NOISE AND AIR POLLUTION MONITORING SYSTEM USING SENSORS

Main Project Report

Submitted by

ANGEL A B

Reg no: FIT20MCA-2023

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications

Of

A P J Abdul Kalam Technological University

Focus on Excellence

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY-683577, ERNAKULAM(DIST)

JULY 2022

DECLARATION

I hereby declare that the report of this project work, submitted to the Department of Computer Applications, Federal Institute of Science and Technology (**FISAT**), Angamaly in partial fulfillment of the award of the degree of Master of Computer Applications is an authentic record of my original work.

The report has not been submitted for the award of any degree of this university or any other university.

Date : **Signature :**

Place: Angamaly **Name :**

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY, ERNAKULAM-683577

Focus on Excellence

CERTIFICATE

This is to certify that the project report titled **"NOISE AND AIR POLLUTION MONITORING SYSTEM USING SENSORS"** submitted by **ANGEL A B**, **(Reg No: FIT20MCA-2023)** towards partial fulfillment of the requirements for the award of the degree of Master of Computer Applications is a record of bonafide work carried out by her during the year 2022.

Project Guide	**Head of the Department**
Ms. Joice T	**Dr. Deepa Mary Mathews**

Submitted for the viva-voce held on.............................. at ....

Examiner :

ACKNOWLEDGEMENT

Gratitude is a feeling which is more eloquent than words, more silent than silence. To complete this project work I needed the direction, assistance and co-operation of various individuals, which is received in abundance with the grace of God.

I hereby express my deep sense of gratitude to **Dr. Manoj George** Principal, FISAT and **Dr. C Sheela** Vice principal, FISAT for allowing me to utilize all the facilities of the college.

My sincere thanks to **Dr. Deepa Mary Mathews** HOD, Department of Computer Applications FISAT ,who had been a source of inspiration. I express heartiest thanks to **Ms. Joice T** my internal guide for his encouragement and valuable suggestion. I express my heartiest gratitude to my scrum master **Ms. Senu Abi** and the faculty members in my Department for encouragement and constructive suggestions and comment during the project work.

Finally I wish to express my thanks to my parents, friends and well-wishers who extended their help in one way or other in preparation of my project. Besides all, I thank God for everything.

ABSTRACT

In recent day scenarios, the non-stop increase in air and sound pollution prove to be a big alarming problem. It has become mandatory to control and appropriately monitor the situation so that the required steps to control the situation can be undertaken. In this project, an IOT-based method is used to monitor and check live the Air Quality Index and the sound pollution of a region, have been proposed.

The recommended technology comprises of two modules namely, the Air Quality Index Monitoring Module, the Sound Intensity Detection Module. Firstly, the Air Quality Index is measured considering the presence of the air pollutants. Then the sound intensity is detected using respective sensors. System uses air sensors to sense presence of harmful gases/compounds in the air and constantly transmit this data to microcontroller. Also system keeps measuring sound level and reports it to the online server over IOT. The sensors interact with microcontroller which processes this data and transmits it over internet. This allows authorities to monitor air pollution in different areas and take action against it. Also authorities can keep a watch on the noise pollution near schools, hospitals and no honking areas, and if system detects air quality and noise issues it alerts authorities so they can take measures to control the issue.

Contents

Chapter 1

INTRODUCTION

The main objective of IOT Air Sound Monitoring System is that the Air and sound pollution is a growing issue these days. It is necessary to monitor air quality and keep it under control for a better future and healthy living for all. Here we propose an air quality as well as sound pollution monitoring system that allows us to monitor and check live air quality as well as sound pollution in an area through IOT. System uses air sensors to sense presence of harmful gases/compounds in the air and constantly transmit this data. Also, system keeps measuring sound level and reports it. The sensors interact with raspberry pi which processes this data and transmits it over the application. This allows authorities to monitor air pollution in different areas and act against it. Also, authorities can keep a watch on the noise pollution near schools, hospitals and no honking areas, and if system detects air quality and noise issues it alerts authorities so they can take measures to control the issue.

Some future consumer applications envisioned for IoT sound like science fiction, but some of the more practical and realistic sounding possibilities for the technology include: Receiving warnings on your phone or wearable device when IoT networks detect some physical danger is detected nearby. Self-parking automobiles. Automatic ordering of groceries and other home. Automatic tracking of exercise habits and other day-to-day personal activity including goal tracking and regular progress reports.

The proposed project is an electronic device with smart features like IOT interfacing, smart measuring of goods, RFID tag scanner etc.., with the record of all the information regarding the goods brought and delivered to the consumers. Basic working of the device is that, it scans the RFID tag which act as smart ration card, knows the requirements of the consumer, gets the acknowledgment or confirmation regarding the purchase, measures the needed good accurately and delivers it to the consumer in a smart way.

Network Devices and the IoT All kinds of ordinary household can be modified to working an IoT system. Wi-Fi network adapters, motion sensors, cameras, microphones and other instrumentation can be embedded in these devices to enable them for work in the Internet of Things. Home automation systems already implement primitive versions of this concept for things like light bulbs, plus other devices like wireless scales and wireless blood pressure monitors that each represent early examples of IoT gadgets.

Chapter 2

PROOF OF CONCEPT

The main aim of this paper was to design and implement an efficient monitoring system through which the required parameters are monitored remotely using the internet and the data gathered from the sensors are stored in the cloud and to project the estimated trend on the web browser.

The goal of building a smart city is to improve quality of life by using technology to improve the efficiency of services and meet residents' needs. Information and Communication Technology allows city officials to interact directly with the public to tell what is happening in the city, how the city is evolving, and how to enable a better quality of life. A Smart City is one with at least one initiative addressing one or more of the following six characteristics: Smart Governance, Smart People, Smart Living, Smart Mobility, Smart Economy and Smart Environment. In this system, an application was developed that is going to bear a hand in this campaign. An area that is being surveyed for estimating how much the area is affected by pollution. Then the people are evacuated to a safe place. The description about the integrated network architecture and the interconnecting mechanisms for the reliable measurement of parameters by smart sensors and transmission of data via internet is being presented. The longitudinal learning system could provide a self-control mechanism for better operation of the devices in monitoring stage. The framework of the monitoring system was based on a combination of pervasive distributed sensing units, information system for data aggregation, and reasoning and context awareness.

