SPECTULATION DOCUMENT

NOISE AND POLLUTION DETECTOR

AIM:

- Our project aims to create a working model to detect the **Noise level** and **Air Quality Index (AQI)** at a particular location.

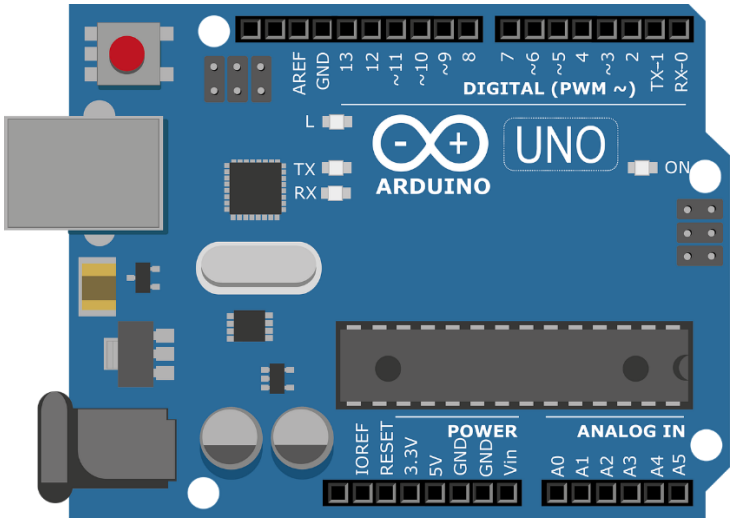
Setting Up Arduino and IDE:

Arduino IDE Setup:

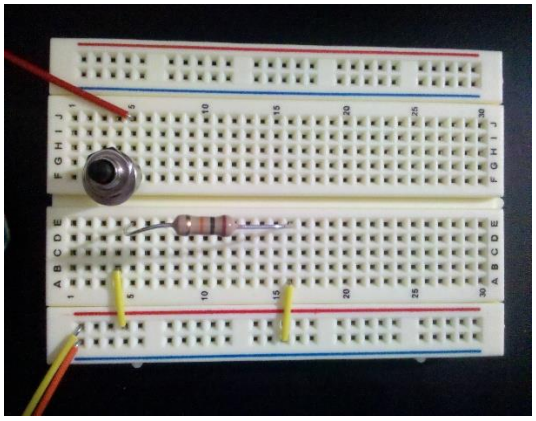
1. Go to this link https://www.arduino.cc/en/Main/Software and download Arduino IDE according to your system.
2. Extract the zip file.
3. Open the extracted folder and run Arduino.exe.
4. Your Arduino IDE is ready to use.

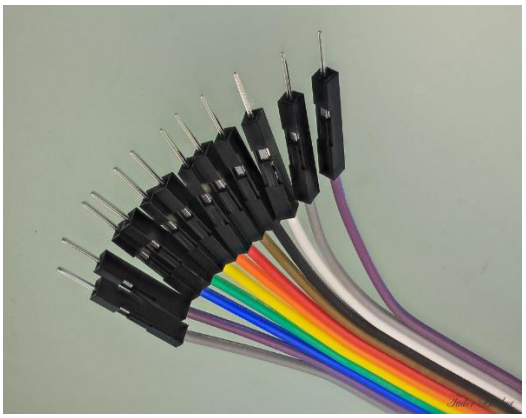
COMPONENTS REQUIRED:

- Of course, we need an **Arduino** board to set up instructions for various sensors. We have used **Arduino UNO** for our project.

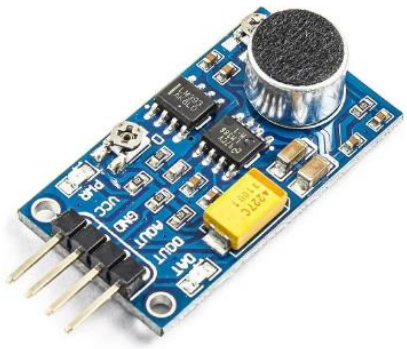


- Also, jumper wire and breadboard are required to make connections.





