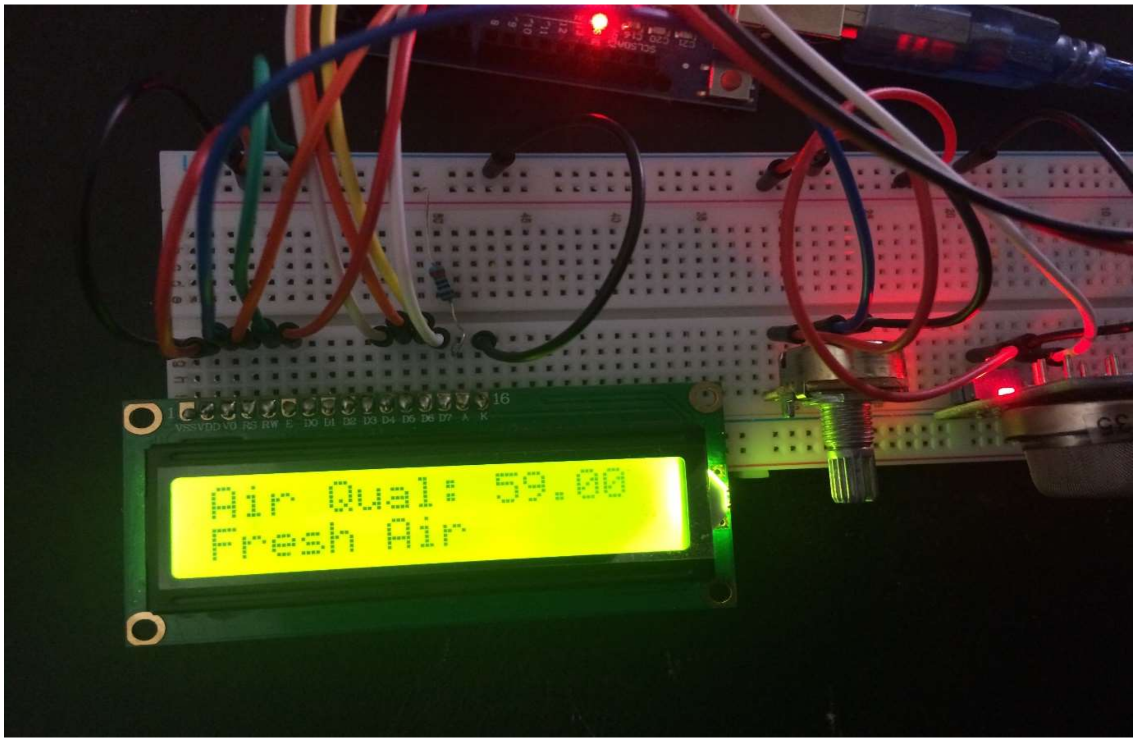


Fig A1: LCD displaying air quality.

