

IOT BASED AIR POLLUTION MONITORING SYSTEM

V. SHRI SARVESH (108118109)
ADITHYA SINEESH (108118005)
BALAJI K S (108118021)

INTRODUCTION

- Air Pollution is the presence of substances in the atmosphere that are harmful to the health of humans and other animals. There are different types of air pollutants such as gases (Ozone, Nitrogen Dioxide, Carbon Monoxide, etc.) and particulates (PM10 and PM2.5).
- Air pollution can cause health problems, like heart attacks, strokes, diabetes and high blood pressure, that have been identified as the pre-existing medical conditions that raise the chances of death from COVID-19 infection.
- Emerging research, including a study from Harvard T.H. Chan School of Public Health, finds that breathing more polluted air over many years may itself worsen the effects of COVID-19.
- Currently, the monitoring of air quality levels is achieved by placing sensors in various locations and the sensors alert if the pollution levels exceed the threshold level.
- However, there hardly exists a mobile solution wherein an individual can contribute to the pollution level checking from the convenience of their homes.
- Moreover, solutions for forecasting gas and particulate matter levels are location specific and require expensive training data (Meteorological data, wind speeds, etc.) which is unavailable in the reach of the common people.

PROPOSED IDEA

- We aim to collect data about the concentration of NO₂, Ozone, PM2.5 and PM10 in different locations and host its visualization on a website.
- Furthermore, we also aim to predict their future concentrations using Machine Learning approaches.
- The future concentration levels that are being predicted is aimed to provide maximum accuracy based on the regions in which the system is placed.
- The proposed device can be placed anywhere in the user's home and the user can obtain real time data of the pollution levels in their immediate locality.
- The accuracy of the prediction model can be improved based on the data sensed from the user's surroundings.
- The data obtained can be also be used for further independent analysis.

PROTOTYPE DESIGNED

- Due to the unavailability of the proposed sensors to be used due to the pandemic, the current prototype designed uses sensors that we were able to procure from local shops.
- The code is designed in such a way that the other sensors, once procured, can be easily added on later.
- The Prototype collects the data about the concentration of CO, Temperature and Humidity.
- The data is visualized in real time, via a Website which is currently hosted locally.
- Real Time data can also be viewed through a Mobile App called Blynk, which is Cloud Hosted.
- Furthermore, we have trained a model that can be used to predict future concentration levels.

DESIGNED ML MODEL

```
#Function to calculate so2 individual pollutant index(si)
def calculate_si(so2):
    si=0
    if (so2<=40):
        si= so2*(50/40)
    if (so2>40 and so2<=80):
        si= 50+(so2-40)*(50/40)
    if (so2>80 and so2<=380):
        si= 100+(so2-80)*(100/300)
    if (so2>380 and so2<=800):
        si= 200+(so2-380)*(100/800)
    if (so2>800 and so2<=1600):
        si= 300+(so2-800)*(100/800)
    if (so2>1600):
        si= 400+(so2-1600)*(100/800)
    return si
data['si']=data['so2'].apply(calculate_si)
df= data[['so2','si']]
```

```
def calculate_ni(no2):
    ni=0
    if(no2<=40):
        ni= no2*50/40
    elif(no2>40 and no2<=80):
        ni= 50+(no2-40)*(50/40)
    elif(no2>80 and no2<=180):
        ni= 100+(no2-80)*(100/100)
    elif(no2>180 and no2<=280):
        ni= 200+(no2-180)*(100/100)
    elif(no2>280 and no2<=400):
        ni= 300+(no2-280)*(100/120)
    else:
        ni= 400+(no2-400)*(100/120)
    return ni
data['ni']=data['no2'].apply(calculate_ni)
df= data[['no2','ni']]
```

