START BUILDING THE IOT AIR QUALITY MONITORING SYSTEM

Ganesh college of engineering

SUBMITTED BY : C. NANDHINI
T. SWATHI
S.SEVVANTHI
M.ARIVUMATHI

1.Introduction

Nowadays, the existing of air quality detector is already been used but still lack of raising awareness of health risk caused by poor air quality. As an example, the tragedy occurred at Sungai Kim-Kim, Pasir Gudang, due to the chemical dump into the river has affected the health of more than 4,000 people, including students, and forcing the government to close at 111 schools in the district.

The implementation of this device helps in solving this kind of problem by showing the results of air quality level not just on LCD screen but also on a web server. Thus, an air quality monitoring system using IoT has been developed to share the air quality level to the public continuously in order to be aware of the unhealthy environment.

With such system implemented at various locations specifically at industrial area or high population cities, the data can be simultaneously monitored online through the real time information that is sent through the system. Thus, people can take further action to prevent themselves from air pollution that caused by industrial activities as the pollutants are released as by-products of these processes by using any alternative road and not easily expose to the unhealthy environment.

2. Air Quality Monitoring

The primary purpose of an air quality monitoring based on the equipment and systems created is to monitor the air pollution and to distinguish between areas where pollutant levels violate an ambient air quality standard and areas where they do not. As health-based ambient air quality standards are set at level of pollutant concentrations that result in adverse impacts on human health, evidence of levels exceeding an ambient air quality standard in an area requires a public air quality agency to mitigate the corresponding pollutant. This safety equipment become a compulsory in every building especially in industrial area in order to be a aware of the air quality level and avoid the hazardous area.

The existing method, some of the existing instruments which is employ analytical techniques to identify gases for air pollution monitoring are Fourier transform infrared (FTIR) instruments, gas chromatographs and mass spectrometers. These instruments provide fairly accurate and selective gas readings. A gas sensor that is compact, robust with versatile applications and low cost could be an equally effective alternative. Some of the gases monitoring technologies are electrochemical, infrared, catalytic bead, photo ionization and solid-state [3]. One of the large scale sensor networks for monitoring and forecasting is Environment Observation and Forecasting System (EOFS). Air pollution monitoring system based on geo sensor network with control action and adaptive sampling rates proposed in also cannot be vast deployment due to high cost [4].

3. Project Design

The development of this system via XAMPP platform allows the air quality level in parts per million (ppm) data to be stored in an online database, thus allowing the public to continuously monitor the air quality level and avoid themselves to be exposed rapidly to these harmful gases. The developed hardware system consists of the MQ135 gas sensor. The gas sensor is able to sense the present of gases through the chemical reaction when the gases flows close to the sensor. The reading of air quality level appears on an I2C LCD. Figure 1 shows the block diagram of project design. There are two mains part of the design which are hardware design and software design. Figure 2 shows the flowchart of the hardware design and software design.

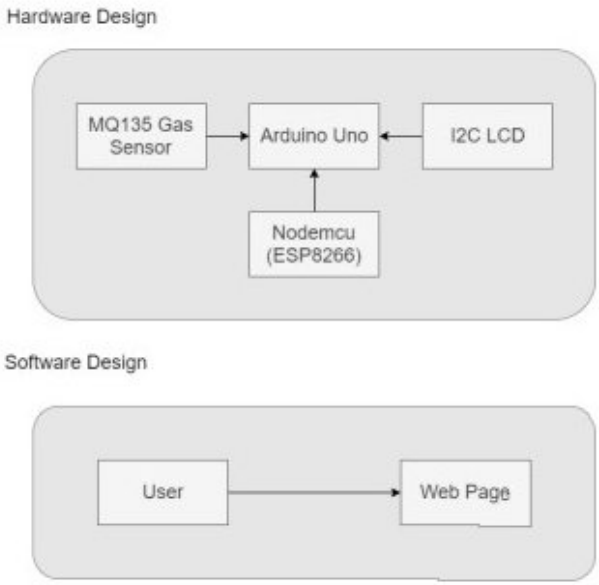


Figure 1: Block diagram of the project design

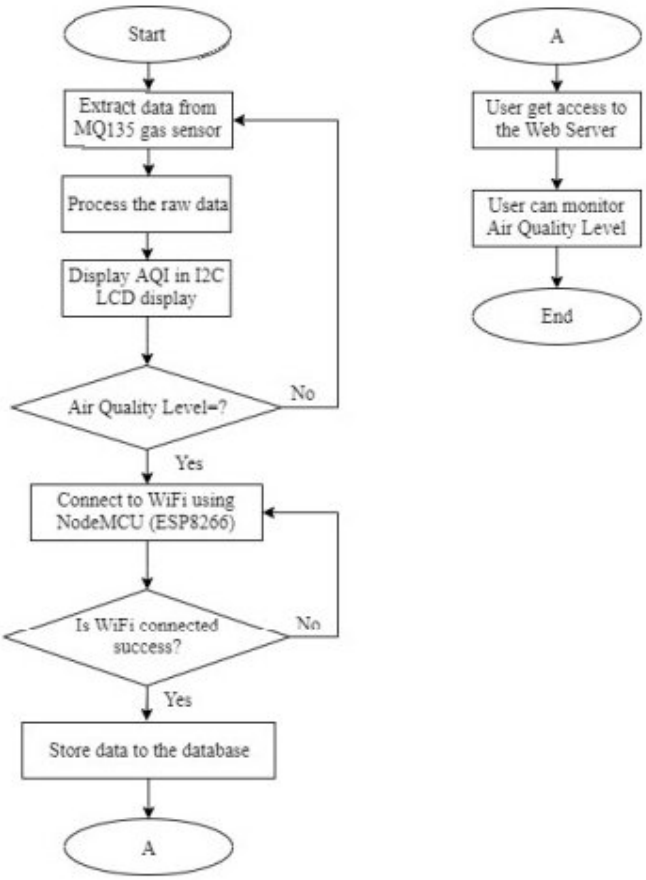


Figure 2: Flowchart of air quality monitoring system

3.1 Hardware Development

A few of suitable components with high performance were used for hardware development. All of the components were at a reasonable cost to integrate to the web server platform of air quality monitoring system. Figure 3 shows the connection of the components for hardware development.

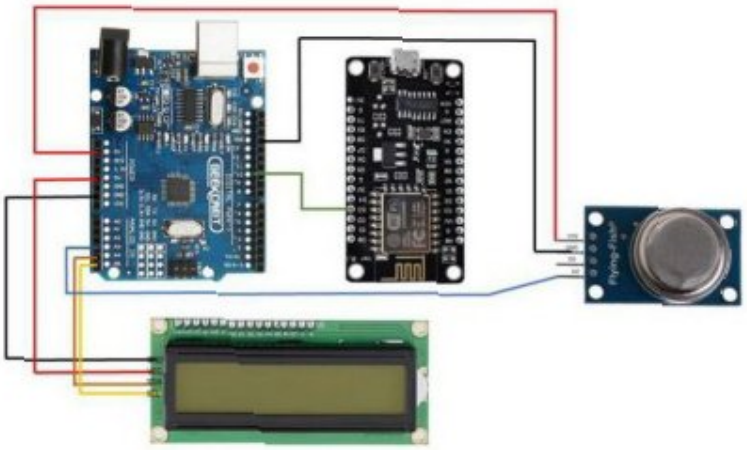


Figure 3: The connection of hardware component

quality level. The GND and VCC pin for Arduino Uno and I2C LCD component are connected to each other. WiFi module, NodeMCU (ESP8266) is connected to the Arduino and act as the bridge between hardware part to the web database.

