North South University

Department of Electrical & Computer Engineering

IoT Based Air Pollution Monitoring & Prediction System

CSE299 – Junior Design Course

Submitted By: *Name:* *ID:*

	Name	ID
1)	**Mohammed Rakib**	**1731176042**
2)	**Sanaulla Haq**	**1721380042**
3)	**Ismail Hossain**	**1722018042**

Submitted To: **Dr. Tanzilur Rahman**
Assistant Professor; ECE Department
North South University

Date of Submission: **20th May, 2020.**

TABLE OF CONTENTS

ABSTRACT

Air pollution has reached catastrophic levels in recent times and is increasing at an alarming rate. To mitigate this problem, public awareness and combined efforts from every individual is a must. For this reason, we have come up with a project that will not only detect pollution levels in the atmosphere accurately but also predict the future pollutant levels. Besides, it will also state the health impacts of long-time exposure to pollution. To implement our project, we are using four different sensors along with a microcontroller to process the data from the sensors. Next, a Wi-Fi module is used to send all the parameter data to a cloud server. From there, data is fetched and further analyzed for selecting the best model for prediction. Finally, various experiments are performed to test the integrity of our dataset and the accuracy of our model in forecasting air pollutant values accurately.

Chapter 1 **INTRODUCTION**

1.1 Overview

We live in a world where the word pollution has become a part and parcel of our daily life. Pollution has now become a serious threat to our existence and healthy living. Every day we are degrading our environment by causing various forms of pollution. Air, water, soil – the three natural resources we cannot live without is constantly being contaminated. Air pollution is one of the leading concerns of the modern world. It may be defined as the presence of harmful particles in the atmosphere that affects human life as well as the natural environment. About 8 million people die each year due to air pollution [1]. Think tanks all around the world are working together to raise awareness levels and reduce the extent of air pollution.

Air is the very thing we breathe and cannot live without even for a minute. So, it's our duty & responsibility to keep the air clean not only for our sake but also for the life around us. For this reason, in this project, we would like to build an IoT device that can monitor and predict the air quality around us. We will be using four different sensors to track the air quality around us. PM (Particulate Matter), CO (Carbon monoxide), NH_3 (Ammonia), and Temperature & Humidity all will be measured. Next, the data will be processed by a microcontroller and sent to a server with the help of a Wi-Fi module. Finally, we will show the user the concentrations of the gases in ppm (parts per million), overall air quality by using AQI (Air Quality Index) and future air quality by using machine learning models.

Figure 1: Air Pollution & Its Health Impacts

1.2 Air Pollution

1.2.1 Definition & Types

When the concentration of harmful gases (CO, CO_2, NO_2, SO_2, and O_3) and particles (Particulate Matter) exceeds the normal limit and become hazardous to all living things, we call it air pollution. There are two types of air pollution. Indoor pollution & Outdoor pollution.

1.2.2 Cause & Effects of Air Pollution

Air pollution is caused by various reasons. Vehicle emissions, exhaust from factories and industries, burning of fossil fuels, forest fires, mining natural resources, using chemicals in agriculture, smoking cigarettes, and volcanic eruptions – all cause air pollution. These have unhealthy effects on humans as well as the environment. Long term exposure to polluted air causes various lung and heart-related diseases along with cancer. Besides, the rise of sea level, depletion of the ozone layer, climate change, global warming are all directly related to air pollution. For this reason, we want to raise awareness levels by building our project.

1.3 IoT

IoT or Internet of Things is a communication system through which multiple devices interact with each other to share data over the internet without any manual human interaction. It is one of the major aspects of the fourth industrial revolution. The application of IoT is endless. Automatic traffic control, smart home, smart cities, smartwatch, smart farming, smart grids, and connected healthcare are some noteworthy examples of IoT. In our project, we are using IoT to send the data from sensors to a server to visualize it and then to a desktop environment to build predictive models for the dataset. The entire process will be seamless without any human intervention. Besides the power consumption is also quite low while implementing IoT. For this reason, we have opted to use IoT in our project.

1.4 ML

ML is a part of AI that allows computer systems to use patterns and algorithms to analyze the data fed to it and give better results by learning without being explicitly told to do so. It is the current big thing in modern science. ML has widespread applications. From virtual assistants like Siri, Bixby to self-driving cars – ML has established its dominance everywhere. In our project, we will be using ML models to detect if the air is polluted or not. We will also use it to predict future $PM_{2.5}$, CO & NH_3 levels. Based on the data from our sensors, the ML model with the highest accuracy will be used for detecting pollution status and predicting pollutant levels of a particular date.

1.5 Aims & Objectives

As air pollution is ever-increasing and ringing alarm bells everywhere, now is the perfect time to do something to make people aware of the consequences especially in a developing country like ours. For this reason, we have focused on delivering a project that can effectively control and reduce the extent of air pollution. The following are the aims & objectives of our project:

- We aim to build a device that can accurately monitor various pollutants (carbon dioxide, carbon monoxide, particulate matter) and comments on the air quality.
- We also aim to build a system that will allow the users to monitor the air quality in real-time from any part of the world over the internet.
- Our system will also predict future pollutant levels based on machine learning models.

1.6 Motivation

Air pollution is one of the prime reasons for financial losses & deaths every year around the world. In our country, nearly 28% of deaths occur every year due to air pollution [2]. Moreover, we incur losses of about 14 billion USD every year due to air pollution [3]. Besides, clean air for breathing is the fundamental right of every citizen. According to the SDG (Sustainable Development Goals) by the UN (United Nations), there are 17 goals for peace and prosperity of our planet. Among them, Goal 3: Ensure healthy lives and promote well-being for all, Goal 13: Take immediate action to combat climate change and its impacts and Goal 15: Manage deforestation, land degradation, biodiversity loss and desertification in a sustainable manner - all these have inspired us in doing the project. All of the above reasons have encouraged us to make an air quality device that can not only precisely comment about the air quality but also predict future pollutant levels so that people can be conscious about the harmful effects of inhaling polluted air and ultimately reduce pollution levels.

