



VIT[®]

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

J Component report

Programme : B.Tech(CSE)
Course Title : Context Aware Computing Systems
Course Code : CSE2035
Slot : D2

Title : Emergency Car Ventilation System

Team Members :
Preyash | 20BPS1022
Biswadeep Ray | 20BPS1165
Disha Gupta | 20BPS1078

Faculty : Dr.Padmavathy T V

Sign:

Date: 15-04-2023

S.no	Content	Page No
1	Acknowledgement	3
2	Abstract	4
3	Introduction	4
4	Literature Survey	6
5	Existing Work	8
6	Proposed System	9
7	System Architecture	9
8	Technology stack	10
9	Working Modules	10
10	Screenshots/Video	20
11	Conclusions	22

Acknowledgment

We would like to extend our heartfelt thanks to our teacher **Dr.Padmavathy T V**, whose guidance and support were invaluable throughout the project. Additionally, we would like to acknowledge the hard work and dedication of our team members **Preyash (20BPS1022)**, **Biswadeep Ray (20BPS1165)**, **Disha Gupta(20BPS1078)**, who have made significant contributions to the project.

We would also like to acknowledge the support and encouragement of our families and friends, who have been a constant source of motivation throughout the project. Their unwavering faith in us has helped us overcome various challenges and stay focused on our goals. Finally, we would like to thank the institution for providing us with the necessary resources and facilities to complete this project.

We recognize that the successful completion of this project was a collaborative effort, and we are grateful for the contributions of everyone involved. Thank you all for your invaluable support."

Abstract:

The purpose of this project is to completely focus on the life of a living being and save it. The Project deals with providing a completely risk free situation inside a car by providing a spot which would open when there is lack of oxygen and as well increase in temperature and will provide fresh air inside the car to maintain car environment. Forgetting a child in the car is most often caused by everyday changes and vague multitasking while the child is in the car. The project helps in reducing the number of deaths caused due to the lack of oxygen in the car by opening the air vents and allowing the oxygen present in the environment to enter the car.

The implementation includes configuring gas and temperature sensors with the Arduino board, NodeMCU ESP8266 as a wifi access point and a servo motor, the gas and temperature sensors would check for the proper breathing conditions. A React Native mobile app is developed to connect to NodeMCU wifi. In abnormal conditions the emergency message would be displayed in user's mobile, the user will have the option to open the vent/windows. In case, the user doesn't respond to emergency through phone for a certain period of time the vents would open automatically using servo motor. The servo motor then comes back to the original position after the abnormal levels of gas and smoke reaches the safe limit. HTTP server hosted on NodeMCU exposing REST APIs is used to turn on and off servo motor based vent. Serial Peripheral Interface and I2C communication channels established between Arduino and NodeMCU for Machine 2 Machine communication enabling exchange of data.

Introduction

“Human comfort zone” how will you define it? As an automobile engineer, we will define it as, an atmosphere that provides relax and causes less fatigue to a human which can be achieved by an air conditioning of the particular cabin that includes maintaining the cabin's temperature between 20 to 25 degree centigrade and removing the humidity from the cabin's atmosphere, in short by fitting air conditioner in that cabin but now the question arises How is it possible in a car's cockpit? A car air conditioning system consists of a compact version of the components of the normal air conditioner that has an evaporator, compressor, condenser, expansion device, and a fan which are fitted in a car to provide air conditioning inside the passenger's compartment. This air conditioning system takes power from the engine's crankshaft /battery inside the car and is operated by the passengers from the cockpit by pressing the button assigned to this system. What if car is standing so the engine is already

stopped but the same time battery of the car goes down too resulting in stopping of air conditioning system and at that particular time there is passenger sitting inside the car, the result would be death of the passenger due to suffocation. The lack of oxygen and high temperature inside the car would lead to a serious accident. Many such cases have been reported all around the world.

According to a survey conducted by the national security council of the United States, every state has reported at least one death since 1998 due to high temperature and suffocation inside the car.