Arduino Uno has a two ways to supply power in board Arduino Uno through an adapter and USB cable. The adapter has a 12 V that flows in the diode as an output Vin directly to the 5 V regulator. The regulator used to decrease Vin output to the fixed voltage 5 V and the current up to 1 A. The microcontroller Atmega328 are supplied by the voltage 5V and other component such as regulator 3.3 V. Besides, the USB cable has a 5V that flow in the PTC as a dropping currents up to 500mA and other 5V component such as microcontroller and regulator 3.3 V. In addition, it is having an IC comparator to compare the voltage in adapter and USB cable and the priority give to the adapter. However, the USB cable also used to program the Arduino Uno in the Atmega328 that consists of bootloader using a serial communication. The Atmega16U2 in board Arduino Uno must be ensure that the Atmega328 in a bootloader mod before the uploading program using an auto reset circuit. The data from USB cable been change in serial to program the Arduino Uno [5].

The micro controller, NodeMCU (ESP8266) which is used for the project is manufactured by Espressif Systems. The micro controller contains WiFi solution that act as a bridge from Arduino Uno board to the assigned WiFi network. NodeMCU (ESP8266) includes USB connector and a rich assortment of pin-outs.

The LCD screen is an electronic display module that are very commonly used in circuits and devices due to its economical nature and ease of programming as well as its ability in displaying all types of special, custom characters and even animations [6]. The 16x2 refers to its display feature whereby 16 characters can be displayed per line over the 2 lines width of the screen, thus making this LCD a 32-character array. Each character is displayed in a 5x7 pixel matrix, with two working registers incorporated within the LCD, namely Command and Data. The command register stores the instructions given to the LCD to perform a predefined task, whereas the data register stores the ASCII value of the characters to be displayed on the LCD.

The MQ135 sensor has capability to detect some typical dangerous gases. The typical dangerous gases are Ammonia, Alcohol, Benzene, smoke and Carbon dioxide. The MQ135 is a suitable gas sensor to be used in this project. It is attached to Arduino Uno board for capturing the air quality level in parts per million (ppm). The gas sensor translates the gas concentration in the form of voltage level before converting the value into parts per million (ppm). The output conversion from voltage to PPM unit, a library was used for MQ135 sensor by using Arduino microcontroller. In order to calibrate the gas sensor MQ135, a recognized concentration of measured gas is required [7].

3.2 Software Development

The software implemented and programmed for this project include the Arduino Software IDE which allows the Arduino program to be uploaded to the Arduino board, as well as the XAMPP online platform which allows users to monitor data on a web page interface.

The Arduino Software IDE is used for writing of the Arduino programming language or code whereby the program can be uploaded to the Arduino board via the Atmega8U2 chip and USB connection to the computer. This allows the board to perform the functions specified in the program, by which the output data or results of this particular program were displayed in the Serial Monitor feature that is incorporated in the Arduino software. Arduino is an easy tool for fast prototyping which allows the development of projects that are fully-functional at a low cost. Arduino board can apply and adapt to various needs and challenges differentiating its offer from simple 8-bit boards to products for

IoT applications, wearable, 3D printing and embedded environments. Arduino simplifies the process of working with microcontrollers and offers a lot of advantages which include inexpensive, available in cross-platform, allow simple, clear programming environment, open source and extensible software as well as hardware [8].

XAMPP is a free and open-source cross-platform (X) web server software stack package developed by Apache Friends, whereby it consists of the Apache HTTP Server (A), MariaDB database (M), and interpreters that incorporate Hypertext Preprocessor or PHP (P) and Perl Programming Language scripts (P) [9] [10]. The cross-platform feature allows it to run on any computer without the need of an operating system, whereas the MariaDB database is a server developed by the MYSQL team. The PHP server-side scripting language that allows web development to be carried out, and the Perl Programming Language is used for the development of a web application [11]. One of the most attractive features of the XAMPP development tool is ability to allow website interfaces and programming languages to be tested on a local computer without requiring access to the internet [12]. Hence, it allows this website to be initially tested before they are uploaded to a remote web server or computer [11]. An additional tool to this software is a password feature that protects the most crucial parts of the package [13].

The XAMPP software is initially downloaded and installed according to the current Windows specification used by the computer. Once this is done, the Command Prompt (CMD) window can be opened and the internet protocol configuration (ipconfig) command can be entered to determine the Internet Protocol (IP) address used to run XAMPP on the computer [12]. This is illustrated in Figure 4, where the IPv4 address is displayed as 192.168.43.158 in the CMD window, which is used as the initials of the link to the online XAMPP database. At the same time, XAMPP control panel is opened and the two essential ports, namely Apache and MySQL are started up, whilst the 'admin' (administrator) option tool for MySQL is also selected to serve as a workbench that provides a unified visual tool for the database development [12]. This process is depicted in Figure 5.

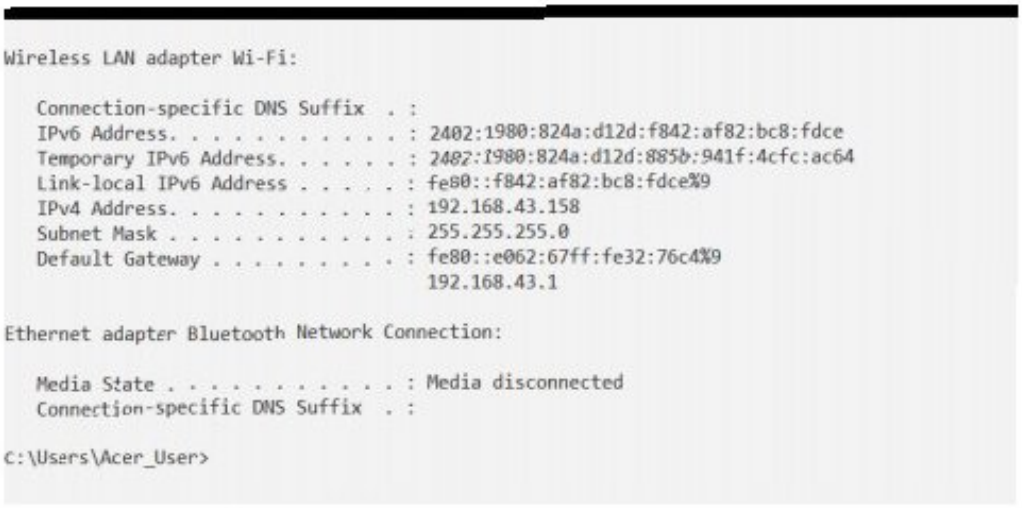


Figure 4: Command Prompt (CMD) windows displaying the IP address

Figure 5: XAMPP control panel windows

Figure 6 shows the main web page of the database displayed online. This database is later used as a platform to store and organize the information of air quality level and time into columns that correlate to the functions of the air quality monitoring system. the database site can be visited by entering the IPv4 address followed by SQLyog, which is the administration tool written in PHP language that is used for the production of these open-sources databases [12].