- We have used four sensors in our project:
  - **soundwave sensor** (to detect the sound intensity nearby the sensor). We will use the unit **dB(decibels)** to measure the sound intensity.



- For AQI calculation, we have considered three parameters, based on which we will create our working model to calculate AQI. These are **CO, Ozone and PM2.5 (**particulate matter having a size>2.5 microns are detected.).
- MQ-7 sensor for CO. (unit used: ug/m^3)

- MQ-131 for ozone. (unit used: ug/m^3)

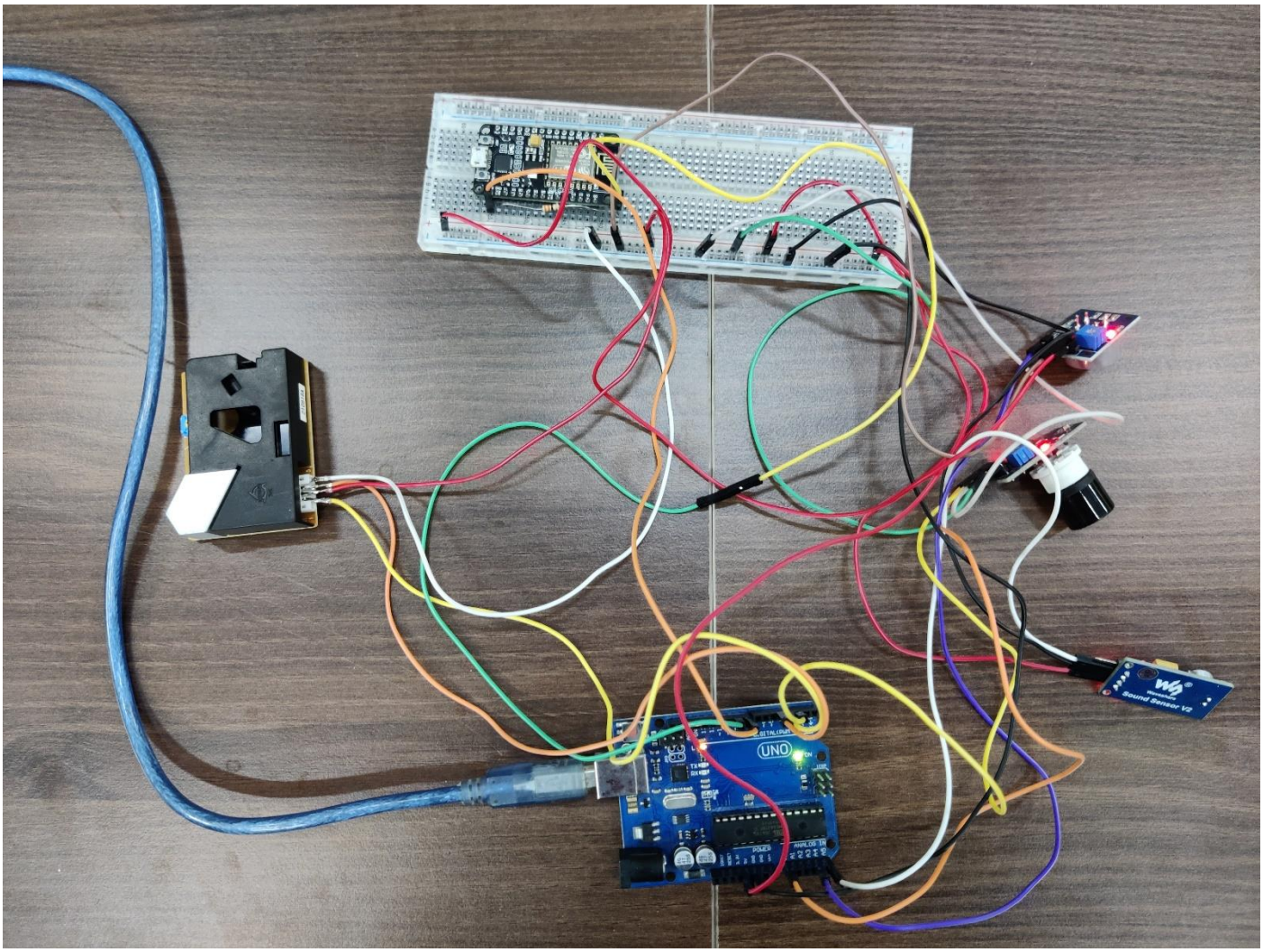
- PM2.5 DSM501A for PM2.5. (unit used: ug/m^3)