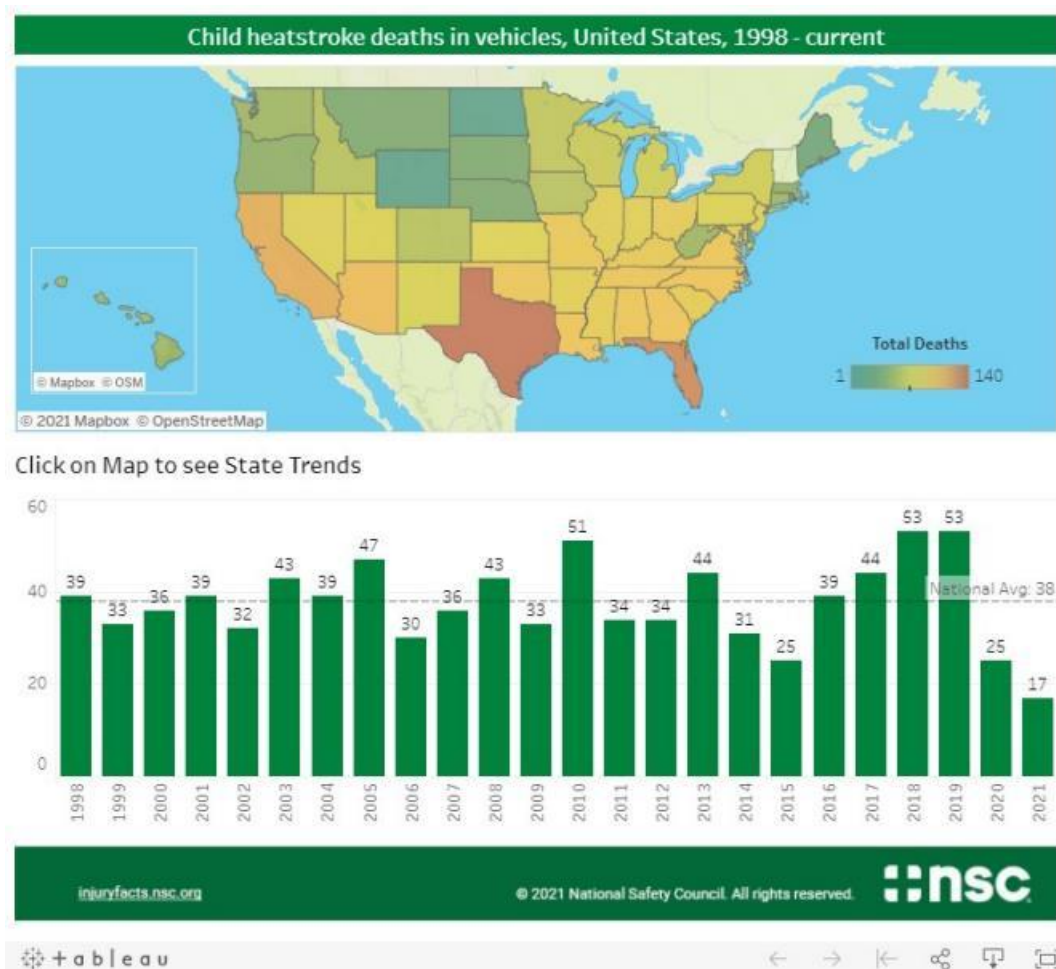


Figure 1. Death rate in USA

Literature Survey

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 a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

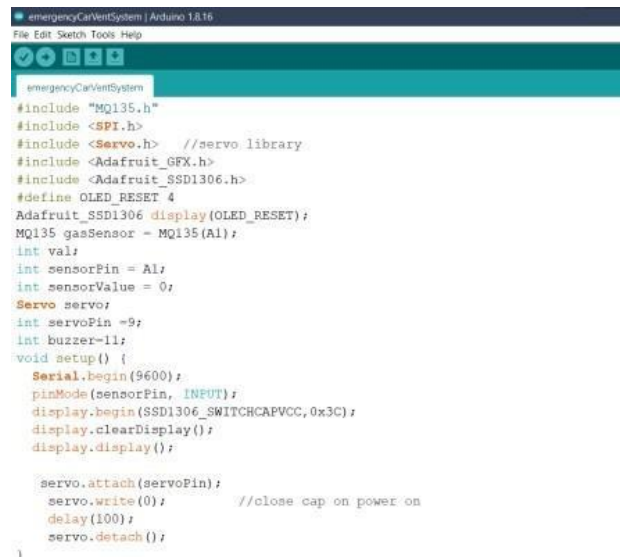
The key features are –

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the microcontroller into a more accessible package.

NodeMCU ESP8266 is a popular microcontroller board that is widely used in Internet of Things (IoT) projects. It is based on the ESP8266 Wi-Fi module and is compatible with the Arduino IDE. Here are some of its key features-

- Its ability to have its own wifi for a certain range and the fact it can communicate with other NodeMCU module
- Wi-Fi connectivity: The NodeMCU ESP8266 has built-in Wi-Fi connectivity, which makes it easy to connect to the internet and communicate with other devices over a wireless network.
- Arduino IDE compatibility: If you prefer to use the Arduino IDE, you can program NodeMCU ESP8266 using the Arduino core for ESP8266. This makes it easy to get started if you are already familiar with Arduino programming.
- GPIO pins: NodeMCU ESP8266 has multiple GPIO pins that can be used for digital input/output or analog input. These pins can be used to control sensors, actuators, and other devices.

- ADC: NodeMCU ESP8266 has an onboard analog-to-digital converter (ADC) that can be used to read analog signals from sensors.
- Easy to use: NodeMCU ESP8266 is easy to use and requires minimal setup. You can connect it to your computer using a USB cable and start programming right away.
- Low cost: NodeMCU ESP8266 is relatively inexpensive compared to other microcontroller boards, making it an affordable option for hobbyists and DIY enthusiasts.



```

emergencyCarVentSystem | Arduino 1.8.16
File Edit Sketch Tools Help

emergencyCarVentSystem
#include "MQ135.h"
#include <SPI.h>
#include <Servo.h> //servo library
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
MQ135 gasSensor = MQ135(A1);
int val;
int sensorPin = A1;
int sensorValue = 0;
Servo servo;
int servoPin = 9;
int buzzer=11;
void setup() {
  Serial.begin(9600);
  pinMode(sensorPin, INPUT);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.display();

  servo.attach(servoPin);
  servo.write(0); //close cap on power on
  delay(100);
  servo.detach();
}

```

Figure 2. Extract of Arduino code

Algorithm to Convert Voltage to temperature:-

The formula to convert the voltage to centigrade temperature for LM35 :-

- Centigrade Temperature = Voltage Read by ADC / 10 mV(mills Volt)
- Now that we have the ADC value in a variable, we have to do two things:
- Convert the ADC Value to equivalent voltage.
- Convert the voltage to temperature reading.

For the first step, we have to divide the ADCValue with the maximum possible value of the ADC. The Arduino ADC is a 10-bit ADC. So, the maximum value can be $(2^{10}-1) = 1023$. We subtract 1 because the count starts from zero. We get the value of ADC as 1023 if the voltage is 5v.

The calculation follows simple unitary method:

If the value is 1023, the voltage = 5

If the value is 1, the voltage = $(5/1023)$

If the value is ADCValue, the voltage is $= (5/1023)ADCValue$

In Arduino, we write this as follows:

float tempVoltage = 0;

tempVoltage = (5/1023)ADCValue;

Since our sensor output is in mV, we will multiply the entire value with 1000 to convert from voltage to milliVolts. So, our new statement becomes:

tempVoltage = ((5/1023)ADCValue)*1000;

Now that we have the voltage of the sensor, we need to convert it to equivalent temperature.

Remember, we saw earlier that the scaling factor of LM35 is 10mV/degree Celsius? That is, For every degree celsius of the temperature, the sensor output is 10 mV.