Figure 6: SQLyog database system

The web page interface is separately developed in the Notepad++ source code editor, which includes the most important home page. Upon completion of both the database and web page, the system is initially tested by adding particular data into the database and subsequently checking if the same information is updated in the web page. If the data transferred is successful, then the system is verified for its functionality. Conversely, an in-depth troubleshooting has to be carried out if the system does not successfully produce the desired response. The entire process of the XAMPP and web page development is depicted in the flowchart provided in Figure 7.

4. Results

The web page interfaces for this system were developed in the Notepad ++ source code editor, with the coding written in Hypertext Preprocessor (PHP) programming language and linked directly to the SQLyog database.

The page allows the user to log into the system in order to monitor air quality level in a real time since the data were sent into the web page for every ten second through the database system as shown in Figure 8.

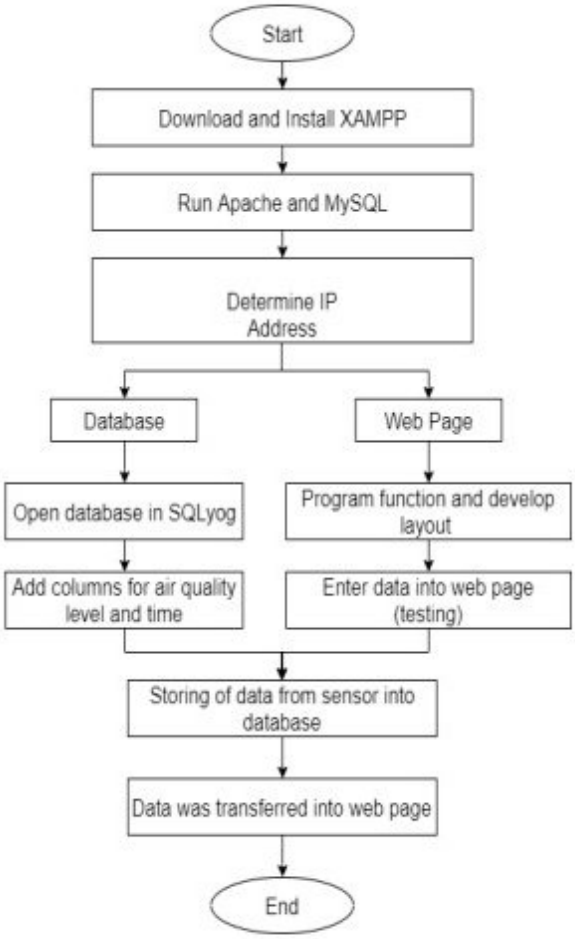


Figure 7: The entire process of XAMPP and web page development

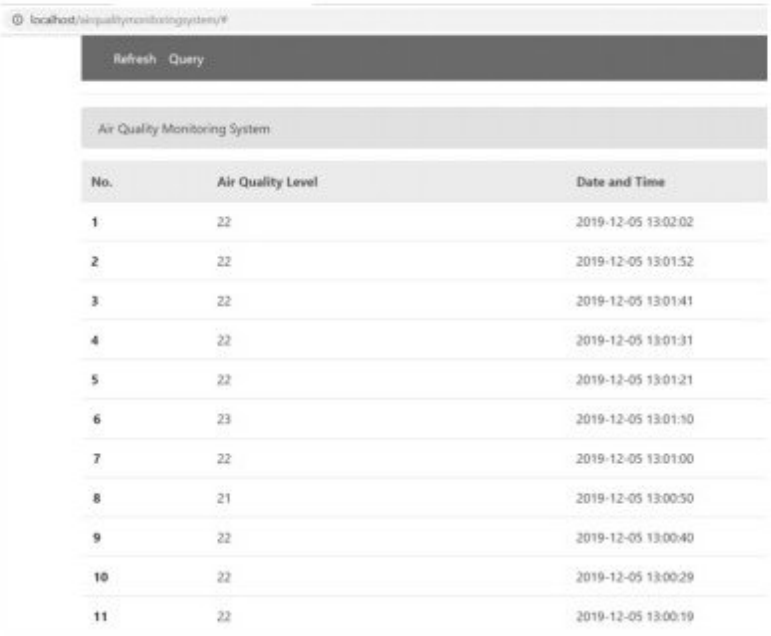


Figure 8: The web page for air quality monitoring system

```
//Program to
#include <SoftwareSerial.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 6, 5, 4, 3);

float t=0;

char data = 0;

// replace with your channel's thingspeak API key
String apiKey = "8NBNB4VQ9F2EEWQM";

// connect 10 to TX of Serial USB
// connect 11 to RX of serial USB
SoftwareSerial ser(10,11); // RX, TX

// this runs once
void setup()
{
  // enable debug serial
  //Serial.begin(9600);
  // enable software serial
  ser.begin(9600);
  lcd.begin(16, 2);

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

```
  lcd.setCursor(0,0);
  lcd.print("Engineers Garage");
  lcd.setCursor(0,1);
  lcd.print("                ");
  delay(3000);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("    IOT AIR");
  lcd.setCursor(0,1);
  lcd.print("QUALITY MONITOR");
  delay(3000);

 // pinMode(12, INPUT);


  // reset ESP8266 WiFi connection AT+CIPMUX=1
AT+CWJAP
  ser.println("AT");
  delay(1000);
  ser.println("AT+GMR");
  delay(1000);
  ser.println("AT+CWMODE=3");
  delay(1000);
  ser.println("AT+RST");
  delay(5000);
  ser.println("AT+CIPMUX=1");
  delay(1000);
```

```
  String
cmd="AT+CWJAP=\"EngineersGarage\",\"egP@$$w0rd?\"";
  ser.println(cmd);
  delay(1000);
  ser.println("AT+CIFSR");
  delay(1000);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("      WIFI");
  lcd.setCursor(0,1);
  lcd.print("   CONNECTED");
 }


// the loop
void loop()
{
  delay(1000);
  t = analogRead(A0);
  Serial.print("Airquality = ");
  Serial.println(t);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("  SENDING DATA");
  lcd.setCursor(0,1);
  lcd.print("    TO CLOUD");
  esp_8266();
```

```
}


void esp_8266()

{

   // TCP connection
AT+CIPSTART=4,"TCP","184.106.153.149",80

    String cmd = "AT+CIPSTART=4,\"TCP\",\"";

    cmd += "184.106.153.149"; // api.thingspeak.com

    cmd += "\",80";

    ser.println(cmd);

    Serial.println(cmd);

    if(ser.find("Error"))

    {

      Serial.println("AT+CIPSTART error");

      return;

    }



  // prepare GET string GET
https://api.thingspeak.com/update?
api_key=LHAG4NSIYJ5UWS6U&field1=0\r\n\r\n

  String getStr = "GET /update?api_key=";

  getStr += apiKey;

  //getStr +="&field1=";

  //getStr +=String(h);

  getStr +="&field1=";

  getStr +=String(t);

  getStr += "\r\n\r\n";
```

```
      Serial.println("AT+CIPSTART error");
      return;
    }

  // prepare GET string GET
https://api.thingspeak.com/update?
api_key=LHAG4NSIYJ5UWS6U&field1=0\r\n\r\n
  String getStr = "GET /update?api_key=";
  getStr += apiKey;
  //getStr +="&field1=";
  //getStr +=String(h);
  getStr +="&field1=";
  getStr +=String(t);
  getStr += "\r\n\r\n";
  // send data length
  cmd = "AT+CIPSEND=4,";
  cmd += String(getStr.length());
  ser.println(cmd);
  Serial.println(cmd);
  delay(1000);
  ser.print(getStr);
  Serial.println(getStr);