Chapter 2 **BACKGROUND STUDY**

2.1 Overview

Before starting our project in full swing, we gathered some knowledge about the current situation of air quality monitoring devices. We looked at numerous journal articles, reports, and research papers regarding the topic and will review three of them over here. Moreover, we thoroughly went through the topics on how to calculate air quality and the parameters required for it. In this chapter, we will be comparing others' work with our project and how we aim to make a better device. Besides, we will also explain the index of air quality and how it is calculated.

2.2 Parameters Reflecting Air Quality

Although there are many hazardous particles in the air, six main pollutants which are considered to be fatal for our health are:

2.2.1 $PM_{2.5}$ & PM_{10}

Particulate matter is a mixture of small solid and liquid particles that remain suspended in the air for long periods. It is classified into two types based on size for better emission control and mitigation strategies. They are $PM_{2.5}$ (particles within the range 0-2.5 microns in size) and PM10 (particles less than or equal to 10 microns in size). Previously, PM10 has been a major concern due to the ease with which it could enter the lungs. But recent studies show that $PM_{2.5}$ is more dangerous as it can enter the deepest passage of our lungs and cause various disorders. Particulate matter contains toxic chemicals that can cause cancer and various respiratory disorders.

2.2.2 CO

Carbon monoxide is the end product of incomplete combustion of gasoline and diesel from various vehicles. It reduces the capacity of blood to carry oxygen, alters the nervous system, and causes headaches, fatigue, drowsiness, respiratory failure, and even death.

2.2.3 NO_2

Nitrogen dioxide is the result of high-temperature combustion processes in vehicles and electrical storms. It is the main cause of lung problems and respiratory diseases in humans. Besides, it causes premature leaf loss and prevents growth in plants.

2.2.4 SO_2

SO_2 is the product of the combustion of gasoline and diesel fuels, burning of fossil fuels for generating electricity, emissions from factories, and volcano eruption. It combines with water to form sulfuric acid which is the main component of acid rain. SO_2 causes discomfort in the eyes & irregularities respiratory system and worsens respiratory diseases such as asthma and chronic bronchitis.

2.2.5 O_3

Tropospheric ozone resides in the ground level of the atmosphere and is the main component of smog. It is formed due to the chemical reaction between NOX and VOC emitted by vehicles, refineries, and power plants. Ground-level ozone causes various problems like asthma, inflammation of lungs, a temporary reduction in lung capacity, and weakening of the immunity system.

2.3 Reviewing Other's Work

Here we will review three papers relating to air pollution and try to compare them with our project for improvement.

2.3.1 Sensing Data Fusion for Enhanced Indoor Air Quality

In this project, they have focused on building an accurate system for measuring IAQ (Indoor Air Quality). For this, they have used a Waspmote Sensor and Meshlium gateway router. The Waspmote sensor is a combination of different sensors for measuring temperature, humidity, CO_2, CO, NH_3, H_2, O_2, ethanol, hydrogen sulfide, and toluene. And the router is used to connect the IoT enabled Waspmote sensor to the cloud through Ethernet, Wi-Fi, or 3G services. Moreover, they have combined Humidex and IAQI (Indoor Air Quality Index) to form EIAQI (Enhanced Indoor Air Quality Index) to take humidity into account while calculating IAQ [4]. To improve the accuracy of predicting air pollutant parameters, they have used EFKF (Extended Fractional-order Kalman Filter) model since it can handle spatial distributions and highly non-linear uncertain property of indoor air quality data properly.

2.3.1.1 Comparison with Our Project

In our project, we are calibrating and combining gas sensors manually whereas in the above project they have used an already manufactured sensor system. They used a combination of 10 sensors but left out a key parameter which is the $PM_{2.5}$ sensor. On the other hand, our project has only 4 sensors but includes the $PM_{2.5}$ sensor since it plays a key role in determining air quality. We are using ARIMA (Autoregressive Integrated Moving Average) as our model for prediction since it suites our data whereas they have used EFKF model to fuse two indexes and create a novel index as well as predict the air pollutant parameters Lastly, they have fused IAQI and humidex to include humidity in their air quality index whereas we have used the general AQI which can include any no. of parameters.

2.3.2 MegaSense: Cyber-Physical System for Real-time Urban Air Quality Monitoring

In this project, they have mainly focused on reducing the exposure to air pollution of urban citizens and make a difference in their related health impacts in a coordinated manner. For this, they have used pollutant data from a wide array of sources. From external systems like government meteorological stations, road traffic data, wind vector data, and weather stations to their personally customized HOPE sensor for an individual - they have taken into account all possible forms of pollutant data through the input API. Next, all these data are cleaned, processed, and analyzed in the cloud. Finally, the personal air pollution exposure of a user as well as the entire city pollution exposure map is displayed via the Exposure App.

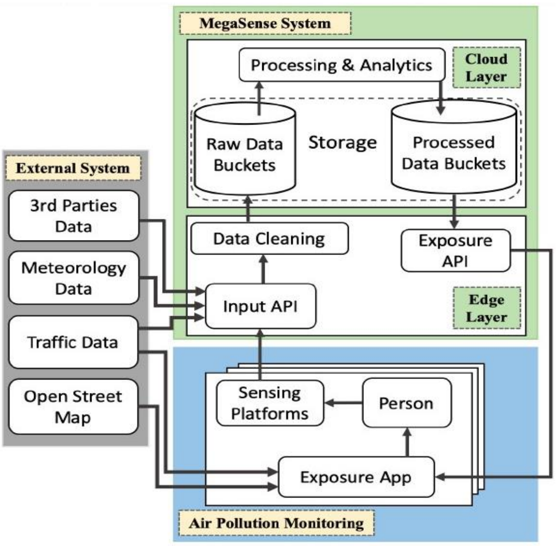


Figure 2: Block Diagram of MegaSense Cyber-Physical System [5]

2.3.2.1 *Comparison with Our Project*

In the project mentioned above, they have implemented a large scale wide-area based pollution monitoring system where their main goal was to reduce the personal exposure to air pollution (both indoor & outdoor) of an individual by suggesting route maps with less pollution and warning the user of high levels of pollution. On the other hand, we are building a weather station that can detect the pollutant levels of a particular area and can also predict future pollution parameters accurately.