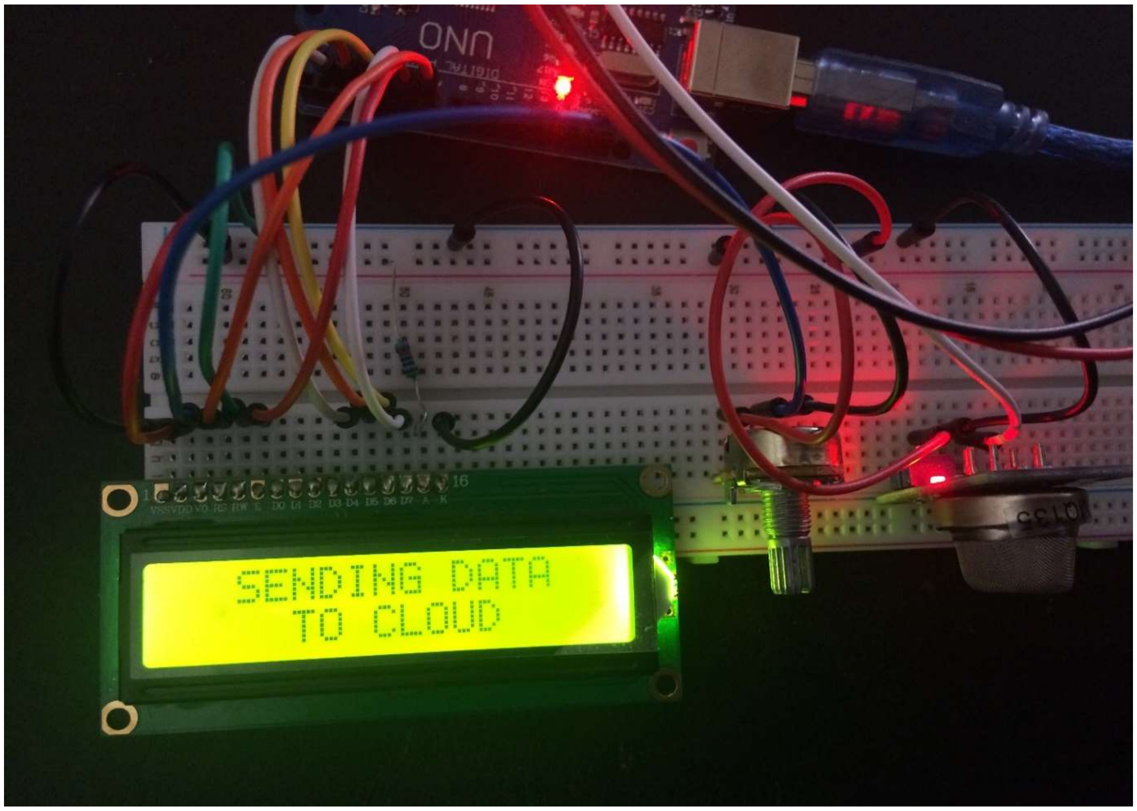
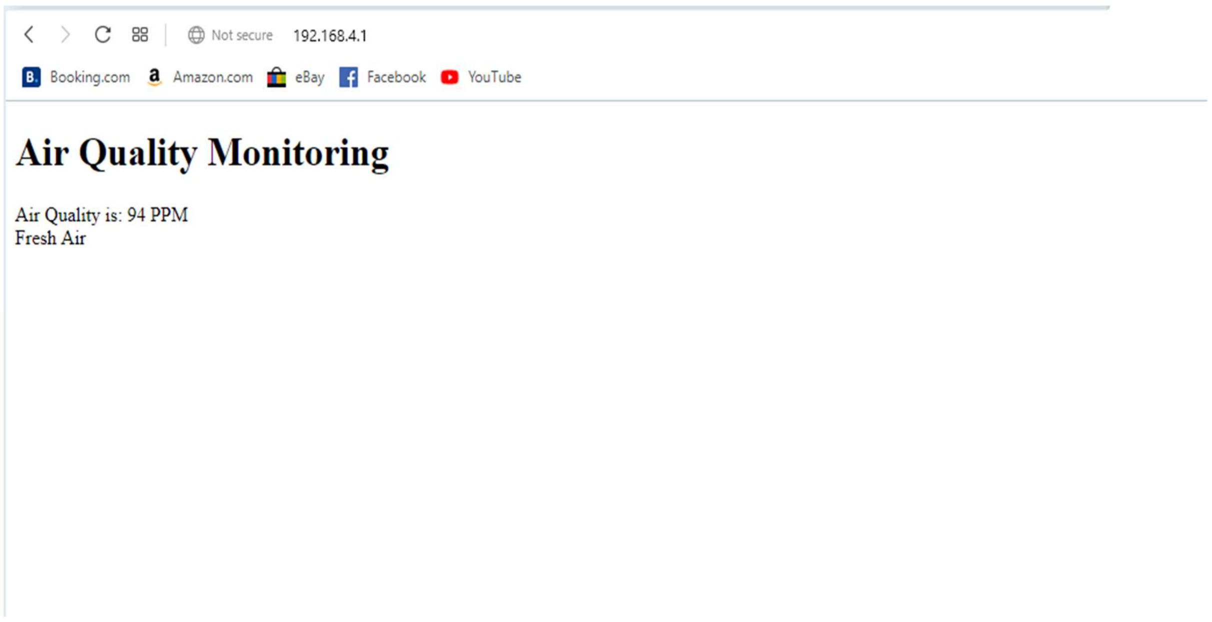


Fig A2: LCD showing that system is sending data to the server.



Air Quality Monitoring

Air Quality is: 94 PPM
Fresh Air

Fig A3: Webpage showing air quality from the server.

