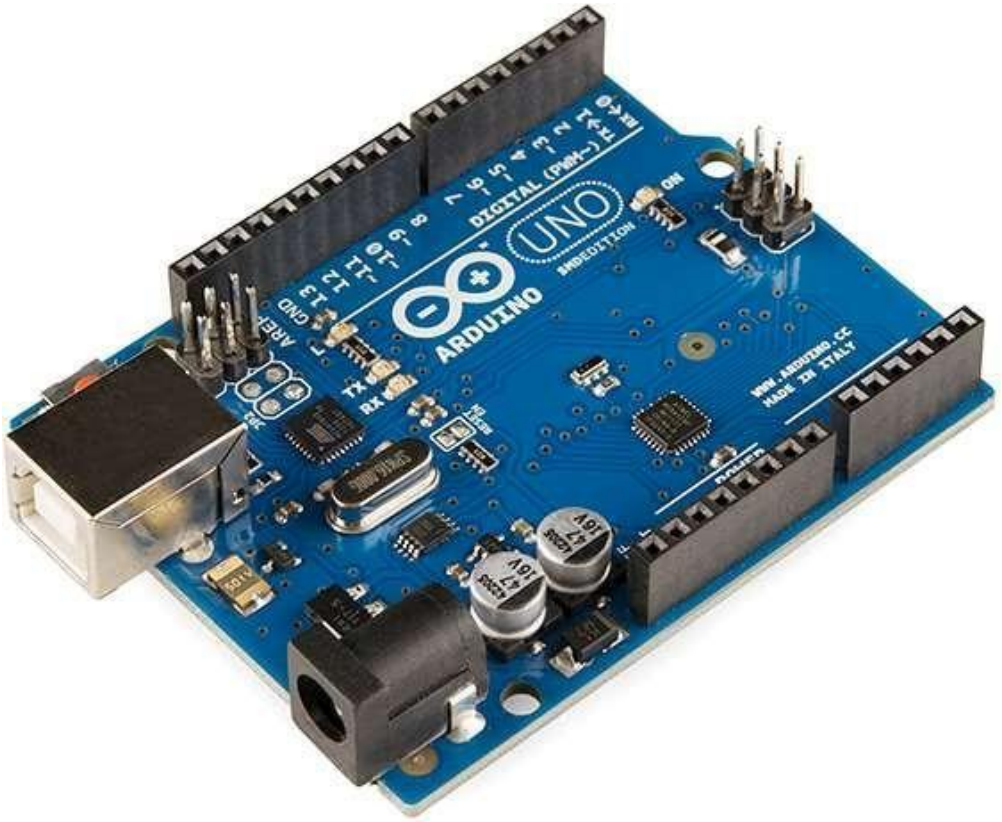
SMOKE AND FIRE DETECTOR USING ARDUINO



SUBMITTED TO:

Akhilesh Dhar Sir

……..

SUBMITTED BY:

Vaibhav Yashdev Kanojia	2020UEA6567
Amogh Alone	2020UEA6584
Harsh Nagar	2020UEA6588
Shivam Thakur	2020UEA6589
Naman Kumar	2020UEA6598
Aayush Kumar Singh	2020UEA6619

ABOUT THE PROJECT

As we all know, architecture represents the accomplishment, wealth, and creativity of a city or a nation. No one wants to afford damage to them. Fire accident is one of the major cause which harms not only buildings but also human beings. The most usual causes of house fires are electrical distribution, lighting systems, cooking, Etc. In this technological era, the use of engineering is the best way to avoid such accidents. Today, we will build an interesting project using Arduino.

Fire alarm detection project demonstrates the use of Arduino to build a fire detection system using the sensors which measure temperature of the environment. The fire alarm detection system remains idle unless there is a fire outbreak. The system in idle condition glows green LED.