2.3.3 Development of an IoT-Based Indoor Air Quality Monitoring Platform

In this project, they have built a system called Smart-Air that can accurately measure the indoor air quality of an area. For this, they have used 5 sensors in the device namely laser dust sensor, smoke sensor, CO sensor, a CO_2 sensor, and temperature & humidity sensor. Next, all these sensors were connected to a microcontroller to send data to the web server via an LTE modem. Whenever the air quality changes, the device changes its color as well as sends an alert message to the webserver. Lastly, they have also built an app to better visualize the indoor air quality.

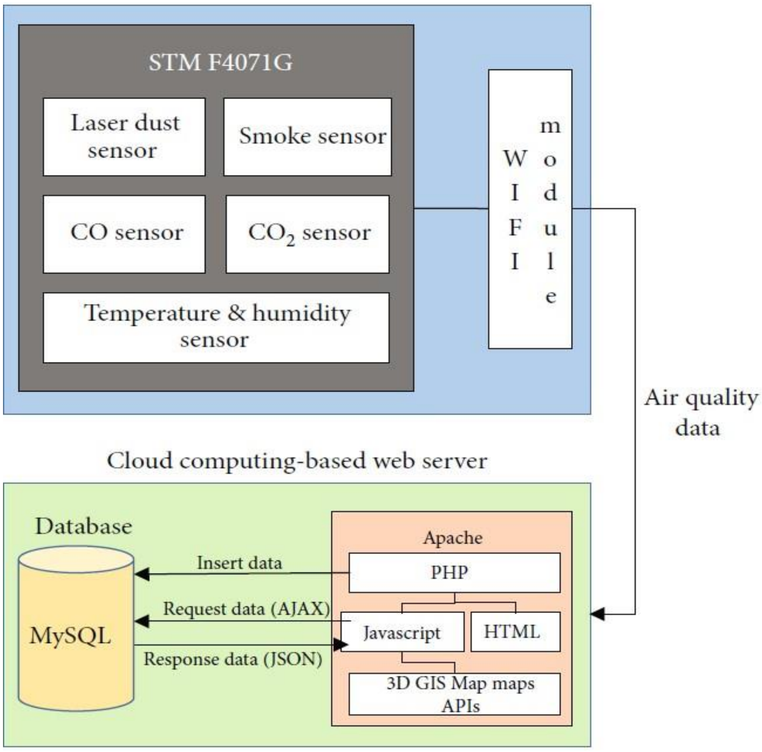


Figure 3: Configuration diagram for the IoT based indoor air quality monitoring platform [6]

2.3.3.1 Comparison with Our Project

The main focus of their project was to monitor indoor air quality and determine its accuracy which they accomplished. In our case, we are also building a similar monitoring system that can monitor air quality (indoor and outdoor) with various sensors. Additionally, we are using the ARIMA model to train all of our parameter data so that apart from measuring pollution levels it can also accurately predict future pollutant levels.

2.4 AQI

2.4.1 What is AQI?

The air quality index is a color-coded numeric system that indicates the quality of air and its health effects in a way that can easily be understood by the general public. It was first introduced in 1968 to make people aware of the bad effects of exposure to polluted air. Generally, the index ranges from 0 to 500. Different countries depict AQI with different ranges and parameters depending upon the environment of that particular area. Since our country has no dedicated AQI, we would be using the index of our neighboring country India.

2.4.2 AQI Table

The breakpoints & AQI range of pollutants according to Indian AQI is given below:

Table 1: Breakpoints of different pollutants in IND-AQI (CPCB, 2014) [7]

AQI Category (Range)	PM_{10} (µg/m^3)	$PM_{2.5}$ (µg/m^3)	NO_2 (µg/m^3)	O_3 (µg/m^3)	CO (mg/m^3)	SO_2 (µg/m^3)	NH_3 (µg/m^3)	Pb (µg/m^3)
Good(0-50)	0–50	0–30	0–40	0–50	0–1.0	0–40	0–200	0–0.5
Satisfactory(51-100	51–100	31–60	41–80	51–100	1.1–2.0	41–80	201–400	0.5–1.0
Moderately polluted(101-200)	101–250	61–90	81–180	101–168	2.1–10	81–380	401–800	1.1–2.0
Poor(201-300)	251–350	91–120	181–280	169 –208	10–17	381–800	801–1200	2.1–3.0
Very poor(301-400)	351–430	121–250	281–400	209–748	17–34	801–1600	1200–1800	3.1–3.5
Severe (401–500)	430+	250+	400+	748+	34+	1600+	1800+	3.5+

2.4.3 AQI & Health Impacts

The AQI ranges and its respective health impacts are given in the table below:

Table 2: Associated health impacts of AQI levels [7]

AQI	Associated Health Impacts
Good (0–50)	No health issues.
Satisfactory (51–100)	Breathing problem for sensitive people.
Moderately polluted (101–200)	Breathing discomfort to people with lung diseases like asthma, and irritation to people with heart diseases, children, and older adults.
Poor (201–300)	Breathing discomfort to people on continuous exposure, and irritation to people with heart disease.
Very poor (301–400)	Respiratory disorders to the people on continuous exposure. The effect is more fatal in people with lung and heart diseases.
Severe (401–500)	It causes respiratory problems even on healthy people, and serious health issues on people with lung/heart disease.

2.4.4 How to Calculate AQI

The following equation is used to convert from concentration to AQI [8]:

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low} \quad \text{; where,}$$

I = the Air Quality Index,

I_{low} = AQI value corresponding to C_{low},

I_{high} = AQI value corresponding to C_{high},

C_{low} = the breakpoint concentration that is $\leq C$,

C_{high} = the breakpoint concentration that is $\geq C$,

C = the pollutant concentration.

2.4.5 Example of AQI

The 24-hr $PM_{2.5}$ concentration of an area is *31 μg/m³*. Calculate its corresponding AQI.