So, if we have the mV reading of the sensor and divide it by 10, the new value will be our temperature. Let's do that:

float temperature = 0;

temperature = tempVoltage/10; //temperature in Celcius

Existing Work/System

In countries like Sudan, the temperature rises up to 45°C when a car is parked directly under the sun. This extreme rise in the temperature levels inside the parked car cabin is caused due to the heat transferred by a combination of conduction, convection and radiation. Studies indicate that each year children die from Hyperthermia (an acute condition that occurs when the body absorbs more heat as a result of being left in a parked car). To determine the specifications of the fan and its placement, temperature variation of a parked car directly under the sun was observed and flow simulation and analysis were performed in ANSYS FLUENT workbench where analysis of HVAC systems based on numerical calculations with sufficient accuracy and acceptable results is now possible for HVAC researchers by using improved computer technology and CFD techniques. This study is considering an axial cooling fan which will be located between the back speakers' area behind the rear seat controlled by an Arduino board based circuit that functions according to analogue signals received from LM35 sensors and based on these readings it operates, utilizing a dedicated battery.

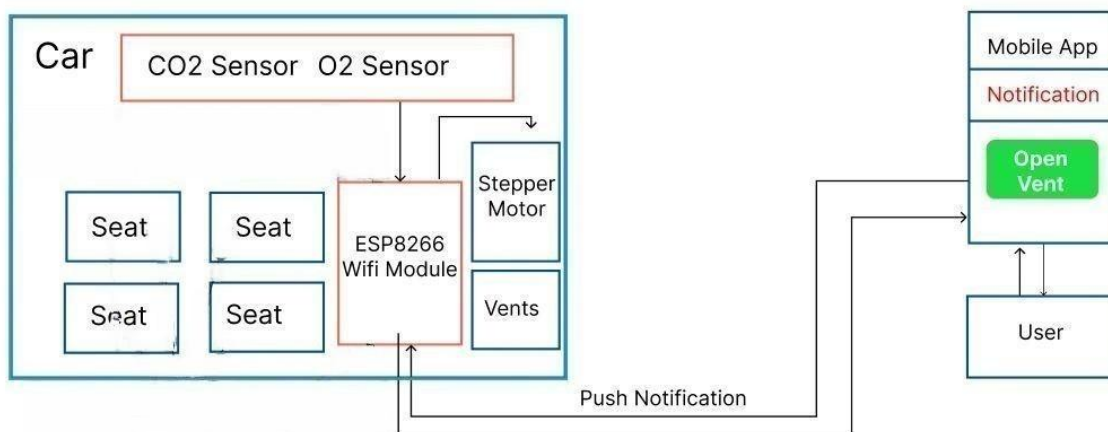
Proposed System

We're working on developing an automatic system that maintains appropriate ventilation in cars in emergency circumstances while also ensuring the safety of the people inside.

Our project “Emergency Car Vent System” includes Arduino UNO-R3 development board, NodeMCU ESP8266, LM-35 temperature sensor , MQ-135 sensor , O-LED , servo motor , jumper wire , breadboard and buzzer. The circuit gets powered. The code is verified and compiled .Then the code is uploaded on the arduino UNO board. Once the code is uploaded on the arduino board the circuits start getting powered. The MQ-135 sensor is tested by creating a toxic environment around it and providing a dummy data. Temperature sensor is tested by increasing the temperature. A box is being used to create a car-like environment and the servo motor is tested.. The ppm values are constantly displayed on the led display.

In abnormal conditions the emergency message would be displayed in user’s mobile along with buzzer beeping. The user will have the option to open the vent/windows. Incase, the user doesn’t respond to emergency through phone for a certain period of time the vents would open automatically using servo motor. The servo motor then comes back to the original position after the abnormal levels of gas and smoke reaches the safe limit. The buzzer also stops beeping. HTTP server hosted on NodeMCU exposing REST APIs is used to turn on and off servo motor based vent. Serial Peripheral Interface and I2C communication channels established between Arduino and NodeMCU for Machine 2 Machine communication enabling exchange of data.

System Architecture



Technology Stack

- Arduino IDE
- MQ135 Sensor
- LM35 Temperature Sensor
- Servo Motor
- OLED Display
- Node MCU
- React Native App

Working Modules and Description

MQ135:



Figure 7. Mq135 Sensor

MQ-135 is a gas sensor and it works on the relation between concentration and conductivity of gases. When target pollution gas exists, the sensor's conductivity gets higher along with the gas concentration rising. The analog output is a concentration, i.e. increasing voltage is directly proportional to increasing concentration. This sensor has a long life and reliable stability as well.

MQ-135 gas sensor applies SnO_2 which has a higher resistance in the clear air as a gassensing material. When there is an increase in polluting gases, the resistance of the gas sensor decreases along with that. To measure PPM using the MQ-135 sensor we need to look into the (R_s/R_o) v/s PPM graph taken from the MQ135 datasheet.

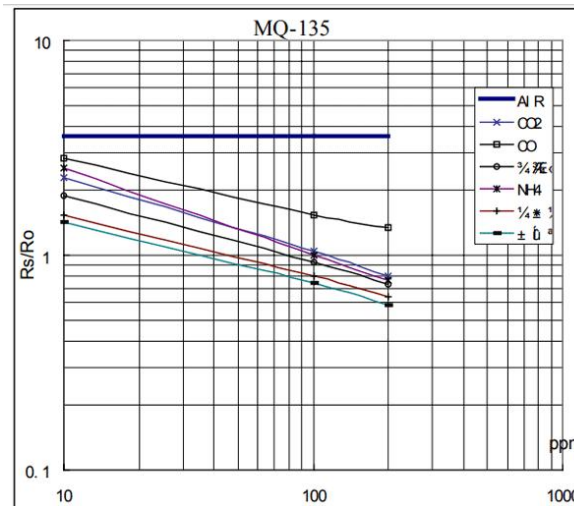


Figure 8. Mq135 graph sensor datasheet graph to calculate ppm

The above figure shows the typical sensitivity characteristics of the MQ-135 for several gases in the condition =: Temp: 20, Humidity: 65%, O₂ concentration 21%, RL=20kΩ, Rs: sensor resistance at various concentrations of gases.