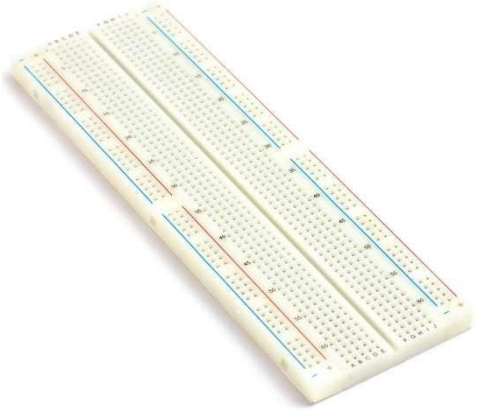
COMPONENTS USED

1. Breadboard
2. 2 Servo Motors
3. LCD Screen
4. Arduino Uno
5. 9V Battery
6. Gas Sensor
7. Push Button
8. Temperature sensor
9. Buzzer
10. LED

Before jumping on procedure, we will now discuss each component one by one

1. Breadboard

A Breadboard or protoboard is a construction base for prototyping of electronics. A breadboard is used to build and test circuits quickly before finalizing any circuit design. The breadboard has many holes into which circuit components like ICs and resistors can be inserted. A typical breadboard is shown below:



The bread board has strips of metal which run underneath the board and connect the holes on the top of the board. The metal strips are laid out as shown below. Note that the top and bottom rows of holes are connected horizontally while the remaining holes are connected vertically.

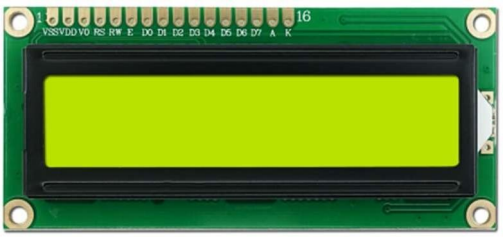
2. Servo Motors

A servomotor (or servo motor) is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity and acceleration.[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

Servomotors are not a specific class of motor, although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system.
A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft.

3. LED screen

A LED display is a flat panel display that uses an array of light-emitting diodes as pixels for a video display.



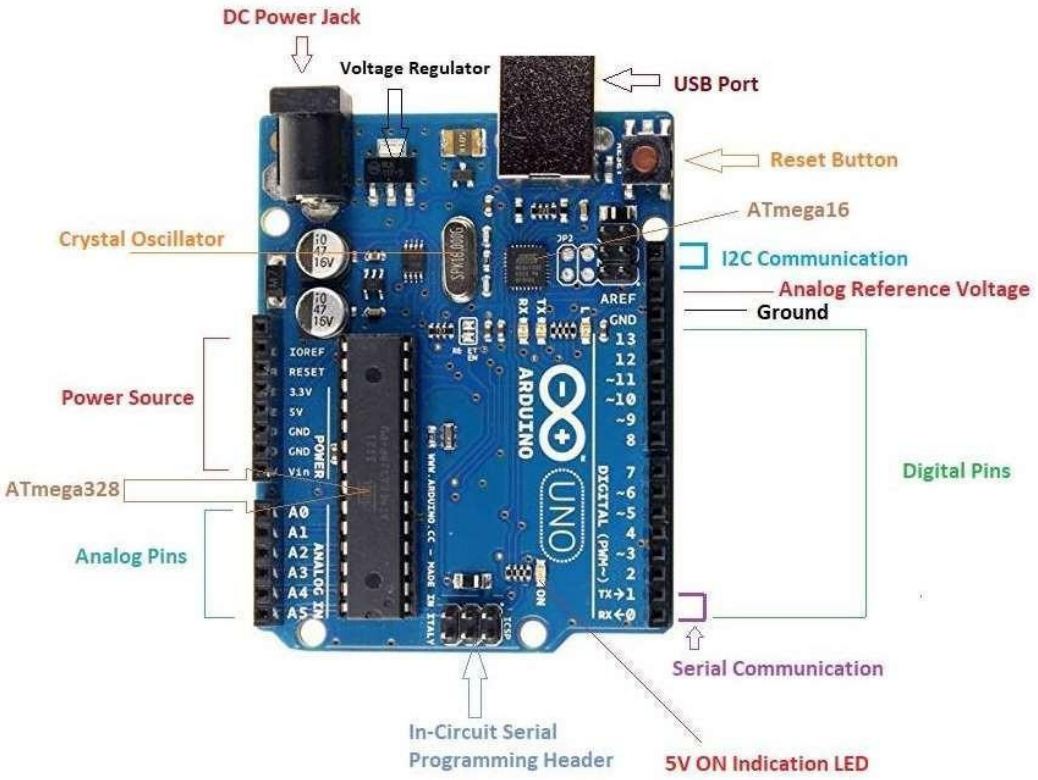
The LCDs have a parallel interface, meaning that the microcontroller has to