We know,

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}}(C - C_{low}) + I_{low}$$

$$I = \frac{100 - 51}{60 - 31}(31 - 31) + 51$$

(Taking values from AQI table)

$$I = 0 + 51$$

$$I = 51$$

So the AQI is 51 which is in the satisfactory range (51-100).

2.5 Summary

In this chapter, we have explained about the parameters that reflect air quality. Moreover, we have reviewed three project works and discussed their advantages and disadvantages. We have also compared them with our project and how we could improve our project by taking ideas from them. Lastly, we have discussed about AQI, how to calculate AQI, which country's index will we be following, and what are the health impacts of different AQI levels.

Chapter 3 **METHODOLOGY**

3.1 Overview

In this chapter, we will explain how everything is done. It contains the design of our project which has been depicted using a block diagram. Besides, it shows the list of all the hardware and software used. It also broadly discusses the theories of the hardware and software and how it is implemented. Moreover, this chapter not only shows the software related codes but also how the relationship between software and hardware is created. Finally, this chapter explains in detail about the machine learning model used for prediction by data preprocessing and analyzing different prediction models.

3.2 System Diagram

In our project, we have selected Arduino Mega 2560 with an ATmega 2560 processor as our microcontroller. We have connected 4 sensors (2 gas, 1 dust, and 1 temperature & humidity) with the microcontroller. These sensors send digital signals in the form (0-1023) to the Arduino board by converting the analog voltage (0V-5V) variations. Arduino has its IDE and its programming language, which is similar to the programming language C/C++. In Arduino IDE, we wrote the necessary codes and uploaded them into our Arduino board to fetch the data from the sensors. Next, after setting up the Wi-Fi module, our data goes to the real-time Cloud Storage (ThingSpeak). From there using the read API, data is fetched for preprocessing. After that, various models were evaluated, and the best model that suites our dataset i.e. ARIMA (Autoregressive Integrated Moving Average) was selected. Then we train our model to predict the future pollutant levels of all the parameters (CO, NH_3, temperature, humidity & $PM_{2.5}$). Finally, various experiments were held to check the integrity of our sensor data and accuracy of our prediction model which are explained in the experiment and results section.

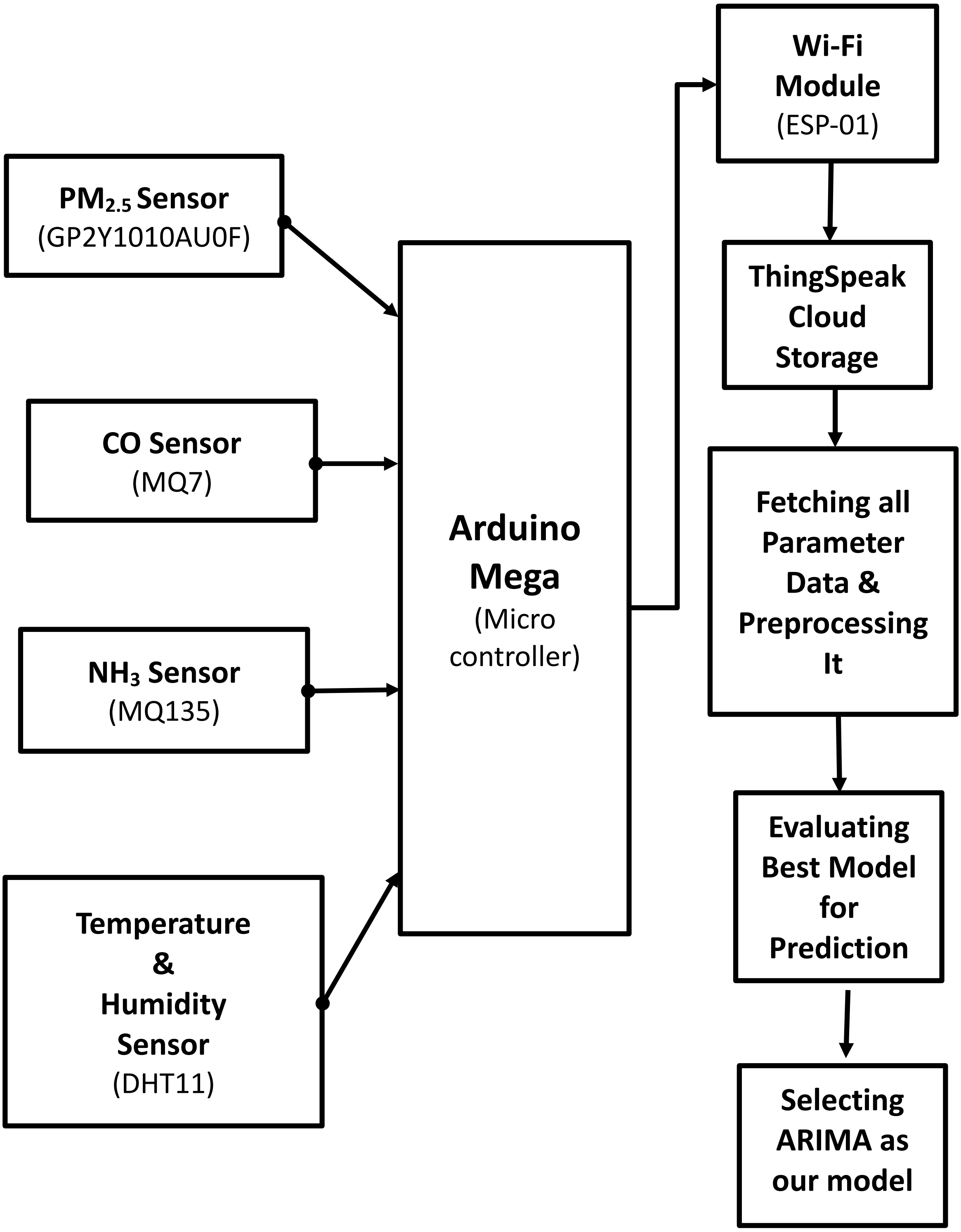


Figure 4: Entire System Diagram of the Project

3.3 Hardware & Software Requirements

- Arduino Mega 2560 Microcontroller
- Temperature & Humidity Sensor (DHT11)
- CO Sensor (MQ-7)
- NH_3 Gas Sensor (MQ-135)
- Optical $PM_{2.5}$ Dust Sensor (Sharp GP2Y1010AU0F)
- Wi-Fi Module (ESP-01)
- Breadboard
- Jumpers wires
- Capacitor & Resistor
- Arduino IDE
- ThingSpeak Real-Time DB
- Python