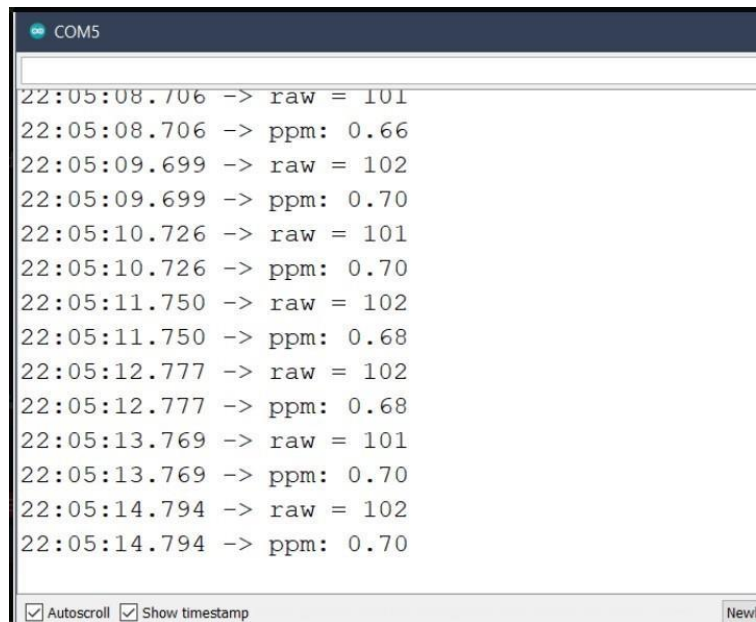
The value of Ro is the value of resistance in fresh air (or the air with which we are comparing) and the value of Rs is the value of resistance in Gas concentration. First we should calibrate the sensor by finding the values of Ro in fresh air and then use that value to find Rs using the below formula:

$$\text{Resistance of sensor}(R_s): R_s = (V_c / V_{RL} - 1) \times R_L$$

V_c: 5V(The working voltage of MQ135) V_{rl}: The read voltage from analog pin. Once we calculate Rs and Ro we can find the ratio and then using the graph shown above we can calculate the equivalent value of PPM for that particular gas. Here is the sample code for mq135 which is used to test mq135. The reading at serial monitor changes as the smoke in room rises and air quality become toxic.

```
void loop() {
  val = analogRead(A1);
  Serial.print ("raw = ");
  Serial.println (val);
  float ppm = gasSensor.getPPM();
  Serial.print ("ppm: ");
  Serial.println (ppm);
}
```

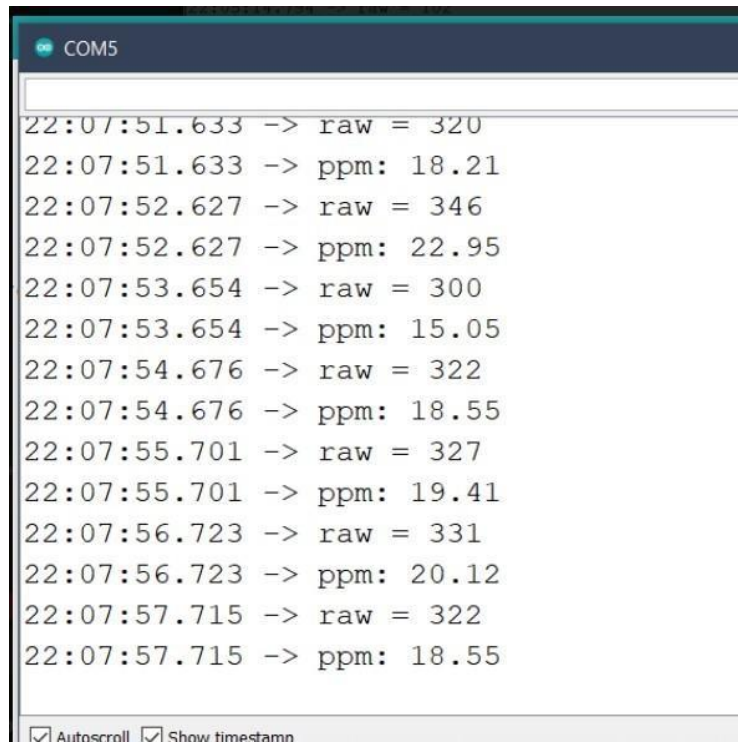
Figure 9. Code for mq135



```
22:05:08.706 -> raw = 101
22:05:08.706 -> ppm: 0.66
22:05:09.699 -> raw = 102
22:05:09.699 -> ppm: 0.70
22:05:10.726 -> raw = 101
22:05:10.726 -> ppm: 0.70
22:05:11.750 -> raw = 102
22:05:11.750 -> ppm: 0.68
22:05:12.777 -> raw = 102
22:05:12.777 -> ppm: 0.68
22:05:13.769 -> raw = 101
22:05:13.769 -> ppm: 0.70
22:05:14.794 -> raw = 102
22:05:14.794 -> ppm: 0.70
```

☒ Autoscroll ☒ Show timestamp New

Figure 10. Output of mq135 before smoke



```
22:07:51.633 -> raw = 320
22:07:51.633 -> ppm: 18.21
22:07:52.627 -> raw = 346
22:07:52.627 -> ppm: 22.95
22:07:53.654 -> raw = 300
22:07:53.654 -> ppm: 15.05
22:07:54.676 -> raw = 322
22:07:54.676 -> ppm: 18.55
22:07:55.701 -> raw = 327
22:07:55.701 -> ppm: 19.41
22:07:56.723 -> raw = 331
22:07:56.723 -> ppm: 20.12
22:07:57.715 -> raw = 322
22:07:57.715 -> ppm: 18.55
```

☒ Autoscroll ☒ Show timestamp

Figure 11. Output of mq135 after smoke

From fig10 and fig11 it is clear that mq135 is reading the toxicity level in the air and the ppm is rising as the air become toxic.