  // thingspeak needs 15 sec delay between updates
  delay(16000);
}
[/restrict]
```

In order for the web page to perform its desired functions, the air quality level file for this page has to be programmed to check whether the air quality level and date and time column has been filled. If the column is filled, the GET request method which accepts the details enclosed within the body of the message is used to determine if the air quality level and time matches to the information stored in the database. Figure 9 shows the line chart data from Google Chart which uses Vector Markup Language (VMP) and Scalable Vector Graphic (SVG). It is needed to create encode within the browser in order to show the level of air quality.

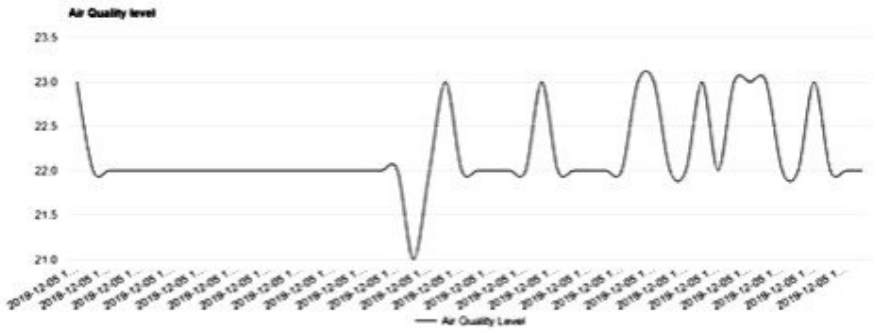


Figure 9: The line chart of air quality monitoring system in normal reading condition

Figure 10, it shows the air quality monitoring system is placed on the table where it should be four to six foot above the ground to record the air quality level in Universiti Tun Hussein Onn Malaysia, Parit Raja. If the air quality level much bigger than the threshold level which is 100, the I2C LCD displays "Unhealthy" on the screen but when the reading is under threshold level then it displays as "Normal".

Figure 10: End product of air quality monitoring system

The air quality monitoring system with IoT have been tested in Universiti Tun Hussein Onn Malaysia (UTHM) for outdoor location. The experiment was done in one day to determine the actual value for normal condition without any existence of harmful gases which means in a safe environment. The normal condition level usually around 15 to 50 ppm which depends on the MQ135 sensor sensitivity in clean air factor resistance. The value start from 50 to 80 which can be defined as lower air pollution presence. Figure 11 shows that the ppm value senses by the MQ135 sensor was in normal condition at UTHM area near Pusat Kesihatan Universiti (PKU). Therefore, the threshold level has been selected setup for 100 parts per million (ppm). In additional, the value of 100 ppm and above can cause to sore eyes, cough and hard breathing. The experiment was scheduled in the morning where the reading slightly increased when there have a few cars passed through the area. Figure 12 shows the reading for air quality index at night time. The value is slightly decreased to 18 ppm when the temperature was decrease due to the rainy weather. The record time for the reading is between 10 to 15 minutes.

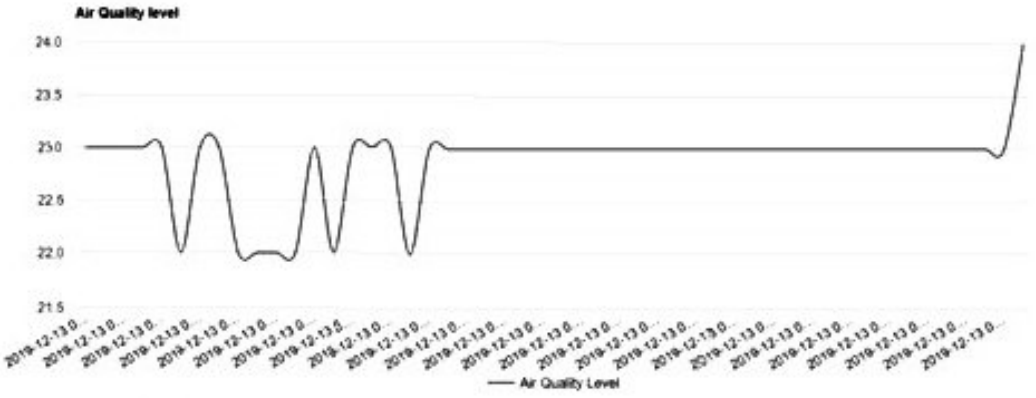


Figure 11: The air quality level in the morning on 13th of December 2019

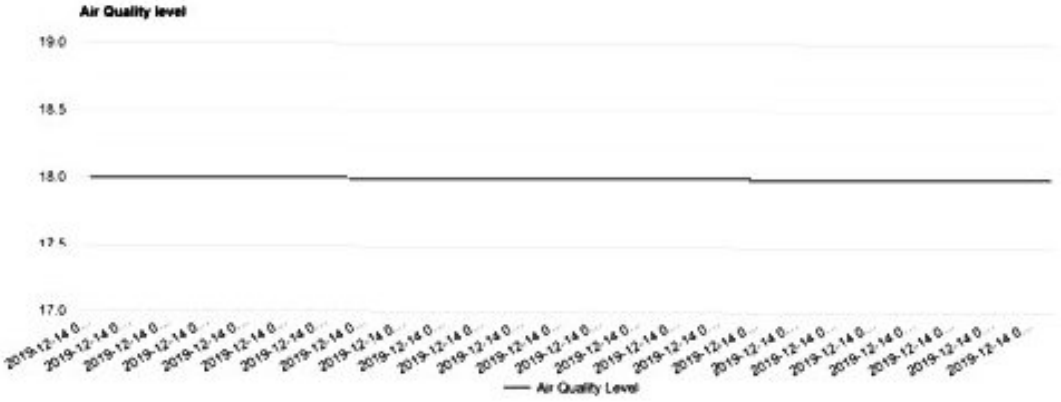


Figure 12: The air quality level in the night on 13th of December 2019

5. Conclusion

As a conclusion, the Air Quality Monitoring System with IoT has achieved the objective that able to detect air pollution or harmful gases by using MQ135 gas sensor. It is also capable of spreading awareness of how important to know the effects of air pollution towards healthy and environment. The following objective which is to develop the monitoring system of air pollution for environmental sensing application using Internet of Things (IoT), has also been fulfilled. The entire system includes the hardware components such as the Arduino Uno microcontroller, MQ135 gas sensor, I2C LCD and NodeMCU (ESP8266) as well as the software components comprising of the Arduino IDE, XAMPP platform and SQLyog for database system. These features are designed in order to work hand-in-hand in order to provide an ideal system for user. The last objective is to verify the function of the system in a different level of air quality level which also been accomplished. The experimental result with a different time and locations in order to get the data collection but for high reading of air quality level which is exceeding to the 100 ppm, couldn't be justified due to the no presence of any harmful gases or air pollution at the location selected. Thus, this ensures that the system is capable of performing the required safety and monitoring purposes.

ESP-01 WiFi module which helps us to connect to the ThingSpeak Platform. The connections between them is mentioned in the connections diagram.

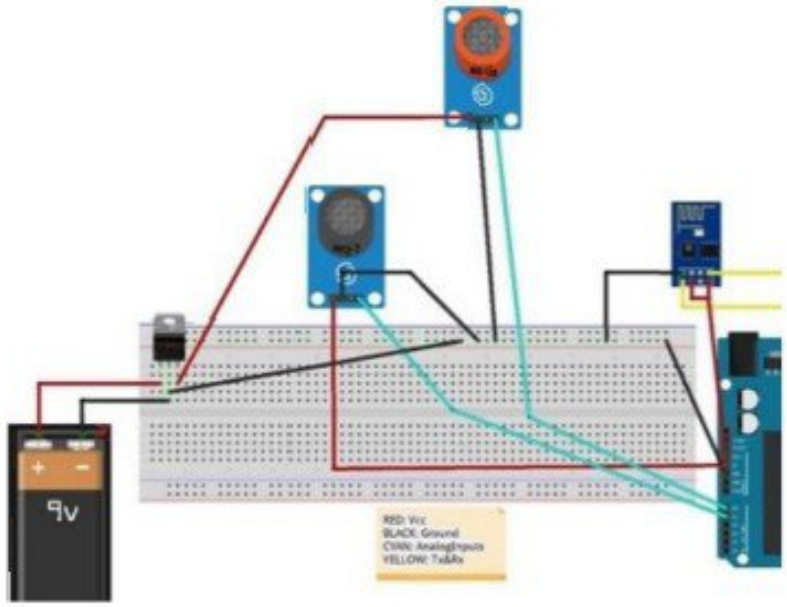


DIAGRAM MADE THROUGH FRITZIG

The most important step is to calibrate the sensor in fresh air and then draw an equation that converts the sensor output voltage value into our convenient units PPM (parts per million). Here are the mathematical calculations derived [6], Fig 5: Internal circuit diagram of MQ135 sensor RS and RL combined From Ohm's Law, at constant temperature, we can derive I as follows:

$$I = V/R \quad (1)$$

From fig , equation 1 is equivalent to

$$I = Vc / Rs + Rl \quad (2)$$

From , we can obtain the output voltage at the load resistor using the value obtained for I and Ohm's Law at constant temperature. V = I*R

$$VRl = [\ Vc/ Rs + Rl\ ] * RL \quad (3)$$

$$VRl = [\ Vc * Rl\ [Rs + Rl\ ] \quad (4)$$