3.4 Hardware Implementation

3.4.1 DHT11 with Arduino

DHT11 sensor is used to measure temperature & humidity. Here we have connected V_{cc} & ground pins of the sensor to the V_{cc} and ground of the Arduino board respectively. Next, we have connected the data pin to the analog A0 pin of the microcontroller for sending the desired data. Later, using Arduino IDE we have coded accordingly to convert the values into temperature and humidity. For better visualization, check *Figure 5*.

3.4.2 MQ-135 with Arduino

We have used the MQ-135 sensor to measure Ammonia (NH_3). Here we have connected Vcc & ground pins of the sensor to the Vcc and ground of the Arduino board respectively. Next, we have connected the data pin to the analog A1 pin of the microcontroller for sending the desired data. Later, using Arduino IDE we have coded accordingly to calibrate the sensor and then convert the values in ppm. For better visualization, check *Figure 5*.

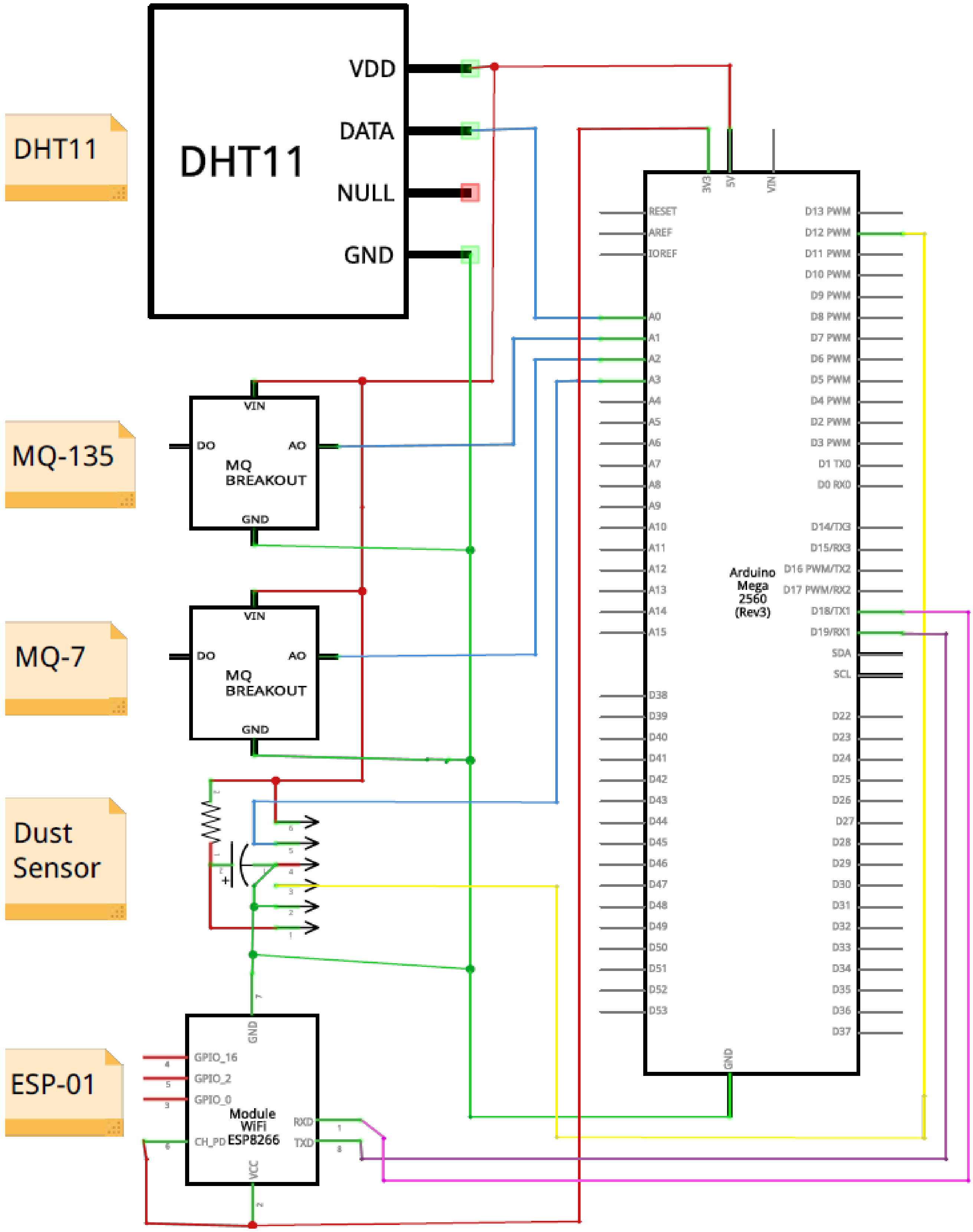


Figure 5: Entire System Schematics

3.4.3 MQ-7 with Arduino

MQ-7 is used to measure the Carbon-monoxide (CO) levels. Here we have connected Vcc & ground pins of the sensor to the Vcc and ground of the Arduino board respectively. Next, we have connected the data pin to the analog A2 pin of the microcontroller for sending the desired data. Later, using Arduino IDE we have coded accordingly to calibrate the sensor and then convert the values in ppm. For better visualization check *Figure 5*.

3.4.4 PM2.5 Dust Sensor with Arduino

We have used a Sharp dust sensor to measure $PM_{2.5}$ levels. Here we have connected Vcc, sensor ground & led ground pins of the sensor to the Vcc and ground of the Arduino board respectively. Next, we have connected the Vout (data) pin to the analog A3 pin of the microcontroller for sending the desired data. Then we connected the VLED to D12 pin through the capacitor (for filtering) and resistor (for the longevity of led). Later, using Arduino IDE we have coded accordingly to calibrate the sensor and then convert the values in mg/m^3. For better visualization check *Figure 5*.