OLED Display:

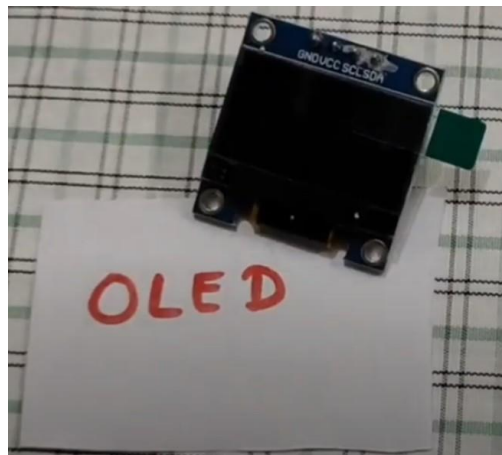


Figure 12. OLED Display

OLED is known as organic light emitting diode. This device comprises an electroluminescent layer manufactured with the organic layer that releases photons of light when current passes through OLED.

When the voltage is applied to the OLED. The current flows from the cathode to anode through the organic layers of the OLED. The cathode gives the electrons to the emissive layer of organic molecules and the anode removes electrons from the conductive layer of organic molecules. At the boundary between the conductive and emissive layer, electron holes are created. These holes are filled by the electrons and the OLED emits light. The color of the OLED depends upon the organic molecules used. Here is the sample code for OLED display. Using this we have displayed the ppm value of gas in air on OLED.

```
display.setTextSize(2);  
display.setCursor(28,5);  
display.println(ppm);  
display.display();  
display.clearDisplay();  
delay(1000);
```

Figure 13. OLED Code

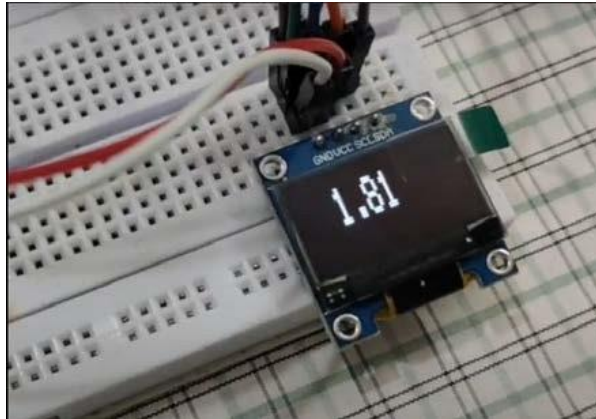


Figure 14. OLED displaying ppm

From fig we can see that the OLED is working and is displaying the value of ppm.

BUZZER:



Figure 15. Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke. Here we have attached the sample code for buzzer which we have used for testing.

```
void loop() {  
    tone(buzzer, 450);  
    delay(1000);  
    noTone(buzzer);  
    delay(1000);  
}
```

Figure 16. Code for Buzzer

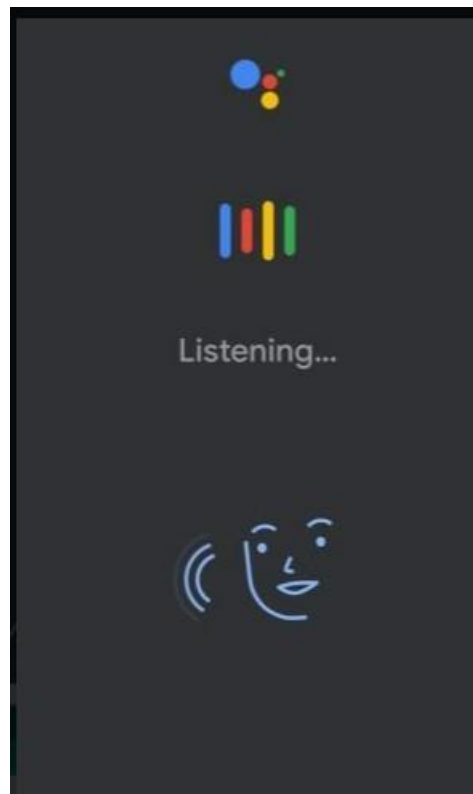


Figure 17. Buzzer Voice detected

We have used google to detect voice and it clearly shows that there is some sound which clears that buzzer is beeping.

TEMPERATURE SENSOR:

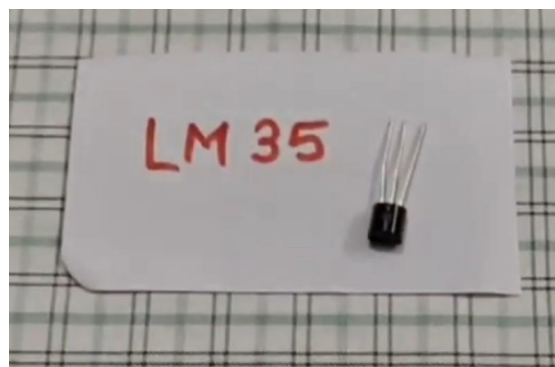


Figure 18. LM35 temperature sensor

LM35, a temperature sensor, uses the basic principle of a diode , where as the temperature increases, the voltage across a diode increases at a known rate. By precisely amplifying the voltage change, it is easy to generate an analog signal that is directly proportional to temperature.