2.1 Objectives

The air quality and noise monitoring program design dependent upon the monitoring specific objectives specified for the noise and air quality management in the selected area of interest. Defining the output influence, the design of the network and optimize the resources used for monitoring. It also ensures that the network is specially designed to optimize the information on the problems at hand. There might be different objectives for the development of the environmental monitoring and surveillance system. Normally, the system has to provide on-line data and information transfer with a direct /automatically/ on-line quality control of the collected data. Several monitors, sensors and data collection systems to be applied to make on-line data handover and control likely. The main objectives stated for the development of an air quality measurement and surveillance program might be to facilitate the background concentration(s) measurements, monitor current levels as a baseline for assessment, check the air quality relative to standards or limit values, detect the importance of individual sources, enable comparison of the air quality data from different areas and countries, collect data for the air quality management, traffic and land-use planning purposes, observe trends (related to emissions), develop abatement strategies, determine the exposure and assess the effects of air pollution on health, vegetation or building materials, inform the public about the air and noise quality and raise the awareness, develop warning systems for the prevention of undesired air pollution episodes, facilitate the source apportionment and identification, supply data for research investigations, develop/validate management tools (such as models), develop and test analytical instruments and to support legislation in relation to the air quality limit values and guidelines.

Chapter 3

IMPLEMENTATION

The features and each of these sensor devices are operated and controlled based on their sensitivity as well as the range of sensing. The necessary sensing and controlling actions will be taken depending upon the conditions, like fixing the threshold value, periodicity of sensing, messages (alarm or buzzer or LED) etc. Based on the data analysis performed in between and from previous experiences the parameter threshold values during critical situations or normal working conditions are determined. Which specify the condition the data is representing which parameter. In the proposed model deals with the intelligent environment. Which means it will identify the variations in the sensor data and fix the threshold value depending on the identified level of CO or noise levels. In this tier, sensed data will be processed, stored in the cloud i.e.in to the Google spread sheets and it will show a trend of the sensed parameters with respect to the specified values. The end users can browse the data using mobile phones, PCs etc. These steps are:

- Deciding system specifications
- Selection of system components
- Design of circuit diagram
- Design of enclosure or structure

3.1 ARCHITECTURE

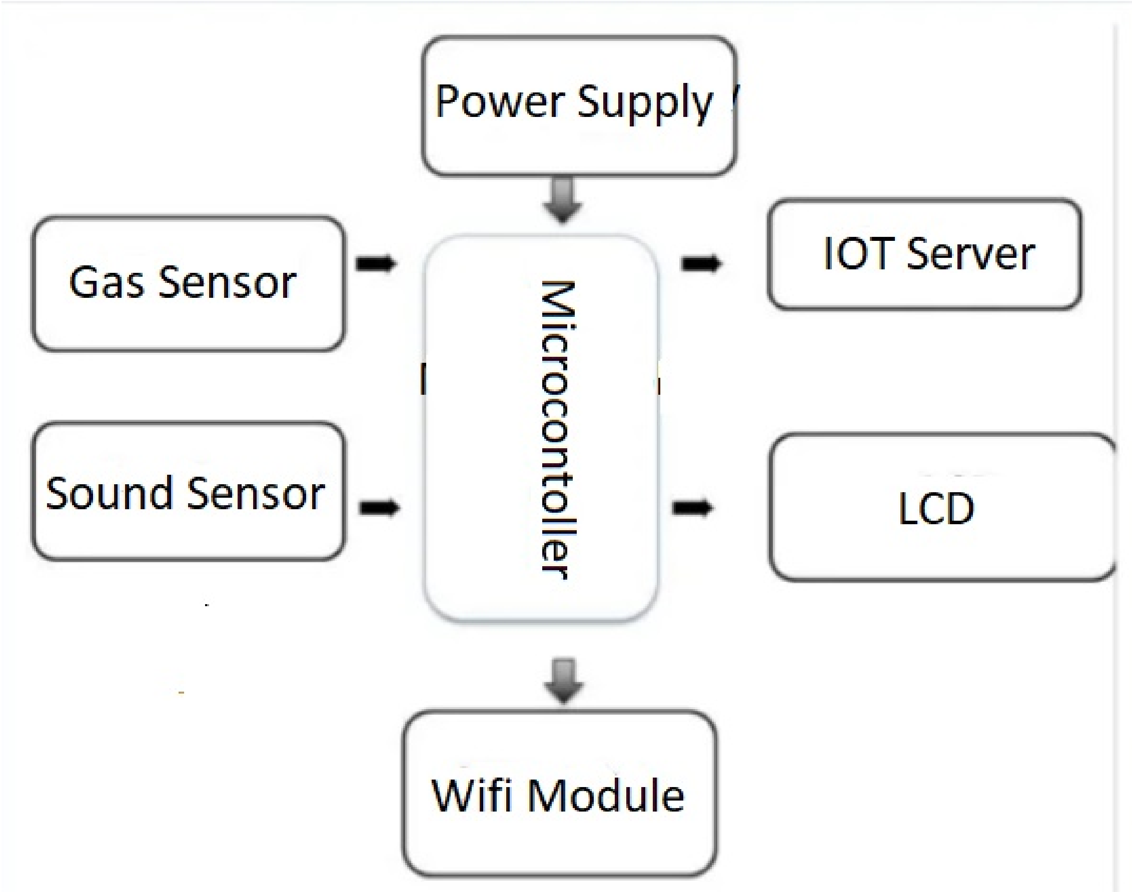

Figure 3.1: Block Diagram of Proposed System

Here we propose an air quality as well as sound pollution monitoring system that allows us to monitor and check live air quality as well as sound pollution in an area through IOT. System uses air sensors to sense presence of harmful gases/compounds in the air and constantly transmit this data. Also, system keeps measuring sound level and reports it. The sensors interact with raspberry pi which processes this data and transmits it over the application. This allows authorities to monitor air pollution in different areas and act against it. Also, authorities can keep a watch on the noise pollution near schools, hospitals and no honking areas. Network Devices and the Internet of Things All kinds of ordinary household gadgets can be modified to work in an IoT system. cameras, microphones and other instrumentation can be embedded in these devices to enable them for in the IOT

3.2 HARDWARE REQUIREMENTS

3.2.1 Arduino Uno

Figure 3.2: Arduino UNO

Arduino Uno - ATMEGA328P

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board. Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package. This Arduino Uno is an original microcontroller board from Arduino officials based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a

USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

Arduino Uno comes with USB interface i.e. USB port is added on the board to develop serial communication with the computer. IDE is equally compatible with Windows, MAC or Linux Systems, however, Windows is preferable to use. Programming languages like C and C++ are used in IDE. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. The ATmega328 on the board comes preprogrammed with a boot loader that allows uploading new code to it without the use of an external hardware programmer. Atmega328 microcontroller is placed on the board that comes with a number of features like timers, counters, interrupts, PWM, CPU, I/O pins and based on a 16MHz clock that helps in producing more frequency and number of instructions per cycle.