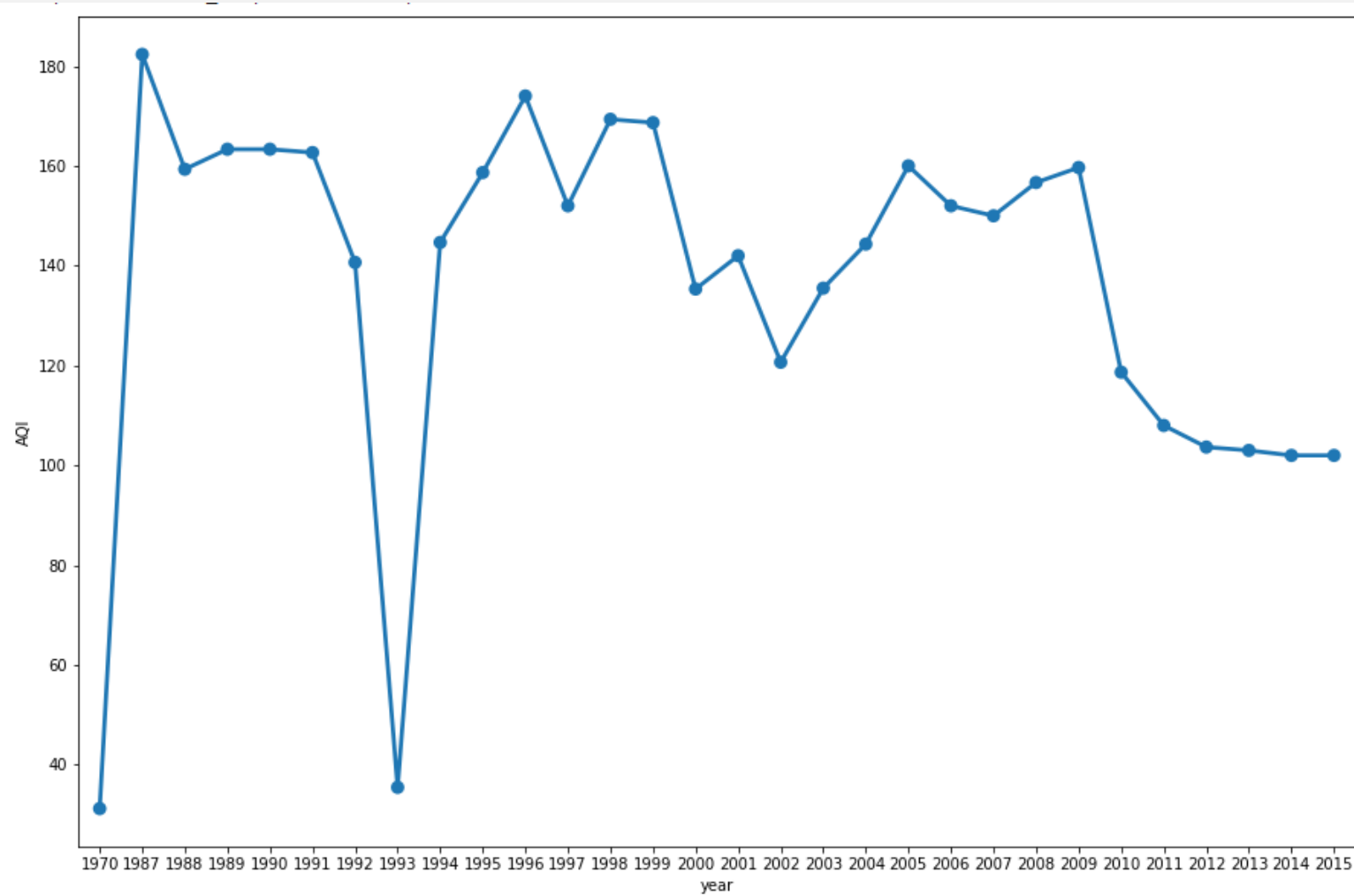
```
def calculate_rspm():
    rpi=0
    if(rpi<=30):
        rpi=rpi*50/30
    elif(rpi>30 and rpi<=60):
        rpi=50+(rpi-30)*50/30
    elif(rpi>60 and rpi<=90):
        rpi=100+(rpi-60)*100/30
    elif(rpi>90 and rpi<=120):
        rpi=200+(rpi-90)*100/30
    elif(rpi>120 and rpi<=250):
        rpi=300+(rpi-120)*(100/130)
    else:
        rpi=400+(rpi-250)*(100/130)
    return rpi
data['rpi']=data['rspm'].apply(calculate_si)
df= data[['rspm','rpi']]
```