manipulate several interface pins at once to control the display. The interface consists of the following pins:

- A **register select (RS) pin** that controls where in the LCD's memory you're writing data to. You can select either the data register, which holds what goes on the screen, or an instruction register, which is where the LCD's controller looks for instructions on what to do next.
- A **Read/Write (R/W) pin** that selects reading mode or writing mode
- An **Enable pin** that enables writing to the registers
- 8 **data pins (D0 -D7)**. The states of these pins (high or low) are the bits that you're writing to a register when you write, or the values you're reading when you read.

4. Arduino Uno

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller. 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.



It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts

General pin functions

- **LED**: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it is off.
- **VIN**: The input voltage to the Arduino/Genuino board when it is using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
- **3V3**: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND**: Ground pins.
- **IOREF**: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source, or enable voltage translators on the outputs to work with the 5V or 3.3V.
- **Reset**: Typically used to add a reset button to shields that block the one on the board.

5. 9V Battery

An electric battery is a source of electric power consisting of one or more electrochemical cells with external connections for powering electrical devices.

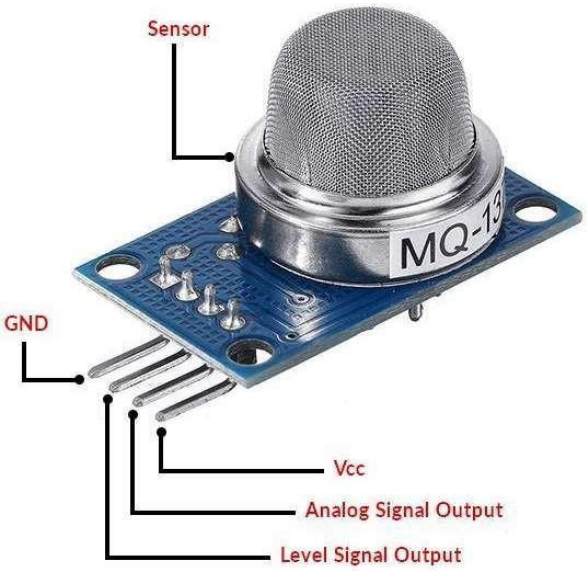
When a battery is supplying power, its positive terminal is the cathode and its negative terminal is the anode. The terminal marked negative is the source of electrons that will flow through an external electric circuit to the positive terminal. When a battery is connected to an external electric load, a redox reaction converts high-energy reactants to lower-energy products, and the free-energy difference is delivered to the external circuit as electrical energy. Historically the term "battery" specifically referred to a device composed of multiple cells; however, the usage has evolved to include devices composed of a single cell.

6. Gas Sensor

Gas sensors (also known as gas detectors) are electronic devices that detect and identify different types of gasses. They are commonly used to detect toxic or explosive gasses and measure gas concentration. Gas sensors are employed in factories and manufacturing facilities to identify gas leaks, and to detect smoke and carbon monoxide in homes. Gas sensors vary widely in size (portable and fixed), range, and sensing ability. They are often part of a larger embedded system, such as hazmat and security systems, and they are normally connected to an audible alarm or interface

A basic gas sensor has 6 terminals in which 4 terminals (A, A, B, B) acts input or output and the remaining 2 terminals (H, H) are for heating the coil. Of these 4 terminals, 2 terminals from each side can be used as either input or output

Now let's see the pin description of the gas sensor module which we will generally use with an Arduino. The gas sensor module basically consists of 4 terminals