3.4.5 ESP – 01 Wi-Fi Arduino

ESP-01 was used as a wireless network to send sensor data to the cloud via the internet. Here we have shorted Vcc and enable pin and then connected it with $V_{3.3v}$ pin of the Arduino board. Next, we have connected the ground pin of the sensor to the ground of the Arduino board. We have connected the Tx (transmit) and Rx (receive) pin of the module to the Rx and Tx of the Arduino respectively. We also have connected the Vout (data) pin to the analog A3 pin of the microcontroller for sending the desired data. Using Arduino IDE, we have configured the Wi-Fi-Module to send data to ThingSpeak server. For better visualization check *Figure 5*.

3.5 Software Implementation

3.5.1 Arduino IDE

Arduino Integrated Development Environment (IDE) is an application that is used to control and interact with different compatible development boards by writing, compiling & uploading the code into them. The IDE was mainly developed so that anyone interested in DIY (do-it-yourself) projects can code easily with a user-friendly syntax and environment. To write code, first of all, a new sketch needs to be opened. The sketch must have two mandatory functions apart from any user-defined functions. They are *void setup()* & *void loop()*. After completing the code, the sketch is compiled and uploaded to the board if no errors are found. Finally, the Arduino board starts executing the sketch.

The *setup()* is a function where things are initialized. In our case, we have initialized port numbers to build a serial communication path among Arduino IDE, Arduino Mega & ESP – 01 Wi-Fi module. We also have defined the pins that will be treated as input or output.

Next comes the *loop()* function. This function keeps running in a continuous cycle until we turn off the board whereas the *setup()* function runs only once to initialize values or start any sensors, The codes that are to be executed repeatedly are placed here. For this reason, codes related to fetching data from sensors and uploading data to ThingSpeak cloud server are written here.

3.5.2 DHT11 with Arduino IDE

DHT11 has a built-in library called *DHT_sensor_library* in the Arduino programming language which makes it easier to write any code. At first, we have defined the DHT sensor type and the pin number so that the Arduino Mega can recognize the sensor and fetch data easily from it. Next, in the *loop()* function, we have called *dht.readHumidity()* & *dht.readTemperature()* to read temperature and humidity respectively. The conversion from analog sensor values to digital values has been done by the built-in library.

3.5.3 MQ-135 & MQ-7 with Arduino IDE

These two gas sensors work on the same principle. They have a built-in resistor (Rs) whose resistance varies based on the gas concentration. If the concentration of gas is high, resistance decreases and if the concentration is low then resistance increases. To calculate the gas concentration from resistance variation, there is a log to log plot in the datasheet. The plot has Rs/Ro ratio in the y-axis and concentration of gas in ppm in the x-axis. Here Ro is the resistance of the sensor in fresh air whose value needs to be calibrated depending on the area it is kept in. On the other hand, Rs is the variable resistance which determines the concentration of gas. After calibrating Ro and getting Rs from the sensors, Rs/Ro is calculated. Next, this ratio along with the slope and intercept of the log to log plot is used to calculate the concentration of gas in ppm in the x-axis. Moreover, there is another resistor called load resistor (RL) which is used to adjust the sensor's sensitivity to change in gas concentration. But before doing all these, we must calibrate Ro first for our area.

3.5.3.1 Calibration Method:

- At first, the average of 500 analog sensor values (0-1023) for Ro is calculated using *analogRead()* function.
- Next, these values are converted to voltage using the following formula:

$$sensor\ voltage = average\ value \times (supply\ voltage/1023);$$

Here *supply voltage* is 5V.

- Then we determined the voltage drop across the Rs resistor using the following voltage divider rule.

$$V_x = R_x \frac{E}{R_T}$$

Here:

$$V_{RL} = R_L \frac{V_{input}}{R_S + R_L}$$

So for Rs: $$R_s = [(V_{input} \times R_L)/V_{RL}] - R_L$$

- After that, we calculated the value of Ro using

$$R_S/R_0 = Resistance\ Ratio\ Constant$$

$$\text{or, } R_0 = R_S/Resistant\ Ratio\ Constant$$

Here, *Resistance Ratio Constant* for MQ-135 is 3.7 & MQ-7 is 11.7. These values are given in the respective datasheet.

- We kept our sensors on for around 24 hours for getting stable Ro values. R_0 of MQ-135 is 12.09 & R_0 of MQ-7 is 2.31.

3.5.3.2 Calculating Sensor Data and Converting It to Microgram Per Meter Cube (µg /m³)

- After calibration, we started coding to fetch data from the gas sensors. First of all, we have repeated the first 3 points from the calibration method without taking the average to get Rs value.

- After getting Rs, we have determined the concentration of gas in parts per million (ppm) using the following equation:

$$ppm = \left(log_{10}\left(\frac{R_S}{R_0}\right) - y\ intercept\right)/slope$$

Here the value of *y-intercept* & *slope* is calculated from the log to log plot of the datasheet.

- Finally, we converted the ppm into µg /m^3 using the following equation:

$$0.0409 \times ppm \times molecular\ mass\ of\ the\ respective\ gas \times 1000$$

Here *0.0409* is a constant and the *molecular mass* of NH_3 is 17.03 g/mol and CO is 28.01 g/mol.

3.5.4 Sharp Dust Sensor with Arduino IDE

The Sharp dust sensor comes pre-calibrated from the factory. But to get gas concentrations in µg /m^3, we had to write code for the Sharp dust sensor.

3.5.4.1 Calculating Sensor Data & Converting to µg /m^3

- After getting the analog sensor value from the sensor, we have converted it into voltage using:

$$sensor\ voltage = sensor\ value \times supply\ voltage/1023$$

- Next, we determined the dust density in µg /m^3 using the following equation:

$$dustSensity = \big((slope \times sensor\ voltage) - y\ intercept\big) \times 1000$$

Here the value of *slope* and *y-intercept* is calculated from the graph given in the datasheet

3.5.5 ESP – 01 with Arduino IDE

ESP – 01 Wi-Fi module can send as well as receive data. It is configured using the *AT* commands that are typed in the serial monitor of Arduino IDE.