$$(VRl * Rs) + (VRl * Rl) = Vc * RL \quad (5)$$

$$VRl * Rs = (Vc * Rl) - (VRl * Rl) \quad (6)$$

$$Rs = (Vc * Rl) - (VRl * Rl)\ VRl \quad (7)$$

$$Rs = (Vc * Rl)\ VRl - Rl \quad (8)$$

Equation 9 help us to find the internal sensor resistance for fresh air

$$Rs = (Vc * Rl)/\ VRl - Rl \quad (9)$$

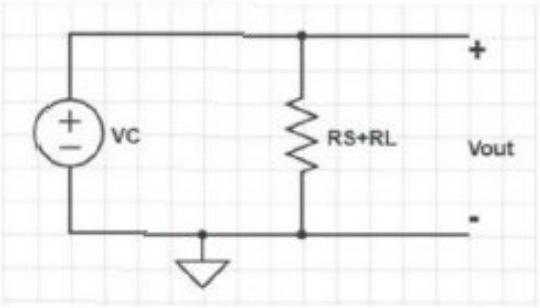


Fig. 3. Internal circuit diagram of MQ135

Equation 10 is depicted from the datasheet mentioned in Fig 6. To calculate R0, we will need to find the value of the Rs in fresh air. This will be done by taking the analog average readings from the sensor and converting it to voltage. Then we will use the Rs formula to find R0. First of all, we will treat the lines as if they were linear. This way we can use one formula that linearly relates the ratio and the concentration. By doing so, we can find the concentration of a gas at any ratio value even outside of the graph's boundaries. The formula we will be using is the equation for a line, but for a log-log scale. The formula for a line is [9]: From above Figure 3, we try to derive the following calculations.

$$y = mx + b \quad (11)$$

For a log-log scale, the formula looks like this:

$$\log10y = m * \log10x + b$$

Now that we have m, we can calculate the y intercept. To do so, we need to choose one point from the graph (once again from the CO2 line). In our case, we chose (5000,0.9)

$$\log(y) = m * \log(x) + b \quad (17)$$

$$b = \log(0.9) - (-0.318) * \log(5000)\ (18)\ b = 1.13 \quad (19)$$

Now that we have m and b, we can find the gas concentration for any ratio with the following formula:

$$\log(x) = \log(y) - b\ m \quad (20)$$

However, in order to get the real value of the gas concentration according to the log-log plot we need to find the inverse log of x: $x = 10 / m$

2.4 Fine particulate matters

The air quality has slowly deteriorated since the Industrial Revolution, and is now further affected by climate change and denser urban vehicle use. In recent years, while Taiwan's smoking population has decreased year by year, the rate of cancers, such as lung cancer, have increased year by year, and poor air quality may be an important factor in this phenomenon, with particle matters having the most serious effects on the human body. Particle matters are referred to as "smog", and can be divided into 4 categories according to the size, including total suspended particulates (TSP), 10μm particle matters (PM10), coarse particulate matters (PM2.5-10), and fine particulate matters (PM2.5), with the details shown in Table 3. Fine particulate matters are mainly discussed in this study.

Table 3 Description of particle matters and particle sizes

Particle size (μm)	Name	Description
<2.5μm	Fine particulate matters (PM2.5)	Entering the bloodstream directly through alveoli
2.5μm~10μm	Coarse particulate matters (PM2.5-10)	Being inhaled by the human respiratory system
<10μm	10μm particle matters (PM10)	Through the nasal cavity to the throat
<100μm	Total suspended particulates (TSP)	Beach sand being suspended in the air

There are 3 main sources of fine particulate matters (PM2.5): natural, primary, and derived. A natural source generally refers to forest fires or volcanic eruptions; primary fine particulate matters are generally produced by incomplete combustion in factories, including many toxic substances, such as organic carbon, dioxin, and heavy metals; derived fine particulate matters refer to oxycarbide and oxysulfide emitted by factories and in petrochemical industries, and oxycarbide and oxysulfide often produce secondary toxic pollutants due to sunlight, such as nitrate and sulfate. Fine particulate matters mainly invade human's blood circulation through alveoli, indicating their tiny sizes, as shown in Figure 2 [6]. After entering the blood, they cause inflammation in organs, such as hearts, blood vessels, brains, and reproductive systems, as shown in Figure 3 [6], which mainly occurs because fine particulate matters are toxic substances containing heavy metals.

PM2.5 = 0.5* mean of the first 12 hours +0.5* mean of the first 4 hours (2-1)

8 valid data are needed in 12 hours

3 valid data are needed in 4 hours

2.5 Wireless transmission

People choose the most suitable methods to transmit data depending on the circumstances, and such transmission modes are divided into wired and wireless transmissions, which have their own advantages and disadvantages, as shown in Table 5. Smart homes show their effects through IoT applications. In smart homes, the transmission distance is basically within the residence, and the data to be transmitted will not be a large amount; hence, without emphasizing transmission rate, wireless transmission was selected in this study. The transmission rate involves several factors, such as indoor environment, sheltered areas, and different floors, meaning the transmission rate at all locations is basically the same on the same floor, provided it is not too heavily sheltered, and while the transmission rate on different floors is more or less affected, the main difference of the transmission rate on different floors is stability. The greater the distance between the floors, the more unstable the transmission rate. The wired network line shall be arranged first when designing smart homes; otherwise, it will be troublesome to modify or build a new wired network. Therefore, if the system is set up on different floors and without a line arrangement, it is suggested that wired transmission is more stable than wireless transmission. However, the disadvantages of wired transmission is that the network cannot be modified or rearranged as easily as those with wireless transmission, thus, wireless networks are gradually replacing wired networks in smart homes.