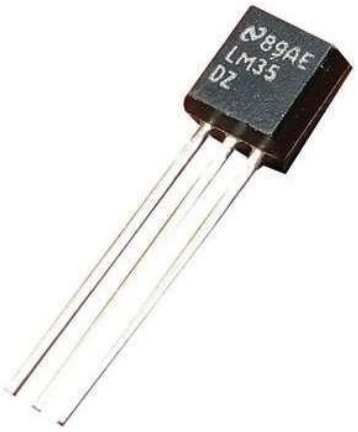
- **Vcc** – Power supply
- **GND** – Power supply
- **Digital output** – This pin gives an output either in logical high or logical low (0 or 1) that means it displays the presence of any toxic or combustible gases near the sensor.
- **Analog output** – This pin gives an output continuous in voltage which varies based on the concentration of gas that is applied to the gas sensor.

7. Push Button

Electric Push Button Switch Function - electric push buttons are used with electric circuits, they contain the electronics required to either make or break circuits, depending on the requirements of the application. They are used to switch devices on and off, for example light switches.

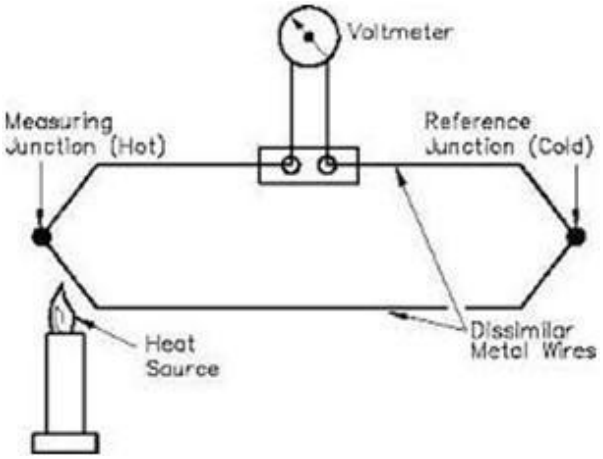
8. Temperature Sensor

Have you ever left your smartphone in your car on a hot day? If so, your screen might have displayed an image of a thermometer and a warning that your phone has overheated. That is because there is a tiny embedded temperature sensor that measures the interior temperature of your phone.



Once the inside of the phone reaches a certain temperature (iPhones shut down at approximately 113 degrees Fahrenheit, for example), the temperature sensor sends an electronic signal to an embedded computer. This, in turn, restricts users from accessing any applications or features until the phone has cooled back down, as running programs would only further damage the phone's interior components.

Non-contact temperature sensors are usually infrared (IR) sensors. They remotely detect the IR energy emitted by an object and send a signal to a calibrated electronic circuit that determines the object's temperature.



9. Buzzer

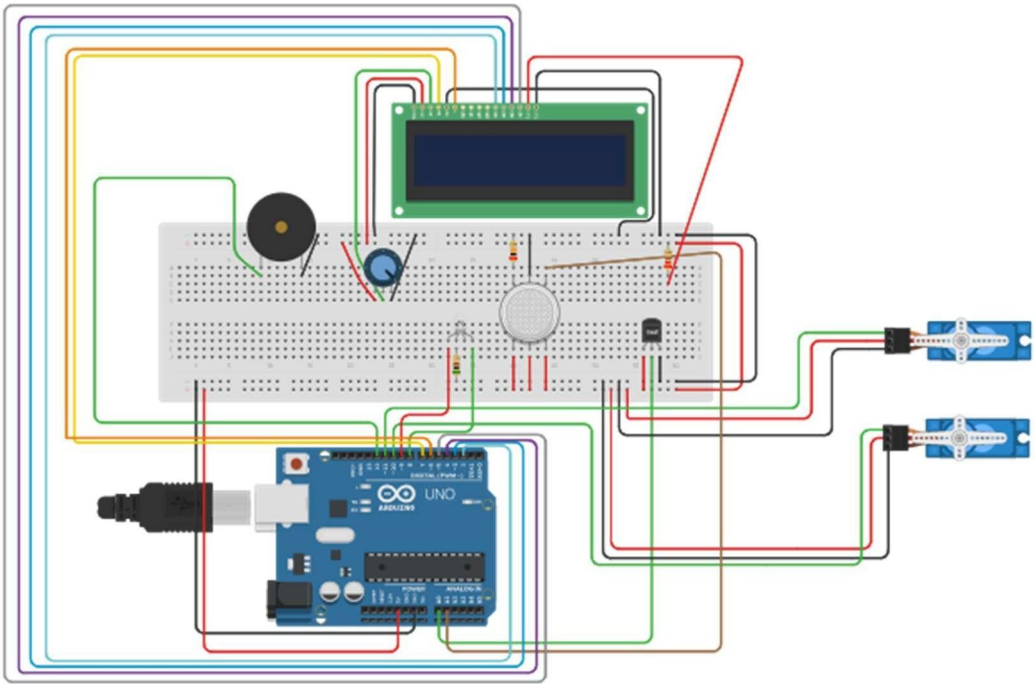
A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or Arduino.
Here the buzzer will beep if our detector detects any smoke.