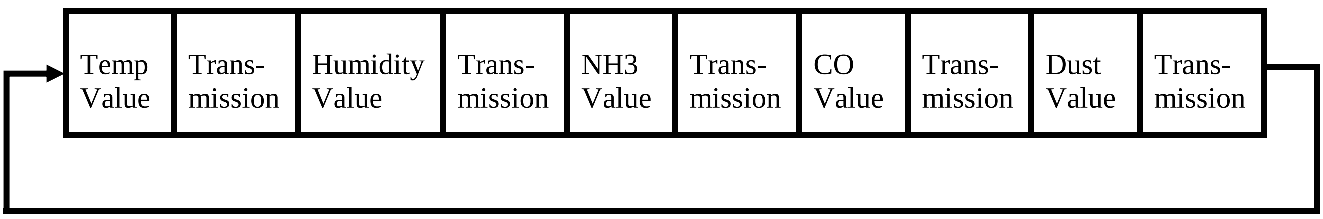
3.5.5.1 Connecting Wi-Fi Module with a Wi-Fi router

- ESP – 01 can be set up as a client or as a host. Here, we have set it up as a client as it serves our purpose. First of all, we have sent the *AT+CWMODE=1* command to set our Wi-Fi module as a client

- Then we have sent *AT+CWJAP="SSID", "PASSWORD"* command to the Wi-Fi module to get connected with any router.

3.5.5.2 Sending Data using ESP – 01 Wi-Fi Module.

- First of all, we have set mux = 0 using the *AT+CIPMUX=0* command, because we want a single connection.
- Next, we have opened a TCP connection using *AT+CIPSTART= "TCP", "184.106.153.149", 80* command where *184.106.153.149* is the IP address of ThingSpeak and *80* is the port number.
- Then we have prepared our API request string to send the data which is -*"GET /update?api_key=XXXXXXXXXXXXXXXX&field1=0"*. Here *X….X* is our channel's write API key and *&field1* is the column in ThingSpeak where a particular sensor data will be stored and *0* is the sensor data.
- Next, we send the *AT+CIPSEND = 70* command into the ESP – 01 Wi-Fi module to fix the length in bits of the API request string. Here length is *70.* This is done as the length needs to be fixed before sending the string in the ThingSpeak server.
- After completing the above steps, we can send the API request String to the ThingSpeak server via the ESP=01 module. In this way, we can send our sensor data one by one to the ThingSpeak server using the API request string.
- The following diagram shows the transmission time required by the Wi-Fi module to upload sensor data one by one. It takes 45 seconds for a sensor data to get uploaded in its field and a particular sensor field gets its consecutive next data uploaded after every 3.5 minutes.



Here, the Transmission time is 45 seconds

Figure 6: Fetching & Sending Data to ThingSpeak

3.5.6 ThingSpeak

ThingSpeak is an IoT based cloud service platform through which you can store data from sensors or any other embedded device using IoT protocols to visualize the data in the form of charts or graphs. In our case, we have used the write API key to send data from our sensors to the respective columns in ThingSpeak. This helped us visualize our data in the form of graphs where the x-axis is the time and y-axis is the sensor data. We also have used the read API key to fetch data from ThingSpeak to Python to analyze the data and build predictive models for the dataset.

3.5.7 Python

Python is a multi-purpose programming language that is simple & convenient to use with a user-friendly syntax. It is used in a variety of places. Some noteworthy examples are data analysis & visualization, artificial intelligence, machine learning & automation. In our case, we have used Python language to fetch, preprocess, and analyze our dataset. We have also used Python to evaluate the best model for forecasting our parameter data and train it. Further details about forecasting are described in the next section.

3.6 Selecting Model for Prediction

3.6.1 Data Preprocessing

Data preprocessing may be defined as converting the raw data from sensors or other devices into a well-organized functional format by trimming out garbage values, noises, and arranging it according to the needs of a model. In our case, while collecting raw data we had to remove null values and also make the data univariate i.e. is a function of only one variable. Since we will be applying a time-series model to predict future values, we ensured that our data only depended on a single variable which is time.

3.6.2 Evaluating Best Model

There are various models for time-series data prediction. But before selecting a model, it is mandatory to see if the model suits our dataset or not. For this reason, we have omitted some models as it would not give us accurate scores. Auto Regression and ARMA (Auto Regressing Moving Average) were also not suitable since these models require a stationary dataset. But our dataset has some non-stationary parameters. Besides, we tried using the models Linear/Polynomial Regression and LSTM (Long Short-Term Memory) but ended up dropping those as they were giving inaccurate results. The main reason was the lack of observations in both cases. Lastly, we also didn't use Seasonal ARIMA (Auto-Regressive Integrated Moving Average) and ARIMAX because of insufficient data as well as our data not being multivariate.

3.6.3 Selecting ARIMA as Our Model

We have selected ARIMA (Auto-Regressive Integrated Moving Average) because it suited our dataset perfectly. The main reason is that this model allows AR (Autoregressive) components, MA (Moving Average) components and it can also determine the level of differencing required to make the data stationary. Autoregression (AR) depicts a changing variable that returns to its own lagged, or prior, values. Integrated (I) shows the differencing of raw data allowing the time series to become stationary, i.e. observations are substituted by the difference between the earlier observation and the current observation. The moving average (MA) constitutes the dependency between a data value and a residual error of a moving average model which was applied to lagged data values.

Chapter 4 **EXPERIMENTS & RESULTS**

In this chapter, we perform various experiments with our dataset to draw certain conclusions. At first, we experiment to observe the parameter & AQI values for seven consecutive days and determine the prime pollutant for indoors. Next, we compare our indoor data with that of an outdoor weather station and analyze the differences. Finally, we test the accuracy of our prediction model (ARIMA) by training it with a dataset of a certain period.