In the features of LM35 it is given to be +10 mills volt per degree centigrade. It means that with an increase of 1 degree Celcius in the temperature, the output at the Vout pin will

increase by 10 millivolts. Here is the sample code for temperature used for testing the sensor. It reads the analog value and converts the analog value to temperature using the formula and displays it on the serial monitor.

```
temp= analogRead(A0);  
temp=temp*0.48828125;  
Serial.print("Temperature: ");  
Serial.print(temp);  
Serial.print("C");  
Serial.println();
```

Figure 19. LM35 code

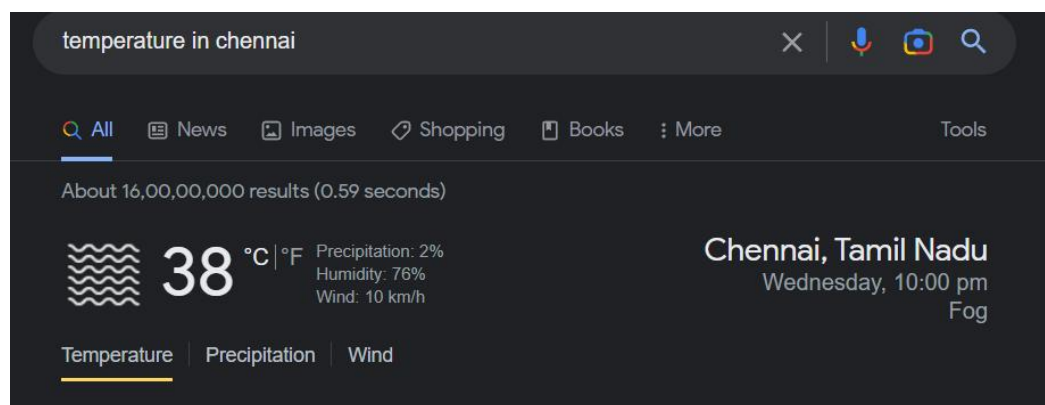


Figure 20. Temperature outside the room

Temperature inside the room would be of course greater than the temperature outside.

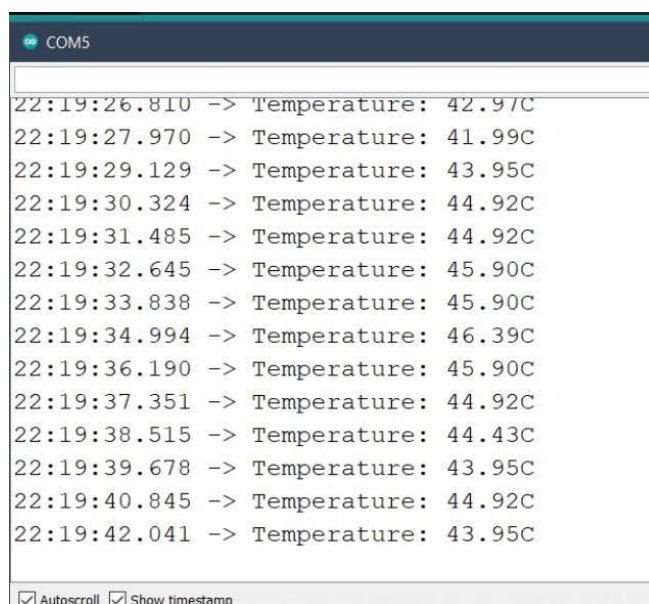


Figure 21. Output for temperature sensor

The output clearly shows that the temperature sensor is working properly and is giving the result.

SERVO MOTOR:

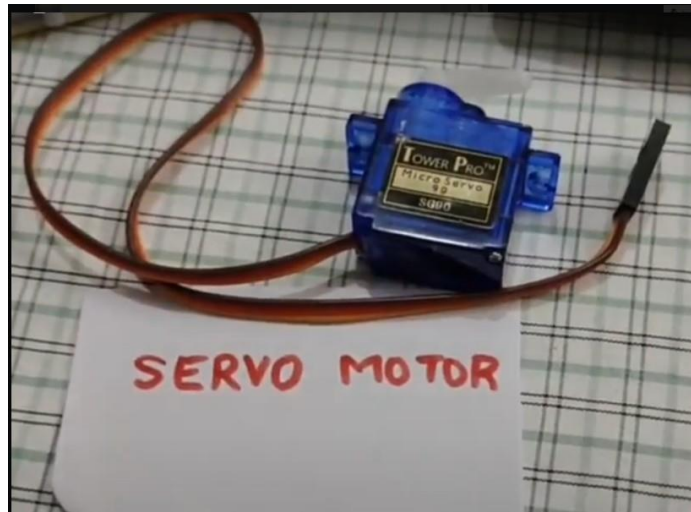


Figure 22. Servo Motor

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly, and a controlling circuit. First of all, we use gear assembly to reduce RPM and to increase torque of the motor. Say at initial position of the servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical signal is given to another input terminal of the error detector amplifier. Now the difference between these two signals, one comes from the potentiometer and another comes from other sources, will be processed in a feedback mechanism and output will be provided in terms of error signal. This error signal acts as the input for the motor and motor starts rotating. Now the motor shaft is connected with the potentiometer and as the motor rotates so does the potentiometer and it will generate a signal. So as the potentiometer's angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches a position that the output of potentiometer is the same as the external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation the motor stops rotating.

Here is the sample code for the servo motor which connects the servo motor to the arduino and allows it to function.

```
servo.attach(servoPin);  
servo.write(0);           //close cap on power on  
delay(100);  
servo.detach();
```

Figure 23. Servo motor code

The below fig and fig show code for the condition when will servo motor open and close...

```
if(ppm>10){  
    servo.attach(servoPin);  
    delay(1);  
    servo.write(0);  
    delay(3000);  
    servo.write(360);  
    delay(1000);  
    servo.detach();
```

```
if(ppm<=10){  
    servo.attach(servoPin);  
    servo.write(0);
```

Figure 24. Condition to control the servo motor's rotation



Figure 25. The lid(vent spot) is closed

```
22:07:54.676 -> raw = 322  
22:07:54.676 -> ppm: 18.55
```