10. LED

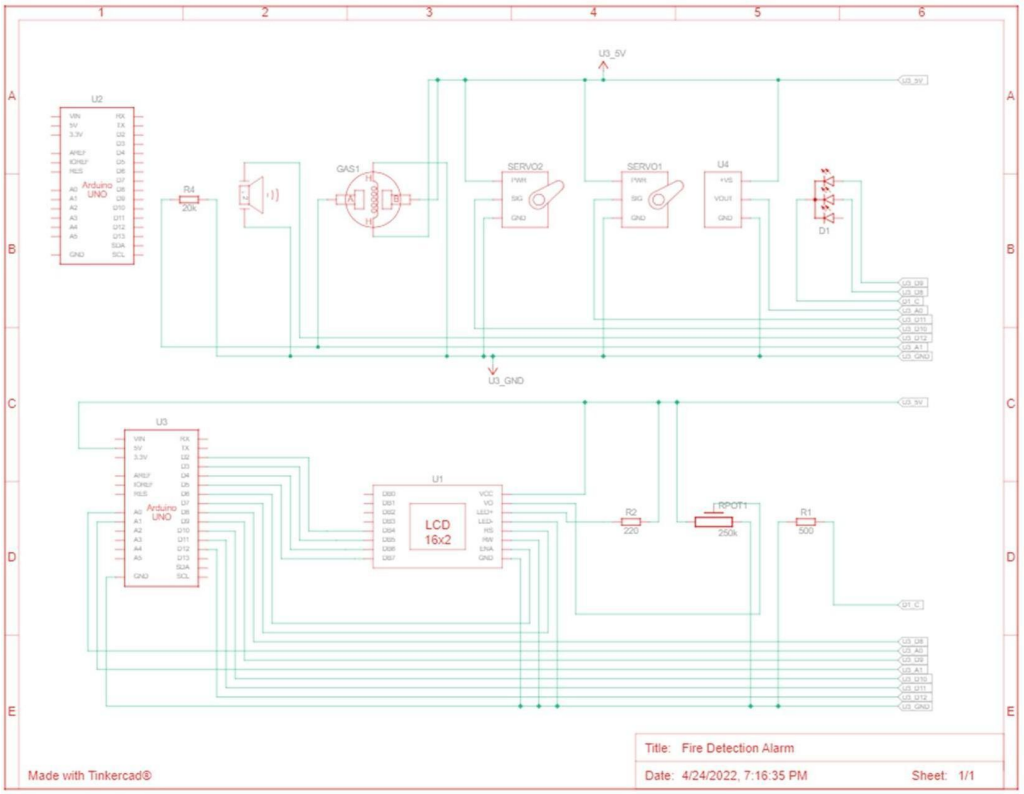
A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The colour of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor.[5] White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device.