Images above shows the steps taken by the microcontroller from Wi-Fi connection to sending data to the server.

5.4 Installation

The system has to be installed to the working environment after testing. This includes the preparation of hardware and software features of the system for it to be used. Installation processes include user training and implementation strategies.

5.5 Implementation strategies

This is a method of moving from the old system to the new system. There are four approaches that can be used to do this process which are phased, direct conversion, pilot and parallel changeover. In this case, the developer chooses parallel changeover so that the air quality values can be compared using the old and new monitoring systems.

5.6 User training

User training is a process that provides the system user with enough skills and knowledge about how to use the new system, perform all the important tasks that the system can perform effectively and efficiently, (Norman,2017). In this case, the user only needs little knowledge about internet and web browser since the data will be displayed on the LCD and on the webpage.

5.7 Maintenance

Maintenance is defined as a process of monitoring and upgrading the system so that it will continue to work as required. This process requires both hardware and software to be upgraded so as to meet any upgrades in the operating and technical environment (George, 2015). Maintenance activities includes preventive, adaptive, corrective and perfective maintenance.

5.7.1 Preventive Maintenance

The main objective of preventive maintenance is to carry out those activities that could cause system malfunction in order to increase the system's lifespan. Activities to do include upgrading the system's hardware and software as well as documenting the system.
Preventive maintenance aims at anticipating the system's errors and correct them before they basically occur.

5.7.2 Adaptive Maintenance

Adaptive maintenance is performed to adapt the current system to the new working environment of the system. It focuses primarily on changing the functionality or adding new features to the system to suit the working environment, (Kendall,2012).

5.7.3 Corrective Maintenance

This is a method that is undertaken to resolve system runtime errors and others that may not been identified during system testing (Stellman and Greene, 2014). The system should always meet the requirements after making the adjustments. Corrective maintenance does not introduce new features and value to the system and organisation but only correct errors.

5.7.4 Perfective Maintenance

Perfective maintenance includes adjustments that needs to be done for the system to improve its performance. The main goal is to improve the functionality of the system. After the user's feedback, perfective maintenance also aims at improving the system.

5.8 Recommendations for future development

The developer recommends the use of a quite number of gas sensors to measure air quality. Mobile Internet of things sensors can also be used so as to generate massive amounts data on air pollution. If we understand what the data says, we can be able to encourage families, industries and schools to do thing differently to reduce the exposing of dangerous pollutants to the atmosphere. These mobile IoT sensors can be taken virtually anywhere and they will be able to take measurements in real time.

This project is going to receive a lot of data, the use of raspberry pi microcontroller is very effective since it uses python operating system. This operating system makes it easy to handle large volumes of data.

The use of machine learning and big data analytics can also be used to analyse large volumes of data generated by the sensors. This can help us better in understanding and predict the air quality. Since machine learning uses computational techniques to find the hidden patterns in data, we can also predict what the air quality will be in the future.

The developer faced challenges interfacing Wi-Fi module to Arduino uno microcontroller. A more advanced board with Wi-Fi capabilities should be used in the future.

To reduce the number of wires and lose connection, a printed circuit board (PCB) should be used.

5.9 Conclusion

Finally, we have come to the end of the project. This chapter conducted several types of software testing to ensure that the new system will work perfectly. The developer also discussed different techniques of implementing the system as well as maintenance operations that can be conducted to maintain the system.

Reference List.

Arduino Open Projects *http://nevonprojects.com/year-projectsforcomputer engineering/*

Diakopoulos, D. (2018). *On*. Freeport, N.Y.: Books for Libraries Press.

Belloc,H (2008), "*IoT Based Smart Home Using Blynk Framework*", ZERONE SCHOLAR, VOL. 1, NO. 1, NOVEMBER 2008

Ghayvha, (2017) *"Air pollution monitoring* ",International Journal of Research in Engineering and Technology, Pune, Maharashtra, India

Weik J, (2015) "*Wireless sensor networks*" International Journal of Electronics Communication and Computer Technology (IJECCT), Vol.3,Issue 2, pp.382-385

Panigraph R. (2016). *Geograpgical Information System* International Journal of Electronics Communication and Computer Technology (IJECCT

Ezhilarasi, T. (2018). *Smart sim cards*. Berlin: Springer.