Figure 26. The rise in ppm of the box

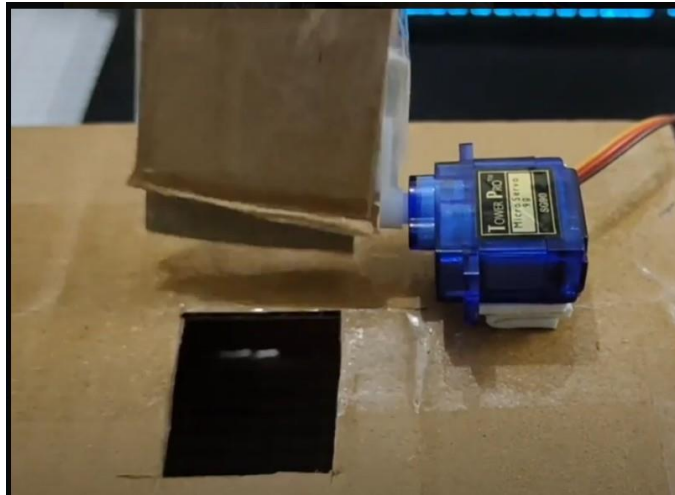


Figure 27. The lid(vent spot) is open now

Initially the lid is closed and then as the ppm rises above the condition the lid gets open. Hence it shows proper working of servo motor.

NodeMCU:



Figure 28. The NodeMCU

- Added NodeMCU ESP8266 working as WiFi Access Point...creating a local wifi network.
- Developed React Native Mobile app which connects to NodeMCU WiFi
- HTTP server hosted on NodeMCU exposing REST APIs to turn on and off servo motor based vent
- User presses button on app which wireless sends request at API endpoint...notifying NodeMCU to turn motor on and off.

- Serial Peripheral Interface and I2C communication channels established between Arduino and NodeMCU for Machine 2 Machine communication enabling exchange of data