THEORY:

- Starting with the connections, firstly, we set up the Arduino UNO board's positive (Vcc) and negative (Gnd) rail on the breadboard.
- Next is the soundwave sensor; connections are:
  - The Vcc to the positive rail.
  - Gnd to the negative rail.
  - Aout to the A5 of Arduino board.
- Next is the CO(MQ7) sensor; connections are:

- The Vcc to the positive rail.
- Gnd to the negative rail.
- Aout to the A4 of Arduino board.

- Next is the O3(MQ131) sensor; connections are:
  - The Vcc to the positive rail.
  - Gnd to the negative rail.
  - Aout to the A0 of Arduino board.
  - Dout to the pin D2 of Arduino board.
- Lastly, we have a PM2.5 DSM501A sensor; connections are:
  - The Vcc to the positive rail.
  - Gnd to the negative rail.
  - PM2.5 Dout to the pin D4 of Arduino board.
  - PM1.0 Dout to the pin D3 of Arduino board.
- We are now done with the connections; we can see an overview of the circuit in the below image.



Next, we will see the implementation of our code:

```
#include <MQ131.h> //Ozone sensor library

#define DUST_SENSOR_PIN_PM10  3  //Must be the pins that
#define DUST_SENSOR_PIN_PM25  4  //support interrupts

#define INTERVAL_COUNTDOWN 1000
#define INTERVAL_READ 30000

#include <KarserDSM501.h>       // ISRs forward declaration
void pm10_handleInterrupt();
void pm25_handleInterrupt();    // init pm10 and pm25 instances

KarserDSM501 pm10(DUST_SENSOR_PIN_PM10, pm10_handleInterrupt);
KarserDSM501 pm25(DUST_SENSOR_PIN_PM25, pm25_handleInterrupt); // handle ISRs
void pm10_handleInterrupt() { pm10.handleInterrupt(); }
void pm25_handleInterrupt() { pm25.handleInterrupt(); }


unsigned long timer = 0;

const int AOUTpin=A4;    // CO sensor AOUT pin
const int AOUTpin1=A5;   // Sound sensor AOUT pin

int conc_O3;     // ozone concentration
```

```
int conc_CO;     // carbon monoxide concentration
int pm2_5;        // (particulate matter)pm2.5
int level_dB;    // sound level in decibal

// defining lower and upper limit for each pollutant
int Lo3,Uo3,Lpm,Upm,Lco,Uco,AQI_L,AQI_U;

String quality;

void setup() {
  Serial.begin(115200);

  pinMode(AOUTpin, INPUT);// set Aoutpin of CO sensor as input
  pinMode(AOUTpin1, INPUT);// set Aoutpin of Sound sensor as input

  MQ131.begin(2,A0, LOW_CONCENTRATION, 1000000);
  Serial.println("Ozone Calibration...");
  MQ131.calibrate();
  Serial.println("Calibration Done");
  Serial.println();
}

void loop() {
```

```
void loop() {

   Serial.println("Sampling...");
   MQ131.sample();
   conc_O3= MQ131.getO3(UG_M3); // gives the output in micro gram/meter cube
   Serial.print("O3 level= ");
   Serial.print(conc_O3);
   Serial.print("ug/m3");

   conc_CO= 1000*analogRead(AOUTpin);//reads the analaog value from the CO sensor's AOUT pin

   Serial.print("\nCO level= ");

   Serial.print(conc_CO);
   Serial.println("ug/m3");

   level_dB= map(analogRead(AOUTpin1),0,1023,0,52); //mapping sensor input value to 0-52 range
   Serial.print("Sound level= ");
   Serial.print(level_dB);
   Serial.println("dB");


   if (!pm10.isReady() && (millis() >= timer + INTERVAL_COUNTDOWN)) {
```

```
  if (!pm10.isReady() && (millis() >= timer + INTERVAL_COUNTDOWN)) {
    timer += INTERVAL_COUNTDOWN;