DESIGNED ML MODEL

```
def calculate_spi(spm):
    spi=0
    if(spm<=50):
        spi=spm
    if(spm<50 and spm<=100):
        spi=spm
    elif(spm>100 and spm<=250):
        spi= 100+(spm-100)*(100/150)
    elif(spm>250 and spm<=350):
        spi=200+(spm-250)
    elif(spm>350 and spm<=450):
        spi=300+(spm-350)*(100/80)
    else:
        spi=400+(spm-430)*(100/80)
    return spi
data['spi']=data['spm'].apply(calculate_spi)
df= data[['spm','spi']]
```

```
#function to calculate the air quality index (AQI) of every data value
#its is calculated as per indian govt standards
def calculate_aqi(si,ni,spi,rpi):
    aqi=0
    if(si>ni and si>spi and si>rpi):
        aqi=si
    if(spi>si and spi>ni and spi>rpi):
        aqi=spi
    if(ni>si and ni>spi and ni>rpi):
        aqi=ni
    if(rpi>si and rpi>ni and rpi>spi):
        aqi=rpi
    return aqi
data['AQI']=data.apply(lambda x:calculate_aqi(x['si'],x['ni'],x['spi'],x['rpi']),axis=1)
df= data[['sampling_date','state','si','ni','rpi','spi','AQI']]
df.head()
```

DESIGNED ML MODEL



DESIGNED ML MODEL

```
# Applying GRADIENT DESCENT

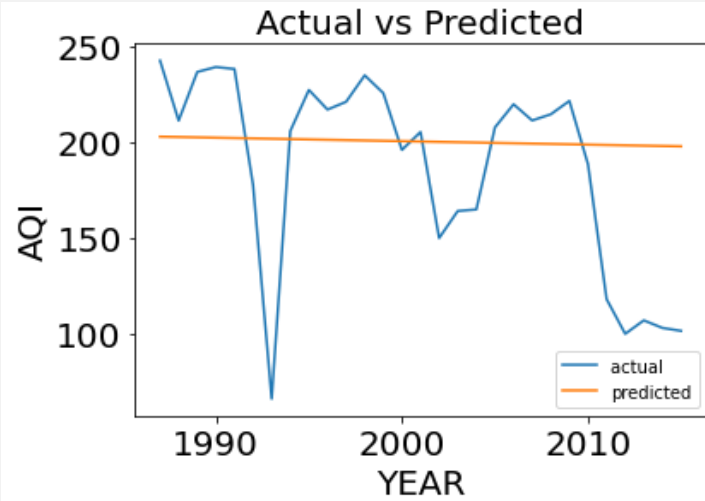
alpha = 0.1 #Step size
iterations = 3000 #No. of iterations
m = y.size #No. of data points
np.random.seed(4) #Setting the seed
theta = np.random.rand(2) #Picking random values to start with

def gradient_descent(x, y, theta, iterations, alpha):
    past_costs = []
    past_thetas = [theta]
    for i in range(iterations):
        prediction = np.dot(x, theta)
        error = prediction - y
        cost = 1/(2*m) * np.dot(error.T, error)
        past_costs.append(cost)
        theta = theta - (alpha * (1/m) * np.dot(x.T, error))
        past_thetas.append(theta)

    return past_thetas, past_costs
past_thetas, past_costs = gradient_descent(x, y, theta, iterations, alpha)
theta = past_thetas[-1]

#Printing the results...
print("Gradient Descent: {:.2f}, {:.2f}".format(theta[0], theta[1]))
```