TECHNICAL SPECIFICATIONS

Microcontroller : Microchip ATmega328P

Operating Voltage : 5 Volts

Input Voltage : 7 to 20 Volts

Digital I/O Pins : 14 (of which 6 can provide PWM output)

UART: 1, I2C: 1, SPPI : 1

Analog Input Pins : 6

DC Current per I/O Pin : 20 mA

DC Current for 3.3V Pin : 50 mA

Flash Memory : 32 KB of which 0.5 KB used by bootloader

SRAM: 2 KB, EEPROM : 1 KB

Clock Speed : 16 MHz

Length : 68.6 mm

Width : 53.4 mm

Weight : 25 g

3.2.2 NodeMCU

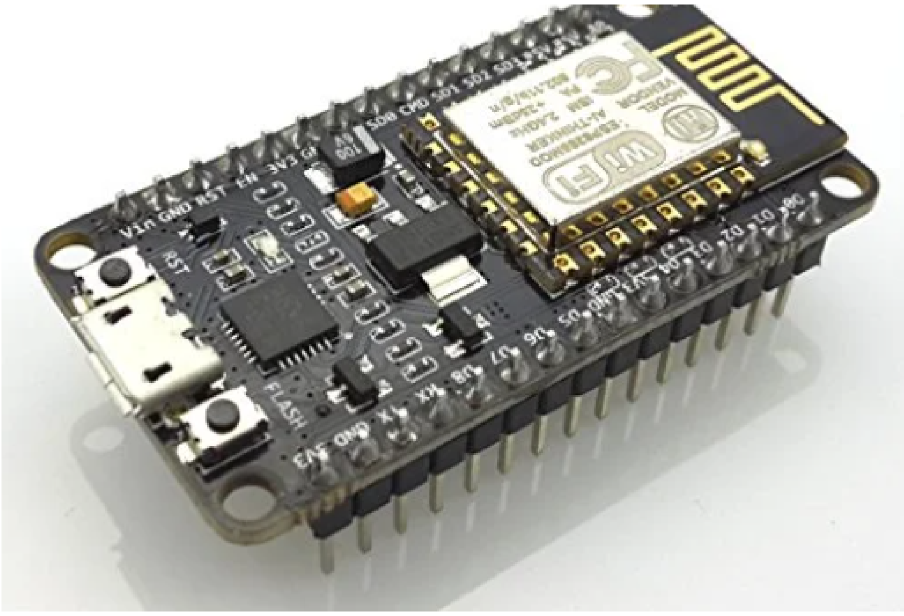

Figure 3.3: NodeMCU

The NodeMCU - ESP8266 has 30 pins in total out of which there are 17 GPIO pins. GPIO stands for General Purpose Input Output. There are the 9 digital pins ranging from D0-D8 and there is only one analog pin A0, which is a 10 bit ADC. The D0 pin can only be used to read or write data and can't perform other options. The ESP8266 chip is enabled when the EN pin is pulled HIGH. When pulled LOW the chip works at minimum power. The board has a 2.4 GHz antenna for a long-range of network and the CP2102 is the USB to TTL converter. The development board equips the ESP-12E module containing ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS.

There's also 128 KB RAM and 4MB of Flash memory (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays. The ESP8266 Integrates 802.11b/g/n HT40 Wi-Fi transceiver, so it can not only connect to a WiFi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. This makes the ESP8266 NodeMCU even more versatile.

the operating voltage range of ESP8266 is 3V to 3.6V, the board comes with an LDO (low dropout) voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA. It has three 3v3 pins along with 4 GND pins. The power supply is via the onboard MicroB USB connector. Alternatively, if you have a regulated 5V voltage source, the VIN pin is used to directly supply the ESP8266. Moreover, it requires 80mA Operating Current and 20 µA during Sleep Mode.

TECHNICAL SPECIFICATIONS

Micro-controller: Tensilica 32-bit RISC CPU Xtensa LX106

Operating Voltage: 3.3V

Input Voltage: 7-12V

Digital I/O Pins (DIO): 16

Analog Input Pins (ADC): 1

UARTs: 1

SPIs: 1

I2Cs: 1

Flash Memory: 4 MB

SRAM: 64 KB

Clock Speed: 80 MHz

USB-TTL based on CP2102 is included onboard, Enabling Plug n Play

PCB Antenna

3.2.3 MQ135 GAS SENSOR

Figure 3.4: MQ135 GAS SENSOR

The MQ-135 Gas sensor can detect gases like Ammonia (NH3), sulfur (S), Benzene(C6H6), CO2, and other harmful gases and smoke. Similar to other MQ series gas sensor, this sensor also has a digital and analog output pin. When the level of these gases go beyond a threshold limit in the air the digital pin goes high. This threshold value can be set by using the on-board potentiometer. The analog output pin, outputs an analog voltage which can be used to approximate the level of these gases in the atmosphere

The MQ135 air quality sensor module operates at 5V and consumes around 150mA. It requires some pre-heating before it could actually give accurate results.

3.2.4 LM393 SOUND SENSOR

Figure 3.5: LM393 SOUND SENSOR

LM393 Sound Detection Sensor Module for Arduino detects whether the sound has exceeded a threshold value. The sound is detected via a microphone and fed into an LM393 op-amp

Measuring the intensity of sound (Note: only detect the existence of sound according to the principle of vibration); Adjustable sensitivity (you can use the blue potentiometer shown in the picture); Working Voltage: 3.3V 5V; Output Mode: Digital Switch Output (low level under woking mode).

3.2.5 OLED DISPLAY

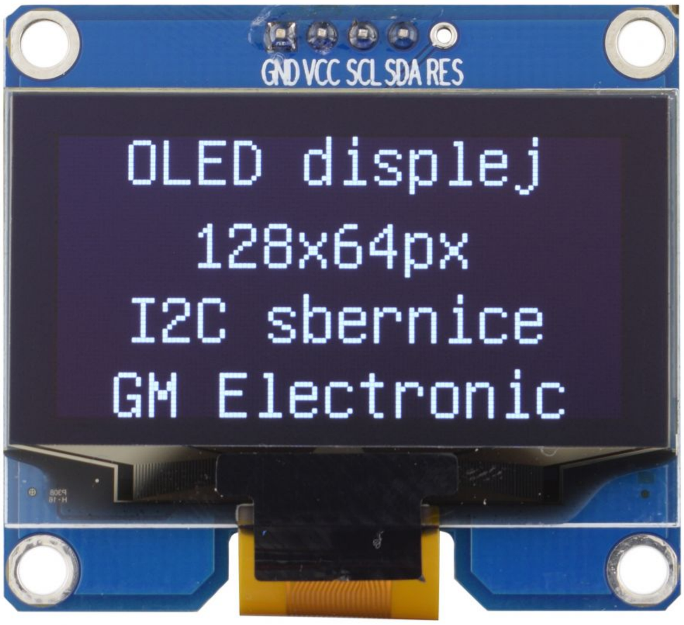

Figure 3.6: OLED DISPLAY

An organic light-emitting diode (OLED or organic LED), also known as organic electroluminescent (organic EL) diode,is a light-emitting diode (LED) in which the emissive electroluminescent layer is a film of organic compound that emits light in response to an electric current. This organic layer is situated between two electrodes; typically, at least one of these electrodes is transparent. OLEDs are used to create digital displays in devices such as television screens, computer monitors, and portable systems such as smartphones and handheld game consoles. A major area of research is the development of white OLED devices for use in solid-state lighting applications.