  } else if (millis() >= timer + INTERVAL_READ) {
    timer += INTERVAL_READ;
    pm2_5=1000*pm25.readPM();
    Serial.print("pm2.5: "+String(pm2_5));
     Serial.println("ug/m3");
  }

// AQI calculation for CO
  if( conc_CO>0 && conc_CO<=1000){          //if conc. of CO is 0 < CO <= 1000ug/m3 then set lower limit =0 and upper limit = 50
     Lco=0;
     Uco=50;
    }
  else if(conc_CO>1000 && conc_CO<=2000){   //if conc. of CO is 1000 < CO <= 2000ug/m3 then set lower limit =51 and upper limit = 100
     Lco=51;
     Uco=100;
    }
  else if(conc_CO>2000 && conc_CO<=10000){  //if conc. of CO is 2000 < CO <= 10000ug/m3 then set lower limit =101 and upper limit = 200
     Lco=101;
     Uco=200;
    }


  else if(conc_CO>10000 && conc_CO<=17000){ //if conc. of CO is 10000 < CO <= 17000ug/m3 then set lower limit =201 and upper limit = 300
     Lco=201;
     Uco=300;
    }
  else if(conc_CO>17000 && conc_CO<=34000){ //if conc. of CO is 17000 < CO <= 34000ug/m3 then set lower limit =301 and upper limit = 400
     Lco=301;
     Uco=400;
    }
  else if(conc_CO>34000){                   //if conc. of CO is  greater than 34000ug/m3 then set lower limit =401 and upper limit = 500
     Lco=401;
     Uco=500;
    }

// aqi calculation for O3
  if( conc_O3>=0 && conc_O3<=50){
     Lo3=0;
     Uo3=50;
    }
  else if(conc_O3>=51 && conc_O3<=100){
     Lo3=51;
     Uo3=100;
    }
  else if(conc_O3>=101 && conc_O3<=168){
     Lo3=101;
```

```
    Uo3=200;
   }

  else if(conc_O3>=169 && conc_O3<=208){
    Lo3=201;
    Uo3=300;
   }

  else if(conc_O3>=209 && conc_O3<=748){
    Lo3=301;
    Uo3=400;
   }

  else if(conc_O3>748){
    Lo3=401;
    Uo3=500;
   }


// aqi calculation for PM2.5
  if( 1000*pm25.readPM()>0 && 1000*pm25.readPM()<=30){
    Lpm=0;
    Upm=50;
   }
```

```
  else if(1000*pm25.readPM()>=31 && 1000*pm25.readPM()<=60){
    Lpm=51;
    Upm=100;
   }
  else if(1000*pm25.readPM()>=61 && 1000*pm25.readPM()<=90){
    Lpm=101;
    Upm=200;
   }
  else if(1000*pm25.readPM()>=91 && 1000*pm25.readPM()<=120){
    Lpm=201;
    Upm=300;
   }
  else if(1000*pm25.readPM()>=121 && 1000*pm25.readPM()<=250){
    Lpm=301;
    Upm=400;
   }
  else if(1000*pm25.readPM()>250 ){
    Lpm=401;
    Upm=500;
   }

// now we are comparing the upper limit of each pollutant
// and we will store the one which has highest upper limit
  if(Uco>Uo3){

  if(Uco>Uo3){
    AQI_L=Lco;
    AQI_U=Uco;
   }
  else{
    AQI_L=Lo3;
    AQI_U=Uo3;
   }
  if(Upm>AQI_U){
    AQI_L=Lpm;
    AQI_U=Upm;
   }

  if( AQI_L==0 && AQI_U==50){
    quality="GOOD";
   }
  else if(AQI_L==51 && AQI_U==100){
    quality="SATISFACTORY";
   }
  else if(AQI_L==101 && AQI_U==200){
    quality="MODERATE";
   }
  else if(AQI_L==201 && AQI_U==300){
    quality="POOR";
```

```
  else if(AQI_L==201 && AQI_U==300){
    quality="POOR";
   }
  else if(AQI_L==301 && AQI_U==400){
    quality="VERY POOR";
   }
  else if(AQI_L==401 && AQI_U==500){
    quality="SEVERE";
   }

  Serial.print("AQI is :");
  Serial.print(AQI_L);
  Serial.print(" - ");
  Serial.println(AQI_U);
  Serial.print("Air Quality: ");
  Serial.println(quality);
  Serial.println();
  delay(60000);

}
```