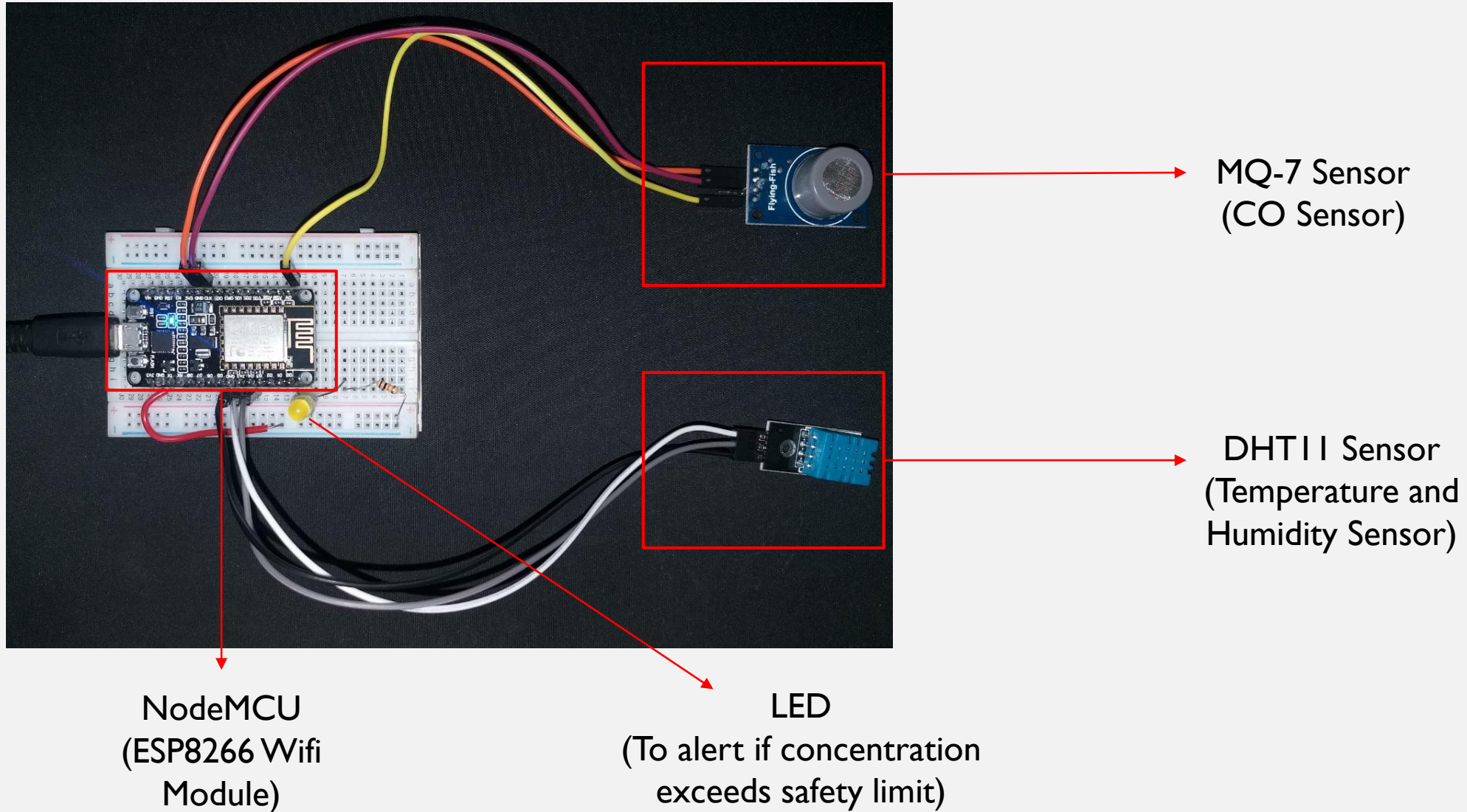

DESIGNED ML MODEL



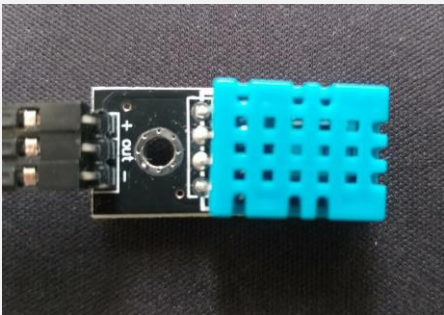
	year	AQI	Actual	Predicted
28	2015	101.258882	101.258882	197.637906
27	2014	102.785280	102.785280	197.818770
26	2013	106.729246	106.729246	197.999634
25	2012	99.696254	99.696254	198.180498
24	2011	117.824783	117.824783	198.361362
23	2010	188.283360	188.283360	198.542226
22	2009	221.368166	221.368166	198.723089
21	2008	214.378174	214.378174	198.903953
20	2007	211.160807	211.160807	199.084817
19	2006	219.623267	219.623267	199.265681
18	2005	207.546049	207.546049	199.446545
17	2004	164.661496	164.661496	199.627409
16	2003	163.875510	163.875510	199.808272
15	2002	149.706871	149.706871	199.989136
14	2001	205.138247	205.138247	200.170000
13	2000	195.772377	195.772377	200.350864
12	1999	225.439218	225.439218	200.531728