There are two main families of OLED: those based on small molecules and those employing polymers.An OLED display can be driven with a passive-matrix (PMOLED) control scheme. In the PMOLED scheme, each row (and line) in the display is controlled sequentially, one by one,[6] whereas AMOLED control uses a thin-film transistor (TFT) backplane to directly, allowing for higher resolution and larger display sizes.

3.3 SOFTWARE REQUIREMENTS

3.3.1 Arduino IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards. The source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures.

User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text.

The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port.

3.3.2 Features of Arduino IDE

- Multi-Platform Application Arduino IDE works on the three most popular operating systems: Windows, Mac OS, and Linux. Aside from that, the application is also accessible from the cloud. These options provide programmers with the choice of creating and saving their sketches on the cloud or building their programs locally and upload it directly to the board.

- Board Management Arduino IDE comes with a board management module, where users can select the board they want to work with at the moment. If they wish to change it, they can do so easily from the drop down menu. Modifying their selection also automatically updates the PORT infos with the data they need in relation to the new board.

- Straightforward Sketching With Arduino IDE, users can create programs called sketches that are built with a text editor. The process is a straightforward one though it has several bells

- The tool is armed with a board management module, wherein users can choose which board they want to use. If another board is needed, they can seamlessly select another option from the dropdown menu. PORT data is updated automatically whenever modifications are made on the board or if a new board is chosen.

3.3.3 Android

Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google. It was unveiled in November 2007, with the first commercial Android device, the HTC Dream, being launched in September 2008.

Most versions of Android are proprietary. The core components are taken from the Android Open Source Project (AOSP), which is free and open-source software (FOSS) primarily licensed under the Apache License. When Android is installed on devices, ability to modify the otherwise FOSS software is usually restricted, either by not providing the corresponding source code or preventing reinstallation through technical measures, rendering the installed version proprietary. Most Android devices ship with additional proprietary software pre-installed,[4] most notably Google Mobile Services (GMS)[5] which includes core apps such as Google Chrome, the digital distribution platform Google Play, and associated Google Play Services development platform.

3.3.4 C (Programming language)

C was originally developed at Bell Labs by Dennis Ritchie, between 1972 and 1973. It was created to make utilities running on Unix. Later, it was applied to re-implementing the kernel of the Unix operating system.During the 1980s, C gradually gained popularity. Nowadays, it is one of the most widely used programming languages.

With C compilers from various vendors available for the majority of existing computer architectures and operating systems. C has been standardized by the American National Standards Institute (ANSI) since 1989 (see ANSI C) and subsequently by the International Organization for Standardization (ISO).

C is an imperative procedural language. It was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal run time support. Despite its low-level capabilities, the language was designed to encourage cross-platform programming.

A standards compliant C program that is written with portability in mind can be compiled for a wide variety of computer platforms and operating systems with few changes to its source code; the language has become available on various platforms, from embedded micro-controllers to supercomputers.

3.4 MODULES

The project is divided into five modules.

1.Sound sensing module

2.Air sensing module

3.Application app

3.4.1 Sound sensing module

In this module eeps measuring sound level and reports it. The sensors interact with microcontroller which processes this data and transmits it over the application.
Microcontroller : NODE MCU
LCD Display

3.4.2 Air sensing module

IOT. System uses air sensors to sense presence of harmful gases/compounds in the air and constantly transmit this data Microcontroller : NODE MCU
LCD Display

3.4.3 IoT based application

By the help of Android App create an UI for the application. Noise and Air pollution monitoring system is work while connect with Nodemcu Wifi module.Via this application and also able to see the current status of the noise and air in the atmosphere.
NodeMCU

Chapter 4

RESULT ANALYSIS

The air and sound pollution monitoring system monitors air and noise pollution using a mobile application. It shows the digital value of air and sound pollution and user can analyse it with a graph. It becomes very easy for us to rectify the levels and air and noise pollution around and plan for a healthy living and surrounding. The figures that are included in our paper shows the way the system works and how the output is obtained from the input after processing

4.1 Testing

Hardware testing is the process of evaluating and verifying that a hardware product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

4.1.1 Test Cases

Testing is based on test cases. It describes which feature or service test attempts to cover. In test cases specify what you are testing and which particular feature it tests.

- Test the normal use of system
- Test the abnormal, but reasonable use of system
- Test the abnormal and reasonable use of system

- Test the boundary Conditions

4.1.2 Functionality Testing

Can be performed on hardware or software products to verify that your product functions exactly as designed. The general purpose of hardware and software functionality testing is to verify if the product performs as expected and documented, typically in technical or functional specifications. Developers creating a new product start from a functional specification, which describes the product's capabilities and limitations.

The prime objective of Functionality testing is checking the functionalities of the software system. It mainly concentrates on Mainline functions, Testing the main functions of an application. Basic Usability, It involves basic usability testing of the system. It checks whether a user can freely navigate through the screens without any difficulties. Accessibility, Checks the accessibility of the system for the user. Error Conditions, Usage of testing techniques to check for error conditions. It checks whether suitable error messages are displayed.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

The Automatic Air Sound management system is a step forward to contribute a solution to the biggest threat. The air sound monitoring system overcomes the problem of the highly-polluted areas which is a major issue. It supports the new technology and effectively supports the healthy life concept. This system has features for the people to monitor the amount of pollution on their mobile phones using the application

So, it becomes very reliable and efficient for the Municipal officials along with the Civilians to monitor environment. Letting civilians also involved in this process adds an extra value to it. As civilians are now equally aware and curious about their environment, this concept of IOT is beneficial for the welfare of the society. And it is implemented using the latest technology.

5.2 Future Scope

The future scope is that device which we are having can be done in a compact way by reducing the size of the device for further implementation or the modifications which can be is that detecting the vehicles amount of pollution which can be determined. In future the range can be made increased according to the bandwidth for the high range frequencies. Further research can be made by making the people in the right direction for their welfare. Therefore, there is another beneficiary by using this device in an app so the all can be used in an GSM mobile phones for their daily updates by increasing their range.