El K, Ahmed E (2012) *"Design and Prototype Implementation of SMS Based Home Automation System*" IEEE International Conference on Electronics Design, Systems and Applications (ICEDSA) pp. 162-167

HIRAL S. DOSHI, MINESH S. SHAH, UMAIR S A. SHAIKH (2017), "INTERNET of THINGS (IoT): INTEGRATION of BLYNK for DOMESTIC USABILITY", VJER-Vishwakarma Journal of Engineering Research Volume 1 Issue 4, December 2017

Jayavardhana G,Rajkumar B, Slaven M, Marimuthu P, "*Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions*". Volume 2, Issue 12, December 2014 International Journal of Advance Research in Computer Science and Management Studies

Kanigoro B et al (2015), *Design and Implementation of Web Based Home Electrical Monitoring, Diagnosing and Control System*.

Kyu H and Jin-Wook B(2007) "*wireless access monitoring and control system based on digital door lock" IEEE Transactions on Consumer Electronics*, Vol. 53, No. 4, pp.1724-1730

Latchman, H. (2013). *Homeplug AV and IEEE 1901*. Hoboken, N.J.: Wiley.

Neng-Shiang L, Li-Chen F, Chao-Lin W (2002) "*An integrated, flexible and Internet based control architecture for home automation system in the internet era*" 2002 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1101 – 1106

Piyare R. and Tazil M (2011)., "*Bluetooth based home automation system using cell phone," in Consumer Electronics (ISCE), 2011 IEEE 15th International Symposium on*, 2011, pp. 192195.

Potts J and Sukittanon S (2012), "*Exploiting Bluetooth on Android Mobile Devices for Home Security Application"*, University of Tennessee at Martin Department of Engineering Martin, TN USA

Rakesh K. D, Vijay R. S, Pooja H. Satpute (2015) *"Internet of Thing Based Home Appliances Control*" International Conference on Computational Intelligence and Communication Networks (CICN), pp. 898-902

Sachin K (2014),"*Home Appliances Control System Based on Android Smart phone*", Department of Electronics and Communication NBN Sinhgad School of Engineering, Pune, Maharashtra, India.

Shengwen C, Chunghuang Y, Chung-Huang Y (2011). "*Design and Implementation of Live SD Acquisition Tool in Android Smart Phone*". 2011 Fifth International Conference on Genetic and Evolutionary Computing. pp. 157-162.

Sriskanthan N and Tan K (2002), "*Bluetooth Based Home Automation System", Journal of Microprocessors and Microsystems*, Vol. 26, pp.281-289

Scheckel, P. (2008). *The home energy diet*. Gabriola Island, B.C.: New Society Publishers.

Tan, K. and Maxion, R. 2002. "Why 6?" *Defining the Operational Limits of Stide an AnomalyBased Intrusion Detector. In Proceedings of the IEEE*

Xiao Yuan, Yuliang Pan, Zaiying Ling. "*The Application of Infrared Remote Controlled Code Lock in the Management of Industrial Machine Parameters*" .Electrical and Control Engineering (ICECE), 2011 International Conference on.2011, pp. 418-421.

Wagner, D. and Soto, P. 2002. *Mimicry Attacks on Host Based Intrusion Detection Systems. In Proceedings of the 9th ACM Conference on Computer and Communications Security*. Washington DC, USA, 255–