	year	AQI	Actual	Predicted
12	1999	225.439218	225.439218	200.531728
11	1998	234.740657	234.740657	200.712591
10	1997	220.903571	220.903571	200.893455
9	1996	216.850189	216.850189	201.074319
8	1995	227.102233	227.102233	201.255183
7	1994	205.636343	205.636343	201.436047
6	1993	65.754613	65.754613	201.616911
5	1992	177.485106	177.485106	201.797774
4	1991	238.060052	238.060052	201.978638
3	1990	239.071032	239.071032	202.159502
2	1989	236.513310	236.513310	202.340366
1	1988	211.076502	211.076502	202.521230
0	1987	242.438652	242.438652	202.702094

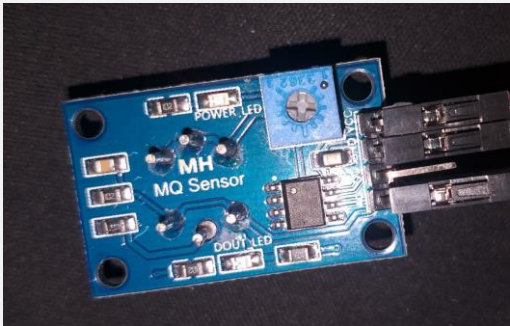
PROTOTYPE CIRCUIT



TECHNICAL DETAILS



Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50 °C	±5 %RH	±2 °C	1	4 Pin Single Row



symbol	Parameters	Technical parameters	Remark
Rs	Surface resistance Of sensitive body	2-20k	In 100ppm Carbon Monoxide
a (300/100ppm)	Concentration slope rate	Less than 0.5	Rs (300ppm)/Rs(100ppm)
Standard working condition	Temperature $-20^{\circ}\text{C} \pm 2^{\circ}\text{C}$ relative humidity $65\% \pm 5\%$ RL: $10\text{K} \Omega \pm 5\%$		
	Vc: $5\text{V} \pm 0.1\text{V}$ VH: $5\text{V} \pm 0.1\text{V}$ VH: $1.4\text{V} \pm 0.1\text{V}$		
Preheat time	No less than 48 hours	Detecting range: 20ppm-2000ppm carbon monoxide	

MQ-7 SENSOR CALIBRATION

```
/*
The coefficients are estimated from the sensitivity characteristics graph
of the MQ7 sensor for CO (Carbon Monoxide) gas by using Correlation function.
```

Explanation :

The graph in the datasheet is represented with the function

$$f(x) = a * (x ^ b).$$

where

$$f(x) = \text{ppm}$$

$$x = R_s/R_0$$

The values were mapped with this function to determine the coefficients a and b.

```
*/
```

```
#define coefficient_A 19.32
```

```
#define coefficient_B -0.64
```

```
//Load resistance 10 Kohms on the sensor potentiometer
```

```
#define R_Load 10.0
```

```
class MQ7 {
```

```
private:
```

```
uint8_t analogPin;
```

```
float v_in;
```

```
float voltageConversion(int);
```

```
public:
```

```
MQ7(uint8_t, float);
```

```
float getPPM();
```

```
float getSensorResistance();
```

```
float getRatio();
```

```
};
```

```
Function is used to return the ppm value of CO gas concentration
by using the parameter found using the function  $f(x) = a * ((R_s/R_0) ^ b)$ 
```

```
@return ppm value of Carbon Monoxide concentration
*/
```

```
float MQ7::getPPM(){
    return (float)(coefficient_A * pow(getRatio(), coefficient_B));
}
```

```
/*
```

```
This function returns voltage from the analog input value
Refer ADC Conversion for further reference
```

```
@param value : value from analogPin
```

```
@return voltage
```

```
*/
```

```
float MQ7::voltageConversion(int value){
    return (float) value * (v_in / 1023.0);
}
```

```
/*
```

```
This function is for the deriving the  $R_s/R_0$  to find ppm
```

```
@return The value of  $R_s/R_{Load}$ 
```

```
*/
```

```
float MQ7::getRatio(){
    int value = analogRead(analogPin);
    float v_out = voltageConversion(value);
    return (v_in - v_out) / v_out;
}
```

```
/*
```

```
To find the sensor resistance  $R_s$ 
```

```
@return  $R_s$  value
```

```
*/
```

```
float MQ7::getSensorResistance(){
    return R_Load * getRatio();
}
```