Table 5 Advantages and disadvantages of transmission modes

Advantages and disadvantages	Wired transmission	Wireless transmission
Advantages	Signals not easily to be disturbed	No wire arrangement
	High security	Easy setup
	Cheap	Easy maintenance
Disadvantages	Transmission rate reduced due to line length	Unstable signals
	Difficult setup	Expensive
	Inconvenient maintenance	Low security

The network architecture of wireless transmission consists of 4 layers: application layer, transport layer, internet layer, and link layer, and is often referred to as the TCP/IP or DoD (Department of Defense) model. With their own responsibilities, all layers are closely related and work together, and the detailed transmission flow chart is shown in Figure 4.

4.1 CO sensor module

This study used MQ-7 as the indoor CO sensor module, as shown in Figure 17. The gas sensitive material used in this sensor is stannic oxide (SnO_2), which is an inorganic compound with low conductivity in general air. Sensor conductivity depends on the CO concentration in the air, where higher concentration leads to higher conductivity. MQ-7 detects CO by the high and low temperature circle detection method, where the voltages for high and low temperatures are 5V and 1.5V, respectively. Low temperature is used to detect CO, and the changes in conductivity can be known with the simple circuit design and be converted into output signals related to the CO concentration; high temperature is used to clean the gases absorbed at a low temperature. MQ-7 is highly sensitive in sensing CO and is a low-cost and suitable sensor for CO detection.

Figure 17 CO sensor module

4.2 CO_2 sensor module

This study used the NDIR infrared sensor module (MH-Z14A) as the CO_2 sensor module, as shown in Figure 18, which mainly senses CO_2 in indoor air using the theory of the non-distributed infrared ray. In addition to long service life, it has internal temperature compensation, digital and analog output, and the sensing range of 0-5000ppm. The detailed specifications are shown in Table 16.

Figure 18 CO_2 sensing module (MH-Z14A)

Table 16 Detailed specifications of CO_2 sensing module

Operating voltage	4-6V
Operating current	Mean 50mA
Detection accuracy	±50ppm

metric, here it is shown which of them eliminates the erroneous components inserted in the signal [34]. Furthermore, it was observed that the average filter is adequate to be implemented by taking n samples with a window of size $k = 25$. Figure 1 shows the signal smoothing obtained by applying the above filters.

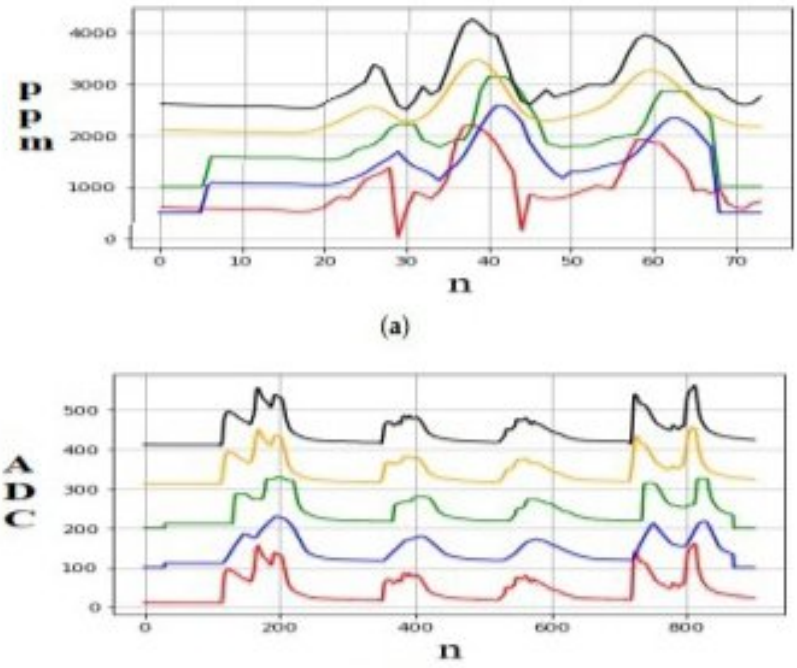


Figure 1. Signal smoothing analysis. (**a**) CO_2 measurement by using the SCD30 sensor. The y-axis is the sensor output in ppm, and the x-axis is the number of samples. (**b**) NO_x measurement by using the MQ-135 sensor. The y-axis is the sensor output given by a 10-bit analog-to-digital (ADC) converter (1024 discrete levels). The x-axis is the number of samples.

3.3. *Voltage Supply and Rain Protection*

The system is placed on the roof of the vehicle and is magnetically fastened. When the key of the vehicle is turned, this action closes the starting circuit, the motor engine starts, and the alternator comes on to supply power to the vehicle and power the system with 12 V, which requires rectification to 5 V. Next, the IoT device turns on and leaves its rain protection case to begin data collection. If the system detects rain or the vehicle turns off, it returns to its initial position inside its case.

4. IoT Architecture

IoT devices have limited computational resources. Therefore, it is necessary to work with lightweight network protocols and services, which are also oriented to the type of information being sent [18]. For this reason, the message sending protocol is the Message Queuing Telemetry Transport (MQTT), because it has a variable light payload messaging service with a publisher/subscriber model [22]. Consequently, Mosquitto (https://mosquitto.org/ accessed on 17 August 2022) is used as the server. Furthermore, following the lightweight computational footprint, the database should not be relational because the information is not concatenated with other information sources. Thus, a time-series database called InfluxDB (https://www.influxdata.com/ accessed on 17 August 2022) is used. The data will be stored for each node and topic (named for each variable in MQTT). In summary, the functionality of each block of the proposed IoT architecture is described as follows:

- **IoT node:** Collect sensor data and send messages via MQTT to the edge server.
- **Edge:** Has the MQTT broker, which receives data from IoT nodes. Then, it stores data in a time series database identifying each node.
- **Cloud:** We use a public data viewer called Grafana (monitoring stack) to securely connect to the database allocated in the edge and make querying from the cloud side.

Figure 2 shows the IoT architecture. Users are considered vehicles and people who can observe and obtain reports.

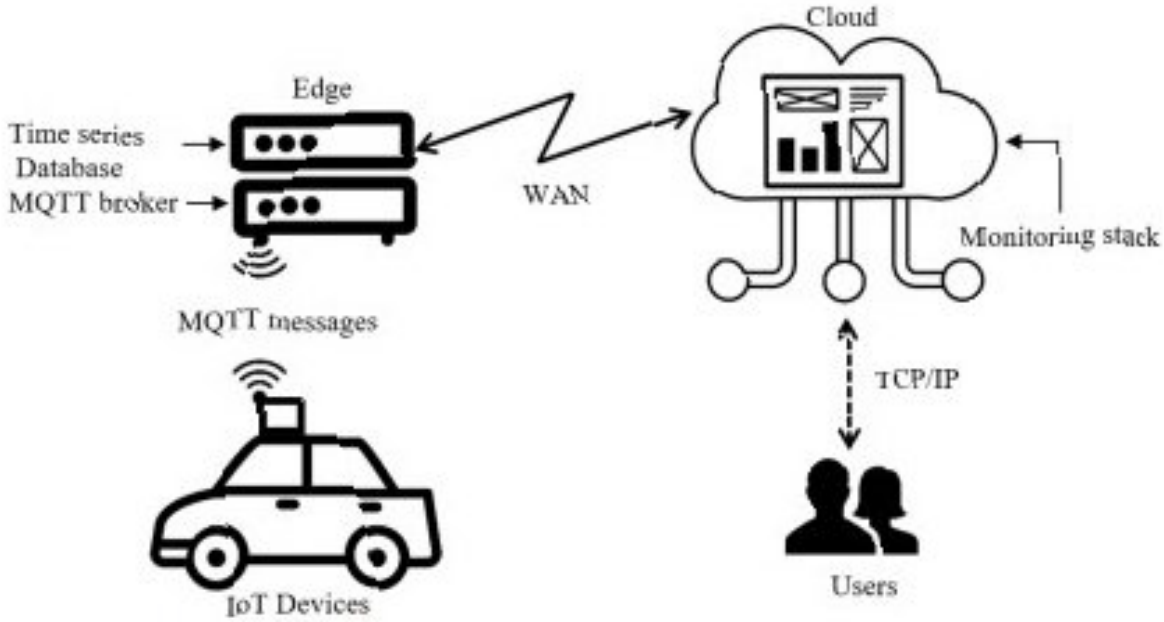


Figure 2. Proposed IoT architecture.

5. Data Analysis

This section presents the data analysis scheme used to locally implement the ML algorithm. It is essential to consider that, in Ibarra, it is known from previous studies that the air quality is acceptable [1,4,10]. However, it shows punctual peaks that generate risk for people.

5.1. Original Data

The data are collected once the IoT device is built with the appropriate sensor calibration. In addition, high-level systems are used to ensure that the data are reliable, such as `Air Quality Station` in conjunction with MaxiMet weather stations GMX-240 from Libelium (https://www.libelium.com/iot-solutions/smart-cities/ accessed on 17 August 2022), and mobile applications that receive information from the different satellites that surround the globe. To describe the air-pollution phenomenon, seven specific data collection schedules (information obtained from government agencies) were established according to vehicle density and the hours of highest traffic flow (8h00, 13h00, and 17h00), normal traffic flow (10h00, 13h00 and 20h00), and reduced traffic flow (2h00). The data acquisition process starts taking 50 samples every two minutes in the above-mentioned schedules, while the vehicle is driven around the city for approximately two months. Additionally, due to the warming-up condition of low-cost sensors, the system waits 5 min to start taking samples. Furthermore, the smoothing algorithms and outlier detection techniques prune incorrect data, improving the quality of the dataset. After that, the data are sent to the InfluxDB database. In this case, the number of samples sent to InfluxDB is equal to 140,000. Moreover, each obtained datum has been classified according to the defined schedule.

At this point, it is important to mention that in this paper, the data set was divided according to the similarity of the values of the different variables into subsets. Therefore, we call classes to these subsets of data, as is carried out in machine learning. For us, a class is a set of data that have the same characteristics. Thus, once the classes were defined, algorithms were trained to generate rules that associate new data with the corresponding

class. That said, in the event that there are values close to two classes, the algorithm makes its decision based on the variable that has the greatest weight or significance. In this way, the data are classified by criteria or functions defined by the algorithm itself.

Taking into account everything said above, the measurements of each variable are now categorized within different air-pollution levels (i.e., classes), which are established by government air-quality measurement networks (e.g., see Quito Metropolitan Network of Atmospheric Monitoring reports (QMNAM) at http://www.quitoambiente.gob.ec/index.php/informes accessed on 17 August 2022). In this paper, following QMNAM reports, the abovementioned classes were defined as follows:

- **Class A**: High levels of pollution with increased incidence of UV rays and high temperatures.
- **Class B**: Acceptable levels of pollution and moderate temperature.
- **Class C**: Low levels of gas concentration and suitable environmental conditions.

It is important to mention that the National Transit Agency of Ecuador imposes the maximum speed in cities at 50 km/h. In addition, the IoT application is focused on collecting data in city zones with high vehicle density, which reduces the chance that the IoT device has inaccurate measures by medium/high vehicle speeds. Furthermore, smoothing algorithms eliminate those errors by comparing samples taken with the same sample rate.

5.2. Outlier Detection