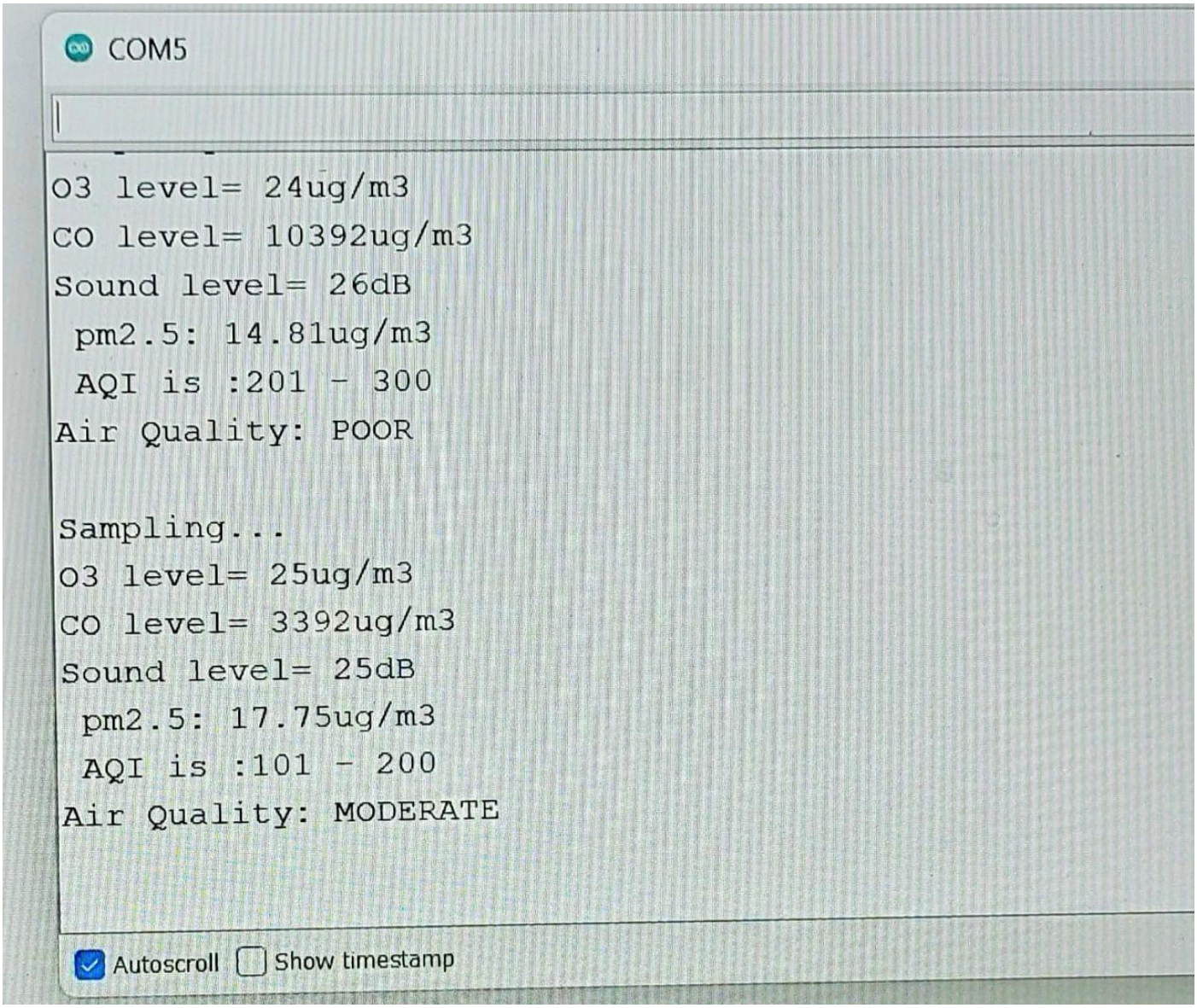
After the connections, as we upload the code from the Arduino IDE to the Arduino board, we will now see the working principle of the code.