CODE SNIPPETS

```
Test | Arduino 1.8.13 (Windows Store 1.8.42.0)
File Edit Sketch Tools Help

Test
DHT dht(DHTPIN, DHT11);
WiFiClient client;

void setup()
{
    Serial.begin(115200);
    delay(10);
    dht.begin();

    pinMode(16, OUTPUT);
    Serial.println("Connecting to ");
    Serial.println(ssid);

    WiFi.begin(ssid, pass);

    while (WiFi.status() != WL_CONNECTED)
    {
        delay(100);
        Serial.print(" ");
    }
    Serial.println("");
    Serial.println("****WiFi connected****");
}

void loop()
{
    humi = dht.readHumidity();
    temp = dht.readTemperature();

    sensorValue = analogRead(0); // read analog input pin 0
    if (sensorValue/60 > 6) {
        // Activate digital output pin 8 - the LED will light up
        digitalWrite(ledpin, HIGH);
    }
    else {
        // Deactivate digital output pin 8 - the LED will not light up
        digitalWrite(ledpin, LOW);
    }

    delay(100); // wait 100ms for next reading
}

Leaving...
Hard resetting via RTS pin...

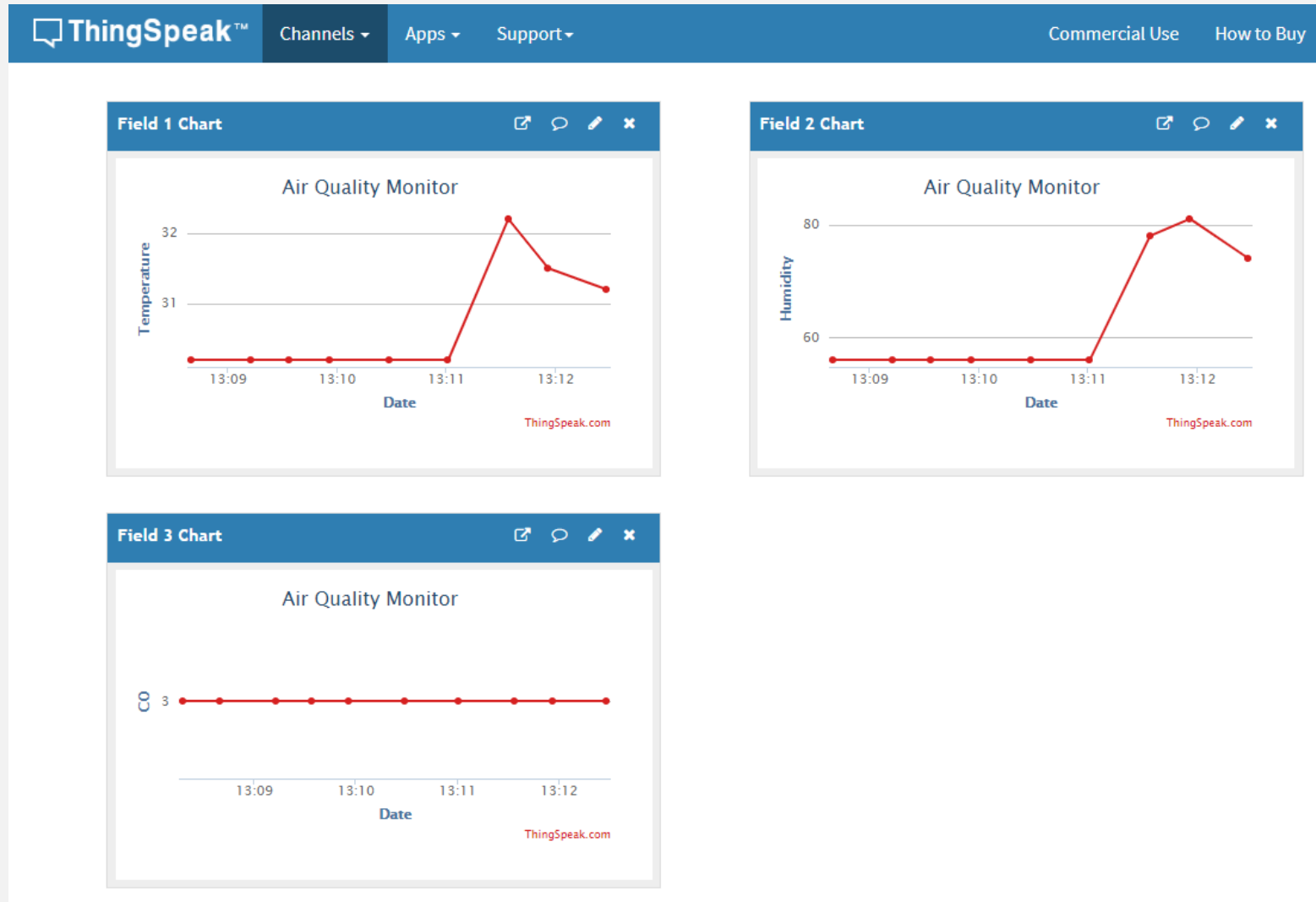
39 NodeMCU 1.0 (ESP-12E Module) on COM4
```

```
COM4
Send

Temperature: 30.20 deg. C Humidity: 59.00 % CO: 4 ppm
Connecting to Thingspeak.
Sending....
Temperature: 30.20 deg. C Humidity: 59.00 % CO: 4 ppm
Connecting to Thingspeak.
Sending....
Temperature: 30.20 deg. C Humidity: 59.00 % CO: 4 ppm
Connecting to Thingspeak.
Sending....
Temperature: 30.20 deg. C Humidity: 61.00 % CO: 4 ppm
Connecting to Thingspeak.
Sending....

☒ Autoscroll ☐ Show timestamp Newline 115200 baud Clear output
```