Setup Screenshots

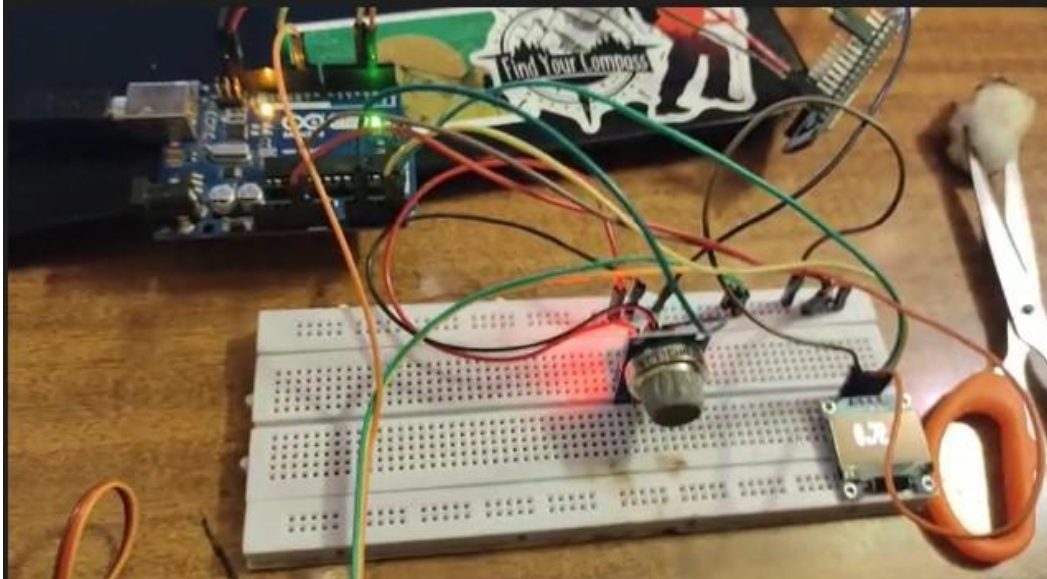


Figure 29. The Apparatus

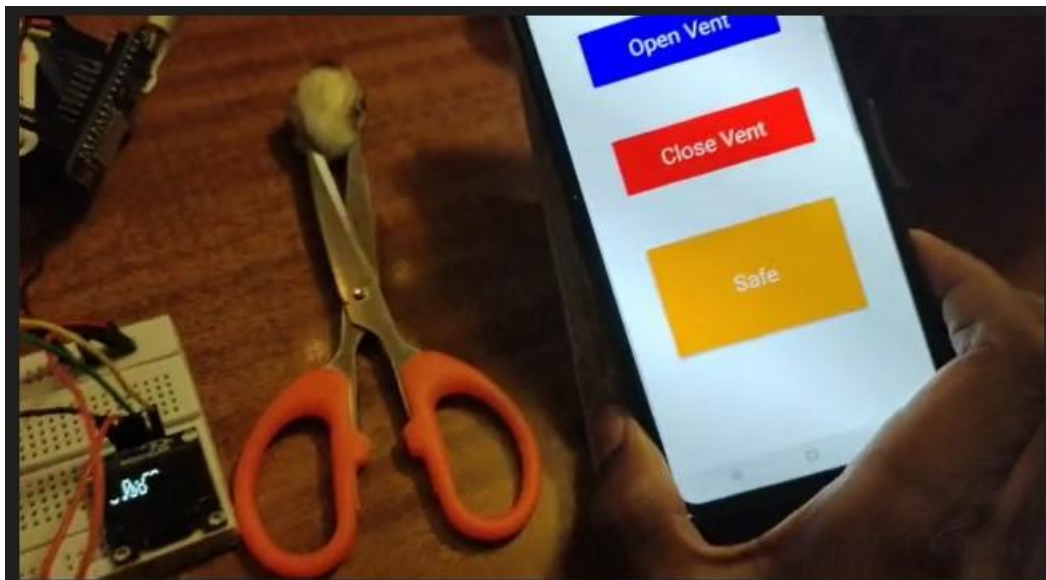


Figure 30. The App displays the safe condition

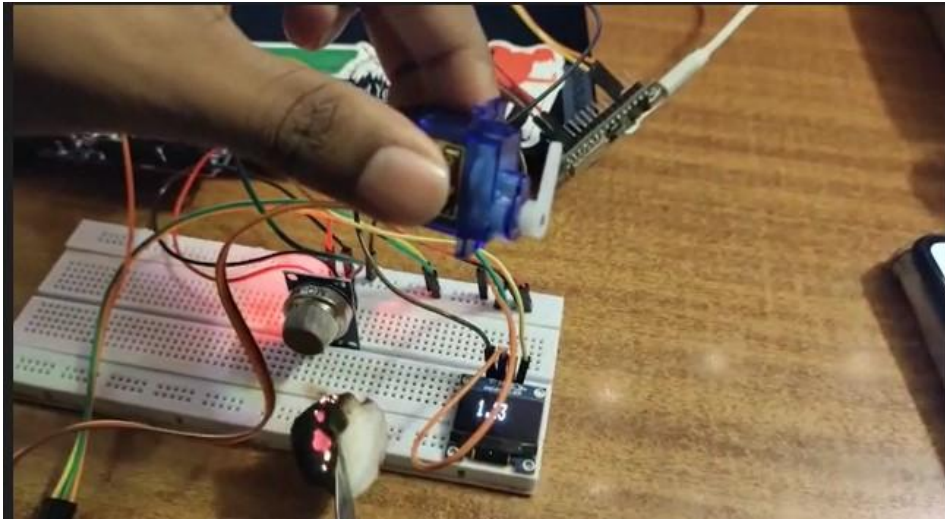


Figure 31. The motor is closed until the ppm value is in safe range

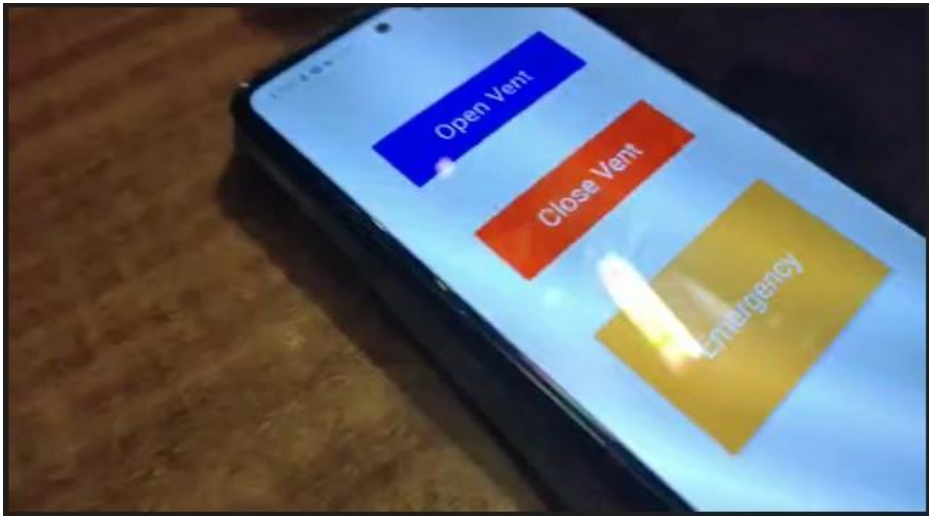


Figure 32. The app shows the emergency message

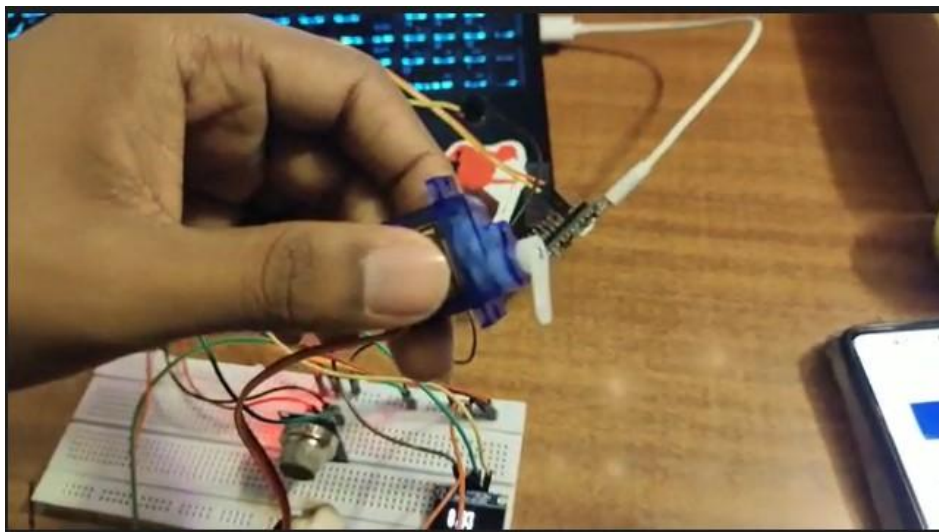


Figure 33. The servo motor turn on when command is received

Demonstration video Link

[https://drive.google.com/file/d/1ma7oQD8TiPPEiMAisBYX2PzfKAeLa8V0/view?usp=share link](https://drive.google.com/file/d/1ma7oQD8TiPPEiMAisBYX2PzfKAeLa8V0/view?usp=share_link)

Conclusion

We are solving a real world problem by our project. Our project intended to make a device which keeps checking the air quality inside a vehicle and if this air quality reaches some toxic/ unbreathable levels, the device opens some ventilation spots. Apart from this our device also signals the nearby persons so that they could also get alerted.

References

1. <https://randomnerdtutorials.com/guide-for-oled-display-with-arduino/>
2. <https://circuitdigest.com/microcontroller-projects/interfacing-mq135-gas-sensor-with-arduino-to-measure-co2-levels-in-ppm>
3. https://www.researchgate.net/publication/327451321_Arduino-Based_Real_Time_Air_Quality_and_Pollution_Monitoring_System
4. <https://www.circuitstoday.com/lm35-and-arduino-interfacing#%3A~%3Atext%3DSo%20lets%20get%20to%20building%20Celsius%20to%20%2B150%20degree%20celsius>