Due to the nonlinearity of the electronic elements and sensor wear, errors appear in the data that may not follow the same distribution and do not have the same trend as the important information. For this reason, they can impair the data acquisition process [35]. Therefore, descriptive unsupervised learning techniques allow elimination of these data in order to find a refined training set. Consequently, the most relevant techniques found in the literature review are as follows: Standard Deviation, Local Outlier Factor, Isolation Forest, Elliptic Envelope, and One-Class SVM [36]. However, it is necessary to know which of the above-mentioned methods suitably fits the data type of the proposed system. Therefore, knowing the statistical distribution of the original samples allows detection of outliers based on quartiles. Figure 3 shows the box plot of the original data set (OD) of the CO variable, and how unsupervised learning techniques prune data that have different distributions. As a result, it can be observed that the anomaly detection algorithms allow the data to be concentrated towards a central tendency while eliminating the distant ones (Gaussian bell). Therefore, it is observed that the Standard Deviation (STD) and Local Outlier Factor (LOCAL) algorithms have similar results. Furthermore, Isolation Forest (ISO) and Elliptic Envelope (ELLIP) algorithms maintain outliers in their corresponding dataset. Finally, One-Class SVM (OSVM) shows the lowest data variability and has no outliers.

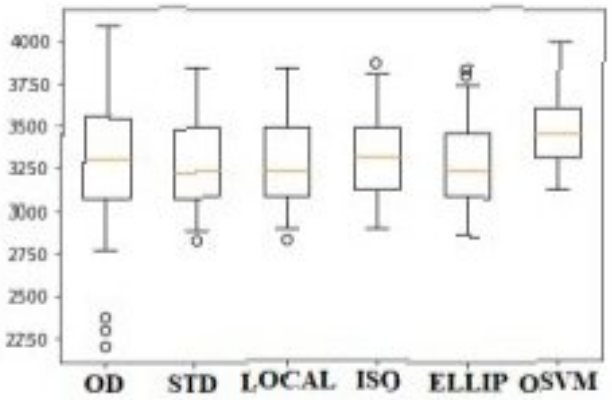
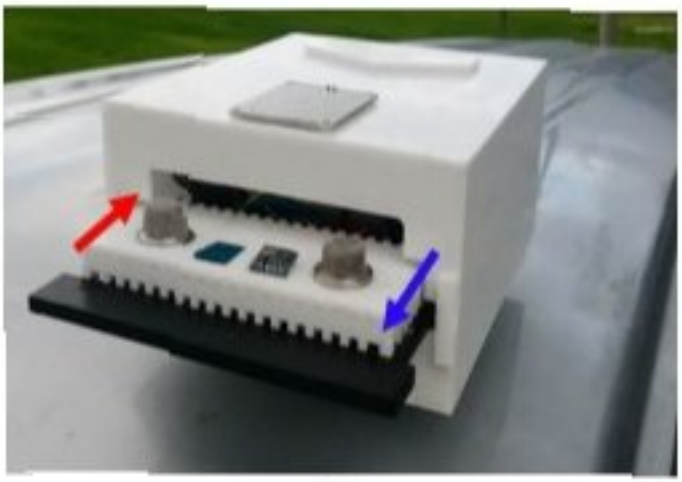


Figure 3. Box plot of data from the outlier detection methods.

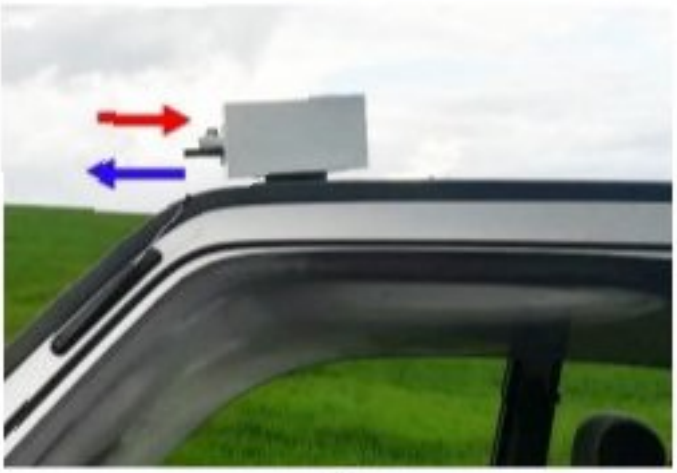
5.3. Supervised Classification

There are several methods of supervised learning that use different mathematical approaches. These approaches define the model complexity and determine the memory

The IoT device is designed to operate outside the vehicle used in this research (see Figure 6). In addition, at the top of the rain protector case, it has a rain sensor that, when activated, causes the system to retract inside the waterproof case and prevents the sensors from directly contacting water. Additionally, the sensors go into power-saving mode to avoid erroneous readings. The IoT device is configured to take samples every minute because it is the average time measured in real tests of how long a vehicle takes to travel roughly 250 m in a city with traffic. Each IoT system has an identifier that is indifferent to the car in which it is placed. Figure 6 shows both the waterproof case that was designed for the system and the installation of the system in the vehicle.



(a)



(b)

Figure 6. Prototype of the IoT device that was built for monitoring environmental conditions. (**a**) Waterproof case. (**b**) IoT device operating on the roof of the vehicle. Red arrow: The rain sensor detects drops on the waterproof case and activates the mechanism to save the sensors. Blue arrow: The car engine powers on and activates the IoT device, which gets out from the waterproof case.

6.3. *Data Analysis*

The algorithms used in the outlier detection techniques and supervised classification models have been established. These models are developed to determine their classification performance. Datasets were randomly divided ten times into training sets and test sets to train ML models, where 10% of the samples were part of the test set. The metrics used were as follows: accuracy, sensitivity, specificity, and recall using the confusion matrix method. It is important to remember that pruned datasets are temporarily stored to train ML models, then just the outlier detection method will prune incoming data. As shown in Table 3, where we present the accuracy of each model, the SVM algorithm performs well below expectations despite using different kernel functions. On the other hand, the Naive Bayes algorithm has an average performance of 75%, which indicates that it has errors in

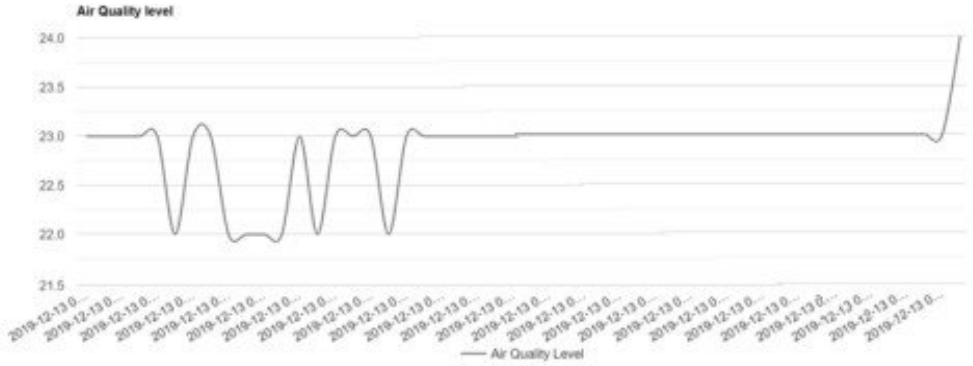


Figure 11: The air quality level in the morning on 13th of December 2019

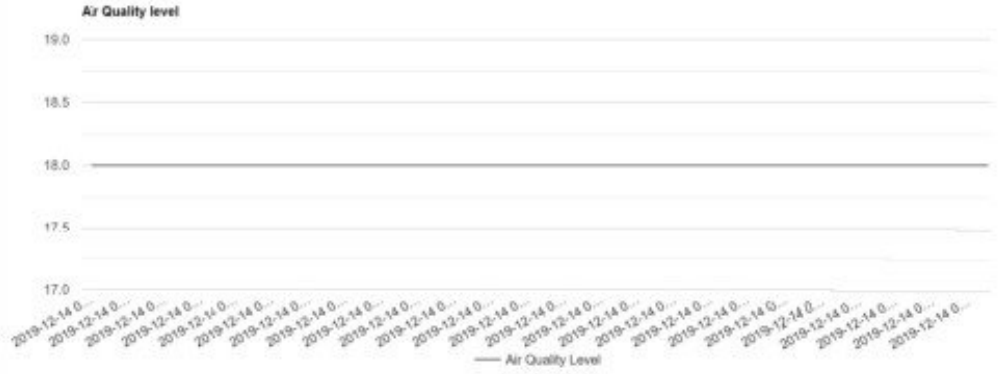


Figure 12: The air quality level in the night on 13th of December 2019

5. Conclusion

As a conclusion, the Air Quality Monitoring System with IoT has achieved the objective that able to detect air pollution or harmful gases by using MQ135 gas sensor. It is also capable of spreading awareness of how important to know the effects of air pollution towards healthy and environment. The following objective which is to develop the monitoring system of air pollution for environmental sensing application using Internet of Things (IoT), has also been fulfilled. The entire system includes the hardware components such as the Arduino Uno microcontroller, MQ135 gas sensor, I2C LCD and NodeMCU (ESP8266) as well as the software components comprising of the Arduino IDE, XAMPP platform and SQLyog for database system. These features are designed in order to work hand-in-hand in order to provide an ideal system for user. The last objective is to verify the function of the system in a different level of air quality level which also been accomplished. The experimental result with a different time and locations in order to get the data collection but for high reading of air quality level which is exceeding to the 100 ppm, couldn't be justified due to the no presence of any harmful gases or air pollution at the location selected. Thus, this ensures that the system is capable of performing the required safety and monitoring purposes.

THANK YOU