CODE SNIPPETS



CODE SNIPPETS

Blynk | Arduino 1.8.13 (Windows Store 1.8.42.0)

File Edit Sketch Tools Help

✓ ↻ 📄 ⬆ ⬇

Blynk

```
BlynkTimer timer;

// This function sends Arduino's up time every second to Virtual Pin (5).
// In the app, Widget's reading frequency should be set to PUSH. This means
// that you define how often to send data to Blynk App.
void sendSensor()
{
    float h = dht.readHumidity();
    float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

    if (isnan(h) || isnan(t)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }

    sensorValue = analogRead(0); // read analog input pin 0
    if (sensorValue/60 > 6) {
        // Activate digital output pin 8 - the LED will light up
        digitalWrite(ledpin, HIGH);
    }
    else {
        // Deactivate digital output pin 8 - the LED will not light up
        digitalWrite(ledpin, LOW);
    }

    // You can send any value at any time.
    // Please don't send more that 10 values per second.
    Blynk.virtualWrite(V5, t);
    Blynk.virtualWrite(V6, h);
    Blynk.virtualWrite(V7, sensorValue/60);
}

void setup()
{
    // Debug console
    Serial.begin(9600);

    Blynk.begin(auth, ssid, pass);
    // You can also specify server:
    //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);
    //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);

    pinMode(16, OUTPUT);
    dht.begin();
}
```

Air Quality Monitor

TEMPERATURE

30.300°C

HUMIDITY

76.000%

CO

5ppm

VISUALIZATION IN WEBSITE

AIR POLLUTION MONITORING SYSTEM

[Home](#)[Temperature](#)[Humidity](#)[CO](#)

AIR POLLUTION

Introduction

Air pollution is a mix of hazardous substances from both human-made and natural sources. Air pollution is a familiar environmental health hazard. We know what we're looking at when brown haze settles over a city, exhaust billows across a busy highway, or a plume rises from a smokestack. Some air pollution is not seen, but its pungent smell alerts you.