- **Sound sensor**: this sensor senses the sound created near the sensor and converts it into the electric signal in digits 0-1023 on the serial monitor. We have mapped this range with 0-52 dB through the unitary method. So whenever a value is sent by the sensor like 523, a sound intensity of (52/1023) x 523 dB is created.

- **CO(MQ7) sensor**: this sensor detects the concentration of the gas CO in the atmosphere. As we have used the unit to be ug/m^3, it displays the CO concentration in ug/m^3 on the serial monitor by exact unitary mapping.
- **O3(MQ131) sensor**: this sensor detects the concentration of the gas O3 in the atmosphere, and as we have used the unit to be ug/m^3, it displays the O3 concentration in ug/m^3 on the serial monitor by exact unitary mapping.
- **PM2.5 DSM501A sensor**: this sensor detects the concentration of the particulate matter(above 2.5-micron size) in the atmosphere, and as we have used the unit to be ug/m^3, it displays the PM2.5 concentration in ug/m^3 on the serial monitor by exact unitary mapping.
- Now, the sound sensor reading is directly printed on the serial monitor as soon as we run the code.
- To measure the pollution level, we have mapped each value with a website's data to calculate the individual AQI; out of the three AQI, the worst one (means the most outstanding value) will be the output AQI of the device. According to the AQI, we also have mentioned if the output AQI is sound, moderate, poor, severe etc.

- We have used the simple logic of calculating the individual AQI according to the official data; if the concentration of a specific pollutant lies in a particular range, the lower and upper limit of that AQI is decided.
- We then compared the upper limits of the three individual AQI; the maximum one gives the max AQI, we print this AQI, and the corresponding remark of the pollution level is also printed.
- We have used the data from the following website:

  https://www.pranaair.com/blog/what-is-air-quality-index-aqi-and-its-calculation/



Helpful links:

- https://learn.sparkfun.com/tutorials/what-is-an-arduino
- https://www.pranaair.com/blog/what-is-air-quality-index-aqi-and-its-calculation/
- Here is our project Github Repo : https://github.com/kumaryash7/Noise-and-Pollution-Detection-System

Challenges Faced:

- Integrating the code for all the sensors was a bit challenging; giving delays at appropriate positions and removing the repetitive statements from different codes was something to be kept in mind while integrating the code.
- While we were running the CO sensor code, initially, it gave some absurd values; we found out that it was because we connected the Vcc with the 5V output; we then joined the Vcc with 3.3V, and then it started working correctly.
- We know that we need to import libraries to work with a particular sensor; for the PM2.5 sensor, the library was not readily available on the internet.
- In the sound sensor output, we may see a significant variation in the consecutive values because of the unnecessary noises and interferences in the surroundings.
- We were constrained to use only some parameters to calculate the AQI, so the calculated AQI might not be accurate, but our main task was to create a working model considering the given parameters; according to that, we have worked on our device.

OUR MENTOR

Shrey Agrawal Sir

OUR TEAM

Prateek Verma

Ayush Arya

Yash Kumar

Sumit Kumar