Appendix A: Microcontroller Code

```
#include "WiFiEsp.h"
#include "MQ135.h"
#include <MySQL_Connection.h>
#include <MySQL_Cursor.h>
const int sensorPin= 0;
int air_quality;
#include <LiquidCrystal.h>

// Emulate Serial1 on pins 6/7 if not present
#ifndef HAVE_HWSERIAL1
#include "SoftwareSerial.h"
SoftwareSerial Serial1(9, 10); // RX, TX
#endif

LiquidCrystal lcd(7,6, 2, 3, 4, 5);
String air_quality_status;
void setup() {
pinMode(8, OUTPUT);
lcd.begin(16,2);
lcd.setCursor(0,0);
lcd.print("    Welcome");
lcd.setCursor(0,1);
lcd.print("      To       ");
delay(3000);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("     AIR");
lcd.setCursor(0,1);
lcd.print("QUALITY MONITOR");
delay(1000);
Serial.begin(115200);
setupWiFi();
//setupSQL();
pinMode(sensorPin, INPUT);        //Gas sensor will be an input to the arduino
lcd.clear();
}

void loop() {

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

lcd.setCursor (0, 0);
lcd.print ("Air Qual:");
lcd.print (air_quality);
lcd.print (" PPM");
lcd.setCursor (0,1);
```

```
if (air_quality<=100)
{
air_quality_status = "Fresh Air";
digitalWrite(8, LOW);
}
else if( air_quality>=100 && air_quality<=179 )
{
air_quality_status = "Poor Air, Open Windows";
digitalWrite(8, HIGH );
}
else if (air_quality>=180 )
{
air_quality_status = "Danger! Move to Fresh Air";
digitalWrite(8, HIGH);   // turn the LED on
}
lcd.print(air_quality_status);
//lcd.scrollDisplayLeft();
loopWiFi();
//loopSQL();
delay(1000);
}


char ssid[] = "Air Quality";         // your network SSID (name)
char pass[] = "12345678";        // your network password
int status = WL_IDLE_STATUS;     // the Wifi radio's status
int reqCount = 0;                // number of requests received

WiFiEspServer server(80);

// use a ring buffer to increase speed and reduce memory allocation
RingBuffer buf(8);

void setupWiFi()
{
 Serial.begin(115200);   // initialize serial for debugging
 Serial1.begin(38400);    // initialize serial for ESP module
 //Serial.println("This is not it");
 Serial1.println("AT");
 delay(100);
 while(Serial1.available()){
  Serial.write(Serial1.read());
 }
 WiFi.init(&Serial1);    // initialize ESP module

 // check for the presence of the shield
 if (WiFi.status() == WL_NO_SHIELD) {
  Serial.println("WiFi shield not present");
  while (true); // don't continue
 }
```

```
  Serial.print("Attempting to start AP ");
  Serial.println(ssid);

  // uncomment these two lines if you want to set the IP address of the AP
  //IPAddress localIp(192, 168, 111, 111);
  //WiFi.configAP(localIp);

  // start access point
  status = WiFi.beginAP(ssid, 10, pass, ENC_TYPE_WPA2_PSK);

  Serial.println("Access point started");
  printWifiStatus();

  // start the web server on port 80
  server.begin();
  Serial.println("Server started");
}


void loopWiFi()
{
  WiFiEspClient client = server.available();  // listen for incoming clients

  if (client) {                               // if you get a client,
    Serial.println("New client");             // print a message out the serial port
    buf.init();                               // initialize the circular buffer
    while (client.connected()) {              // loop while the client's connected
      if (client.available()) {               // if there's bytes to read from the client,
        char c = client.read();               // read a byte, then
        Serial.print(c);
        buf.push(c);                          // push it to the ring buffer
        // you got two newline characters in a row
        // that's the end of the HTTP request, so send a response
        //if (buf.endsWith("\r\n\r\n")) {
        Serial.println("Mih");
        sendHttpResponseToClient(client);
        break;
        //}
      }
    }

    // give the web browser time to receive the data
    delay(10);

    // close the connection
    client.stop();
    Serial.println("Client disconnected");
  }
}
```

```
bool sendHttpResponseToClient(WiFiEspClient client)
{
  client.print(
    "HTTP/1.1 200 OK\r\n"
    "Content-Type: text/html\r\n"
    "Connection: close\r\n"  // the connection will be closed after completion of the response
    "Refresh: 20\r\n"        // refresh the page automatically every 20 sec
    "\r\n");
  client.print("<!DOCTYPE HTML>\r\n");
  client.print("<html>\r\n");
  client.print("<h1>Air Quality Monitoring </h1>\r\n");
  client.print("Air Quality is: ");
  client.print(air_quality);
  client.print(" PPM");
  client.print("<br>\r\n");
  client.print(air_quality_status);
  client.print("<br>\r\n");
  client.print("</html>\r\n");
}

void printWifiStatus()
{
  // print your WiFi shield's IP address
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print where to go in the browser
  Serial.println();
  Serial.print("To see this page in action, connect to ");
  Serial.print(ssid);
  Serial.print(" and open a browser to http://");
  Serial.println(ip);
  Serial.println();
}
```