From smog hanging over cities to smoke inside the home, air pollution poses a major threat to health and climate. The combined effects of ambient (outdoor) and household air pollution cause about seven million premature deaths every year, largely as a result of increased mortality from stroke, heart disease, chronic obstructive pulmonary disease, lung cancer and acute respiratory infections.

Causes of Air pollution

Health and Environmental Effects

Even healthy people can experience health impacts from polluted air including respiratory irritation or breathing difficulties during exercise or outdoor activities. Your actual risk of adverse effects depends on your current health status, the pollutant type and concentration, and the length of your exposure to the polluted air. It can also cause other health problems including:

- Aggravated respiratory disease such as emphysema, bronchitis and asthma
- Lung damage, even after symptoms such as coughing or a sore throat disappear
- Wheezing, chest pain, dry throat, headache or nausea
- Reduced resistance to infections
- Increased fatigue and weakened athletic performance

Effects of Air pollution

VISUALIZATION IN WEBSITE

AIR POLLUTION MONITORING SYSTEM

Home

Temperature

Humidity

CO

Temperature Measurement



Temperature

Temperature is a physical quantity that expresses hot and cold. It is the manifestation of thermal energy, present in all matter, which is the source of the occurrence of heat, a flow of energy, when a body is in contact with another that is colder or hotter.

In India, temperatures typically range from -2°C to 40°C , but can reach 47°C in summer and -4°C in winter.

Temperature in India

VISUALIZATION IN WEBSITE

AIR POLLUTION MONITORING SYSTEM

Home

Temperature

Humidity

CO

Humidity Measurement



Humidity

Humidity is the concentration of water vapor present in the air. Water vapor, the gaseous state of water, is generally invisible to the human eye. Humidity indicates the likelihood for precipitation, dew, or fog to be present.

Absolute humidity describes the water content of air and is expressed in either grams per cubic metre or grams per kilogram. Relative humidity, expressed as a percentage, indicates a present state of absolute humidity relative to a maximum humidity given the same temperature

In India, humidity typically range from about 44% to 80% depending upon the region.

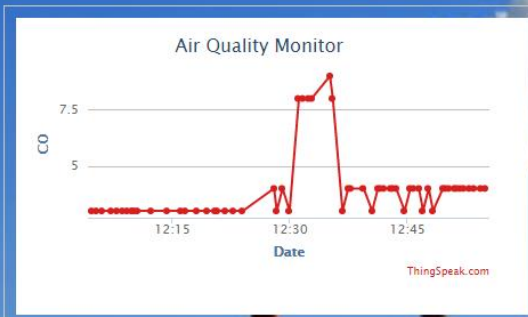
Humidity in India

VISUALIZATION IN WEBSITE

AIR POLLUTION MONITORING SYSTEM

[Home](#)[Temperature](#)[Humidity](#)[CO](#)

CO Measurement



Carbon Monoxide

Carbon monoxide (chemical formula CO) is a colorless, odorless, and tasteless flammable gas that is slightly less dense than air. It is toxic to animals that use hemoglobin as an oxygen carrier when encountered in concentrations above about 35 ppm causing carbon monoxide poisoning. Some carbon monoxide is also produced in normal animal metabolism in low quantities, and is thought to have some normal biological functions. In the atmosphere, it is spatially variable and short-lived, having a role in the formation of ground-level ozone.

Levels of carbon monoxide exposure range from low to dangerous:

- Low level: 50 PPM and less
- Mid level: Between 51 PPM and 100 PPM
- High level: Greater than 101 PPM if no one is experiencing symptoms
- Dangerous level: Greater than 101 PPM if someone is experiencing symptoms

CHALLENGES FACED

- Unavailability of the PM2.5 and SO2 sensors as they are out of stock.
- Calibration of the MQ-135 sensor to detect NO2 as according to the datasheet, the Load resistance must be around 20Kohm, whereas the flying fish module it generally comes with is only connected with a 10Kohm resistor.
- Interfacing more than one analog sensor with the NodeMCU (Interfacing Arduino UNO with the NodeMCU to get additional analog ports is currently being looked into)
- Sensor Calibration.
- Difficulty in obtaining all the required data at present to train the ML Model.

THANK YOU!