APPENDIX

A.1 Coding

```
#define SOUND_SENSOR_PIN A0  // VP pin
const int sampleWindow = 2000; //2 Sec
unsigned int sample;
int db;

#define SMOKE_SENSOR_PIN A3 // VN pin
int Smoke;

int Smoke_limit=65;
int Sound_limit=93;

#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define OLED_RESET     -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RES

#include <WiFi.h>
#include <HTTPClient.h>
```

```
const char* ssid     = "POCO M2 Pro";
const char* password = "alaina123";
String apiKeyValue = "tPmAT5Ab3j7F9";

String sensorName = "Noise&Air Quality";
String sensorLocation = "EKM";

String Notification="";

void setup() {
  Serial.begin(115200);
  pinMode (SOUND_SENSOR_PIN, INPUT);
  pinMode (SMOKE_SENSOR_PIN, INPUT);

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.display();
  delay(500);

WiFi.begin(ssid, password);
  Serial.println("Connecting");

      display.clearDisplay();
     display.setTextSize(1);
     display.setTextColor(WHITE);
     display.setCursor(25,30);
     display.println("Searching for WIFI");
     display.display();
```

```
while(WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}

 display.clearDisplay();
 display.setTextSize(1);
 display.setTextColor(WHITE);
 display.setCursor(0,0);
 display.println("WiFi network with IP:");
 display.setCursor(20,15);
 display.println(WiFi.localIP());
 display.display();
 delay(4000);

Serial.println("");
Serial.print("Connected to WiFi network with IP Address: ");

Serial.println(WiFi.localIP());

}
void loop() {

 Smoke=analogRead(SMOKE_SENSOR_PIN);
  Serial.print("Smoke:");
  Serial.print(Smoke);
  Smoke=map(Smoke,0,4096,0,100);

  unsigned long startMillis = millis(); // Start of sample window
```

```
  float peakToPeak = 0;  // peak-to-peak level
  unsigned int signalMax = 250;  //minimum value
  unsigned int signalMin = 4096;  //maximum value

  // collect data
  while (millis() - startMillis < sampleWindow)
  {
    sample = analogRead(SOUND_SENSOR_PIN);  //get reading from microp

            if (sample < 4096)
              {
               if (sample > signalMax)
                  {
                   signalMax = sample;  // save just the max levels
                  }
               else if (sample < signalMin)
                  {
                   signalMin = sample;  // save just the min levels
                  }

              }
}

  peakToPeak = signalMax - signalMin;  // max - min = peak-peak ampli
  Serial.print("     peakToPeak:");
  Serial.print(peakToPeak);

  db = map(peakToPeak, 200, 3000, 0, 100);  //calibrate for deciBels
```

```
  Serial.print("    Loudness:");
  Serial.println(db);

Display();
Sent_Data();
}


void Display(){

     display.clearDisplay();
     display.setTextSize(1);
     display.setTextColor(WHITE);

     display.setCursor(0,0);
     display.print("Sound level:");
     display.print(db);
     display.println(" dB");

     display.setCursor(0,20);
     display.print("AIR Quality: ");
     display.println(Smoke);



     if(db>=Sound_limit){

     display.setCursor(20,35);
     display.print("Noise Pollution");
```

```
    Notification="Noise Pollution";
    }

    if(Smoke>=Smoke_limit){
    display.setCursor(10,48);
    display.print("Bad Air Quality");
    Notification="Bad Air Quality";
    }
    if(db>=Sound_limit && Smoke>=Smoke_limit){
    display.setCursor(20,35);
    display.print("Noise Pollution");
    display.setCursor(10,48);
    display.print("Bad Air Quality");
    Notification="Noise & Air pollution";
    }
    if(db<Sound_limit && Smoke <Smoke_limit){
     Notification="Normal";
    }


    display.display();
    delay(100);
}


void Sent_Data(){

  if(WiFi.status()== WL_CONNECTED ){
```

```
      HTTPClient http;
      http.begin("http://192.168.218.31/sensordata/post-esp-data.php"
      http.addHeader("Content-Type", "application/x-www-form-urlencod
      String httpRequestData = "api_key=" + apiKeyValue + "&sensor=" +
      Serial.print("Upload_Data: ");
      Serial.println(httpRequestData);

      int httpResponseCode = http.POST(httpRequestData);
      Serial.print("Upload_ResponseCode: ");
      Serial.println(httpResponseCode);
      delay(1000);

  if(httpResponseCode==200){
     display.clearDisplay();
     display.setTextSize(1);
     display.setTextColor(WHITE);
     display.setCursor(5,15);
     display.println("Data saved in server");
     display.setCursor(30,35);
     display.println("Thank you");
     display.display();
     delay(3000);
     }
     else if(httpResponseCode<0){
     display.clearDisplay();
     display.setTextSize(1);
     display.setTextColor(WHITE);
     display.setCursor(30,15);
     display.println("Server Error");
```

```
display.setCursor(30,35);
display.println("Try Again");
display.display();
delay(3000);
}

}



  else {
      Serial.println("WiFi Disconnected");

      }


}
```