PROCEDURE, CIRCUITS AND ARDUINO CODE

The working of the project is pretty straight forward. With the help of above-mentioned sensors, the information about the surrounding is fed to the processor, then a decision is made weather to alarm or not. If the smoke and temperature exceed the normal room threshold then alarm will light up and the buzzer will make sound, otherwise the system will remain at standby.
Now we will see the connections made for the circuit in order to perform fire and smoke detection.



The connection of the circuit will look like this. Now to have a better understanding of the circuit we will have a look at the schematic diagram of the circuit.



Audruino Code:

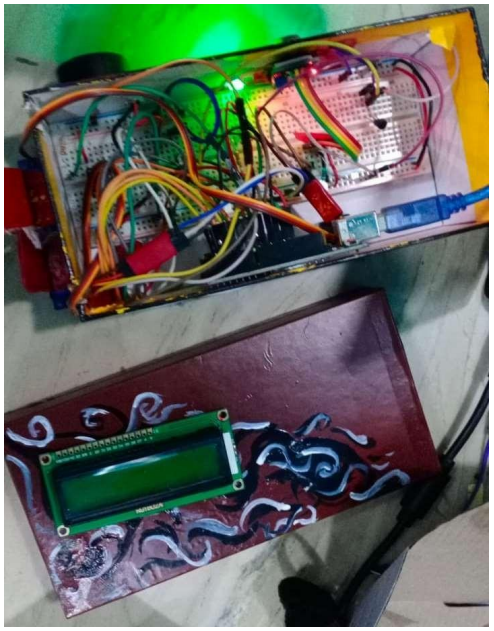
```
#include<Servo.h>//header file for servo
#include <LiquidCrystal.h>//header file for LCD


//first of all we will use the TMP36 which is a temperature sensor that outputs
//a voltage that's proportional to the ambient temperature.

// We'll use analog input 0 to measure the temperature sensor's signal pin.
//Temparature Sensor
const int temperaturePin = 0; //The output of tmp36 is connected to A0 of arduino

//buzzer
const int buzzer = 12; //buzzer is connected to D12 on the arduino

//Gas Sensor
int gasSensorPin=A1;//Gas sensor output is connected to A1 of Arduino
int sensorval;//For storing the value sensed by gas sensor

//Doors
Servo
servo1,servo2; int
servo1Pin=11; int
servo2Pin=10;

//RGB LED
int red_led=9;//Red terminal of RGB LED is connected to D9 of Arduino
int green_led=8;//Green terminal of RGB LED is connected to D8 of Arduino

//LCD
LiquidCrystal lcd(7, 6, 2, 3, 4, 5);//Sets the interfacing pins on Arduino that are
connected to LCD
//(rs, enable, d4, d5, d6, d7)
//7-Rs,6-E(Enable), 5,4,3,2 are the inputs->4 bit mode

//reset button
int buttonstate = 0;
const int resetbtn = 13;
int repeat = 0;

void setup()
{
 pinMode(buzzer, OUTPUT);//set the pin connected to the buzzer as an output

 servo1.attach(servo1Pin);
 servo2.attach(servo2Pin);
 servo1.write(90);//Intially both doors are closed(i.e, 90 degrees)
 servo2.write(90);
```

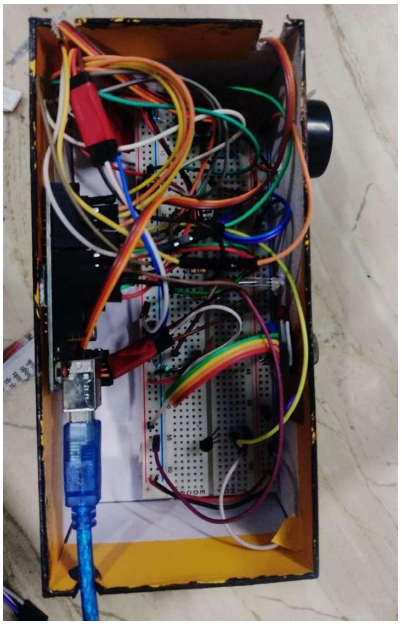
```
 pinMode(red_led,OUTPUT);
 pinMode(green_led,OUTPUT);
 pinMode(resetbtn,INPUT);
 //Serial.begin(9600);

 lcd.begin(16,2);//initialisation of 16*2 LCD
}

void loop()
{
        //for buzzer and tmp36 temp sensor
        float voltage, degreesC;
        voltage = getVoltage(temperaturePin);
        degreesC = (voltage - 0.5) * 100.0;

        sensorval=analogRead(gasSensorPin);
        //Serial.print(sensorval);
        buttonstate =
        digitalRead(resetbtn);

 if(buttonstate == HIGH) {
        repeat = 0;
 }


        if(degreesC>37 || sensorval>700 || repeat == 1)
        {
    repeat = 1;

    tone(buzzer, 800, 800);

    servo1.write(0);
    servo2.write(0);

    lcd.clear();
    lcd.setCursor(0,0);//row 0 column 0
          lcd.print("DANGER!!");
    lcd.setCursor(0,1);//row 1 column 0
     lcd.print("VACATE Building!");

    digitalWrite(red_led,HIGH);
    digitalWrite(green_led,LOW);

    delay(1000);
    tone(buzzer,600,800);
    digitalWrite(red_led,LOW);
    delay(400);
        }
        else{
    servo1.write(90);
```

```
      servo2.write(90);
      delay(1000);

      digitalWrite(green_led,HIGH);
      digitalWrite(red_led,LOW);

      lcd.clear();
            lcd.setCursor(0,0);//column 0 row 0
            lcd.print("SAFE");
      lcd.setCursor(6,0);//column 6 row 0
      lcd.print(degreesC);
      lcd.print("C");
      lcd.setCursor(0,1);
      lcd.print("Gas Conc.:");
      lcd.print(sensorval);
   }

}
float getVoltage(int pin)
{

  return (analogRead(pin) * 0.004882814);
}
```

RESULTS:

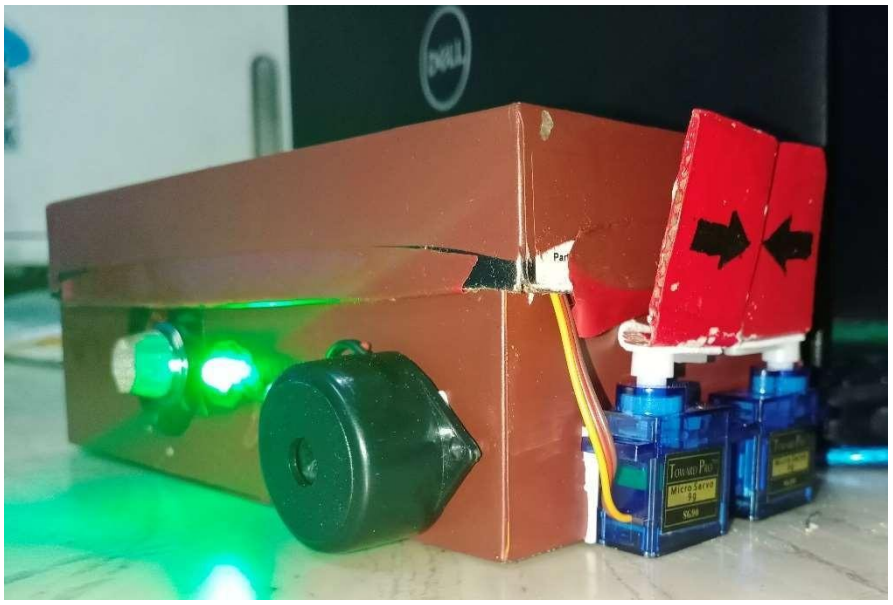
This is the final smoke and fire detector and to make it more presentable we have kept the circuit in a box with input and output openings only and hiding all the wirings.





When no smoke or Temperature increase is detected

The LED will emit green light when all the conditions are normal that is no smoke and sudden increase in temperature is noticed. No Buzzer will be activated



When Smoke and Temperature increase is detected

The LED will emit red light when it senses abnormal amount of smoke and increase in temperature. The Buzzer will also be activated and produce sound to alarm people.