A.2 Screenshots

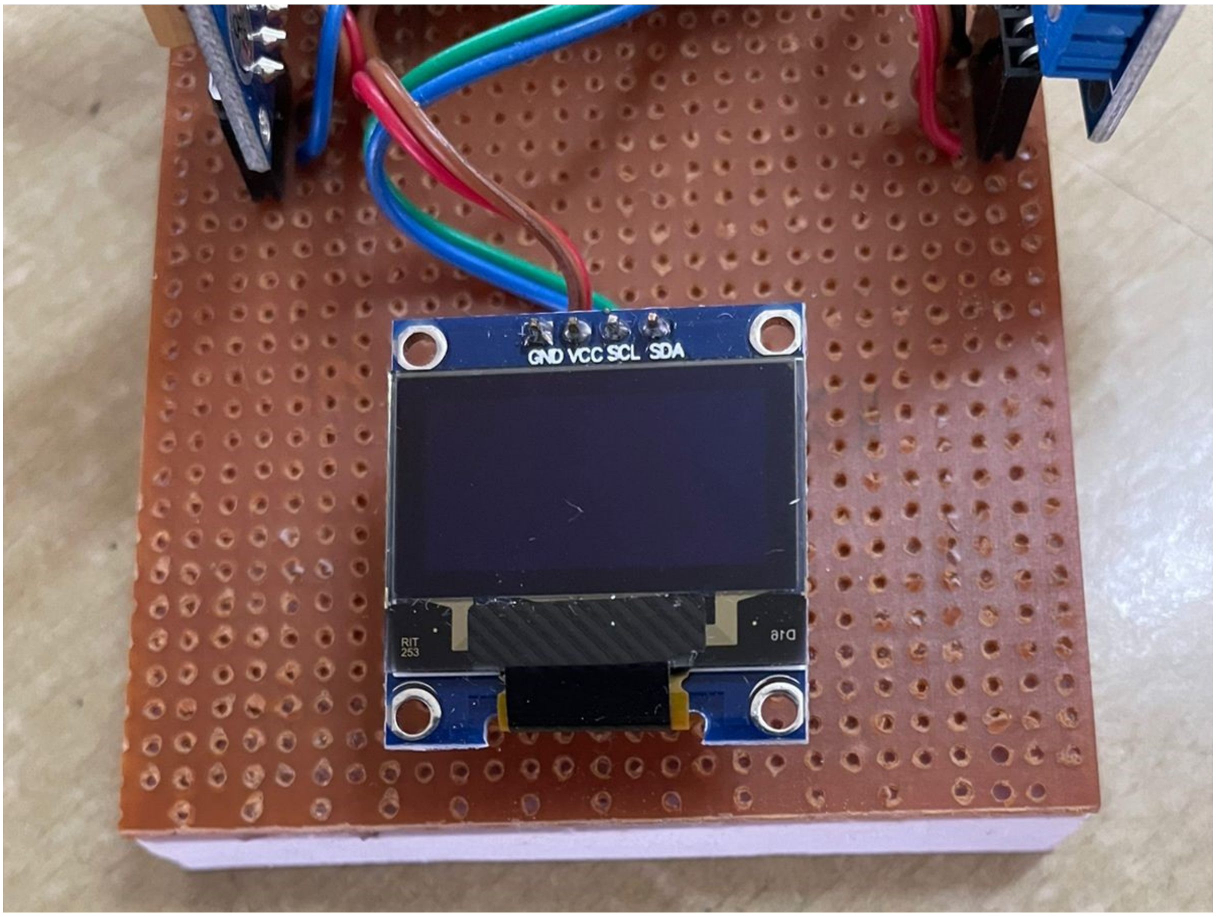

Figure 5.1: OLED Display

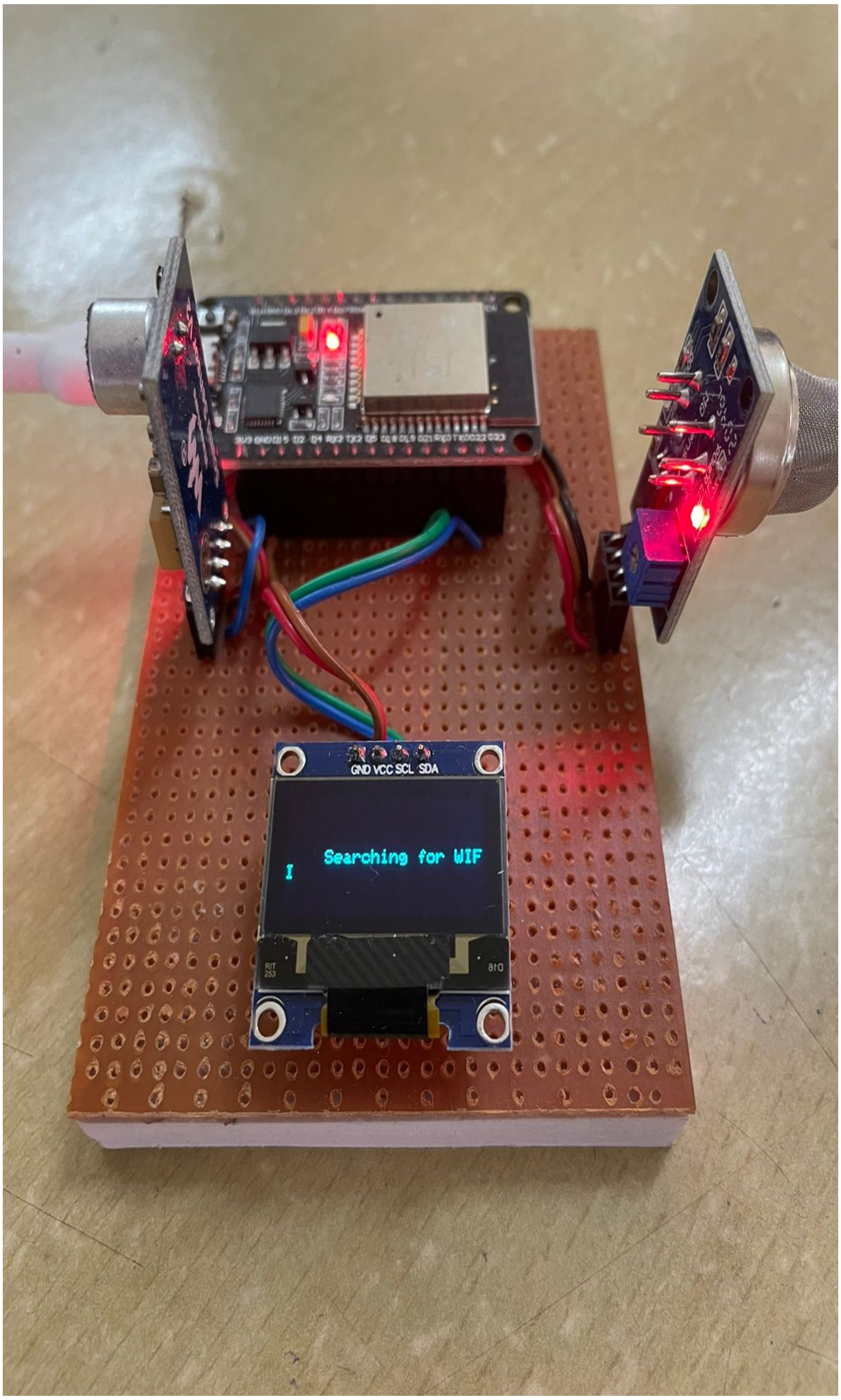

Figure 5.2: OLED searching for Wifi

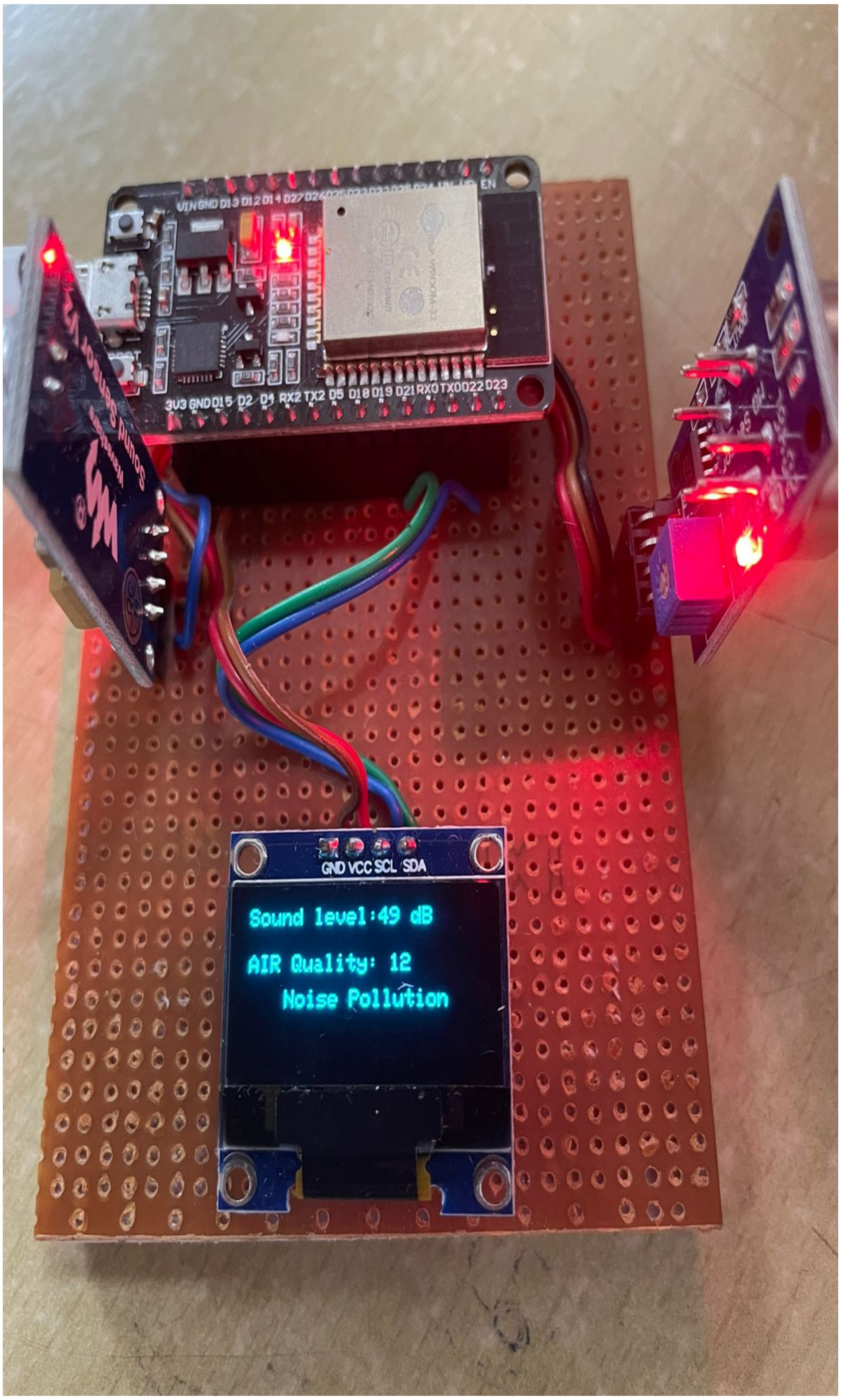

Figure 5.3: OLED displaying when noise or air pollution detected

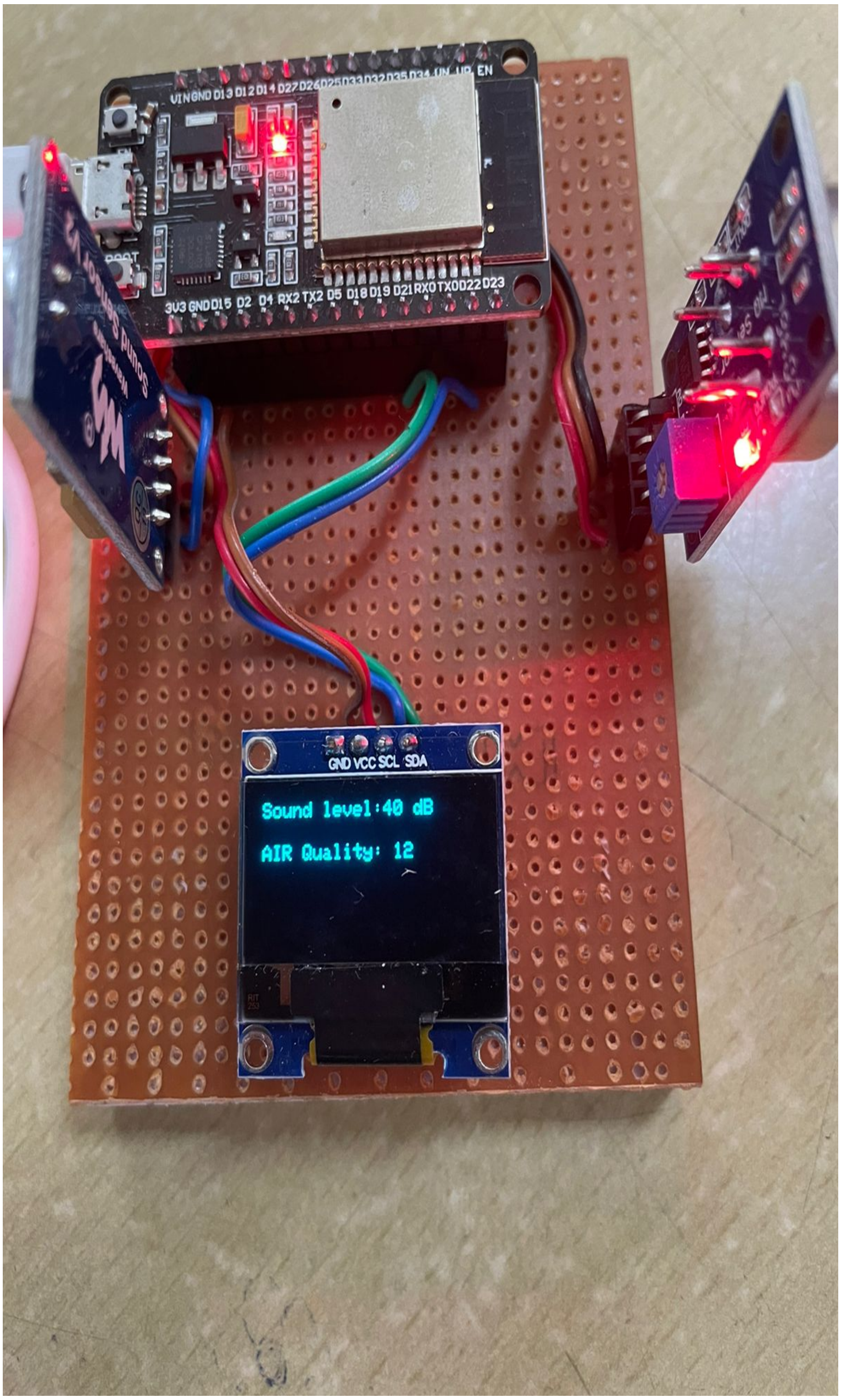

Figure 5.4: OLED displaying the noise and air pollution when its normal

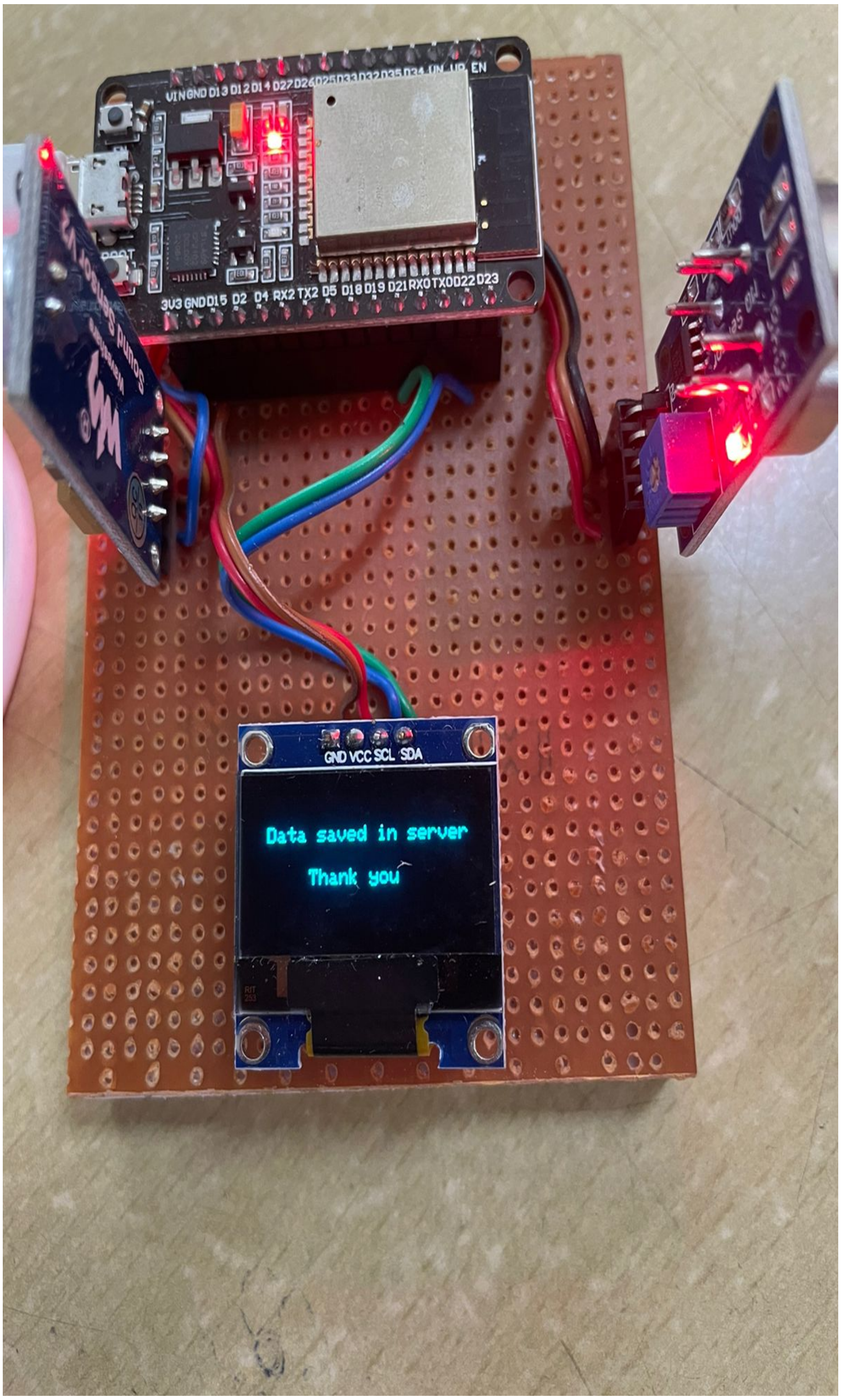

Figure 5.5: OLED displaying when data saved in server

ID	Date $ Time	Sensor	Location	Sound (DB)	Status
1352	2022-08-11 11:41:38	Air&Noise Quality	EKM	11	Normal
1351	2022-08-11 11:41:31	Air&Noise Quality	EKM	11	Normal
1350	2022-08-11 11:41:24	Air&Noise Quality	EKM	11	Normal
1349	2022-08-11 11:41:17	Air&Noise Quality	EKM	11	Normal
1348	2022-08-11 11:41:11	Air&Noise Quality	EKM	11	Normal
1347	2022-08-11 11:41:04	Air&Noise Quality	EKM	11	Normal
1346	2022-08-11 11:40:57	Air&Noise Quality	EKM	11	Normal
1345	2022-08-11 11:40:51	Air&Noise Quality	EKM	11	Normal
1344	2022-08-11 11:40:44	Air&Noise Quality	EKM	11	Normal
1343	2022-08-11 11:40:37	Air&Noise Quality	EKM	11	Normal

Figure 5.6: Sound data displayed in application

SENSOR DATA -AIR

ID	Date $ Time	Sensor	Location	Air (ppm)	Status
1351	2022-08-11 11:41:31	Air&Noise Quality	EKM	86	Normal
1350	2022-08-11 11:41:24	Air&Noise Quality	EKM	87	Normal
1349	2022-08-11 11:41:17	Air&Noise Quality	EKM	86	Normal
1348	2022-08-11 11:41:11	Air&Noise Quality	EKM	86	Normal
1347	2022-08-11 11:41:04	Air&Noise Quality	EKM	72	Normal
1346	2022-08-11 11:40:57	Air&Noise Quality	EKM	87	Normal
1345	2022-08-11 11:40:51	Air&Noise Quality	EKM	60	Normal
1344	2022-08-11 11:40:44	Air&Noise Quality	EKM	87	Normal
1343	2022-08-11 11:40:37	Air&Noise Quality	EKM	87	Normal
1342	2022-08-11 11:40:30	Air&Noise Quality	EKM	69	Normal

Figure 5.7: Air data displayed in application

REFERENCES

- [1]Navreetinder Kaur, Rita Mahajan, Deepak Bagai,“Air Quality Monitoring System based on Arduino Microcontroller,” International Journal Innovative Research in Science, Engineering and Technology (IJIRSET),Vol 5, Issue 6- June 2016.
- [2]Shadrach Tunde,Akinkaude, kowawole, Peter Fasae, “A Survey of Noise Pollution in Ado-Ekiti Metropolis Using Mobile Phone,” Science Technology Department, Science Research Publishing, October-2015.
- [3] Diego Méndez, Alfredo J. Pérez, Miguel A. Labrador,Juan José Marrón, ”P-Sense: A participatory sensing system for air pollution monitoring and control”(Department of Computer Science and Engineering, University of South Florida, Tampa, USA) Published on IEEE at: 12 May 2011 , DOI: 10.1109/PERCOMW.2011.5766902.
- [4] Jiong Jin, Jayavardhana Gubbi, Slaven Marusic, Marimuthu Palaniswami, ”An Information Framework for Creating a Smart City Through Internet of Things,”Published in: IEEE Internet of Things Journal (Volume: 1, Issue: 2, April 2014)DOI:10.1109/JIOT.2013.2296
- [5] A. R. Al-Ali, “A Mobile GPRS-Sensors Array for Air Pollution Monitoring,” IEEE SENSORS JOURNAL, VOL. 10, NO. 10, OCTOBER 2010.