4.1 Observing Parameter & AQI Values from Our Sensors & Determining Prime Pollutant for Indoors

Table 3: 7 day 24hr – Avg values & AQI of all parameters from 22nd to 28th April

Date	24hr – Avg					AQI			Overall AQI
	$PM_{2.5}$ µg/m³	CO mg/m³	NH_3 µg/m³	Temp	Humidity	$PM_{2.5}$	CO	NH_3	
22/04/2020	118	0.31	1177	30 °C	62%	293	16	294	294 Poor
23/04/2020	125	0.41	1111	31 °C	65%	304	21	278	304 Very Poor
24/04/2020	104	0.25	823	30 °C	67%	244	13	207	244 Poor
25/04/2020	111	0.33	913	29 °C	68%	270	16	229	270 Poor
26/04/2020	115	0.35	785	29 °C	66%	284	17	196	284 Poor
27/04/2020	124	0.41	868	30 °C	66%	303	21	218	303 Very Poor
28/04/2020	123	0.42	741	29 °C	69%	303	21	185	303 Very Poor

The table shows the 7-day 24hr-Avg values of all the parameters from 22nd to 28th April. All of these were calculated after properly calibrating the sensors. We fail to observe any proper trend in the data for all the parameters. Moreover, we see that overall AQI was determined by $PM_{2.5}$ for the majority of the cases although NH_3 comes at a close second. The AQI values reflect that the air quality was "poor" or "very poor" for all the days. This is caused by several reasons. Typically, indoor $PM_{2.5}$ levels are estimated to be the same as outdoor $PM_{2.5}$ levels provided that the house is free from any strong combustive activities like smoking. So in our case, the high indoor PM levels are mainly due to pollution from outdoors. On the other hand, high indoor levels of NH_3 are caused due to the emission of NH_3 from refrigerants, indoor concrete walls, cleaning products, smoke, and exhaled human breath & sweat. So we can conclude that the main pollutants of indoor air pollution are $PM_{2.5}$ followed by NH_3.

4.2 Comparing Our Indoor Data to a Renowned Outdoor Weather Station

Table 4: Indoor vs Outdoor parameter values of 7 days

Date 22nd - 28th April / Data Type	$PM_{2.5}$ (7 days Avg)	Temperature (7 days Avg)	Humidity (7 days Avg)
Indoor	**117 μg/m³**	**30 °C**	**66%**
Outdoor	**47 μg/m³**	**26 °C**	**82%**

The table shows the 7 day average of indoor and outdoor values [9]. These are not the same and it will never be the same since the factors affecting the pollutant levels indoor and outdoor are different. Besides, we are comparing our indoor data to a weather station situated quite far away from our station which results in different pollution levels. Moreover, there is a ±5% error in our sensors. For all these reasons, indoor and outdoor air quality differs to a great extent.

4.3 Training Our Model for Prediction & Testing Its Accuracy

4.3.1 Data Analysis

An ARIMA model is characterized by 3 terms. They are d (the no. of differencing required to make time-series stationary), p (order of Autoregression (AR) term), and q (the order of moving average (MA) term). We need to find the d, p & q values for all of our parameters to train our model.

Using the Augmented Dickey-Fuller (ADF) test, we can find out if data is stationary or not. If the null hypothesis of the ADF test is less than 0.05 and the test statistic of the parameter is less than the critical value, then the parameter is considered stationary and thus we set the value of d to zero. Otherwise, if the data is non-stationary, we need to perform differencing to make it stationary. In this case, d represents the no. of differencing required to make the parameter stationary.

Table 5: Augmented Dickey-Fuller Test (ADF) of all parameters

Parameter	No. of observations	Test Statistic	Critical Value(1%)	Null hypothesis	Result
Temperature	**139**	**-3.87**	**-3.48**	**0.002278**	**Stationary**
Humidity	**143**	**-4.95**	**-3.48**	**0.000028**	**Stationary**
$PM_{2.5}$	**143**	**-4.52**	**-3.48**	**0.000181**	**Stationary**
CO	**138**	**-4.00**	**-3.48**	**0.001395**	**Stationary**
NH_3	**138**	**-2.67**	**-3.48**	**0.079018**	**Non-stationary**

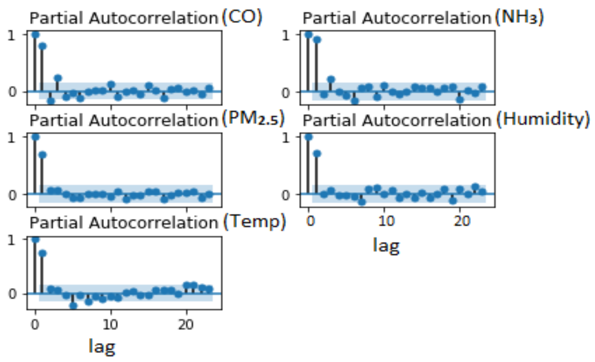
From the table we can see that, 4 out of 5 parameters are already stationary since the null hypothesis of these parameters is less than 0.05 and their test statistic values are less than critical values (1%). Here a critical value of 1% signifies that we are 99% confident that our parameter is stationary when we compare it with the parameter's corresponding test statistic. So the 4 parameters Temperature, Humidity, CO & $PM_{2.5}$ don't require differencing and their d value is zero. The only non-stationary parameter in our dataset is NH_3 which can be made stationary by differencing. After differencing NH_3 one time, we get the following values from the ADF test:

Table 6: ADF Test of NH_3 after differencing one time

	No. of observations	Test Statistic	Critical Value (1%)	Null hypothesis	Result
NH_3	**138**	**-10.73**	**-3.48**	**0.000000**	**Stationary**

So we observe that NH_3 has become stationary after differencing one time. Thus, the d values for Temperature, Humidity, CO, NH_3 & $PM_{2.5}$ are 0, 0, 0, 1 & 0 respectively.

Now, to find out p (order of AR) and q (order of MA), we have graphed the PACF (Partial Auto Correlation Function) plot & ACF (Auto Correlation Function) plot respectively.



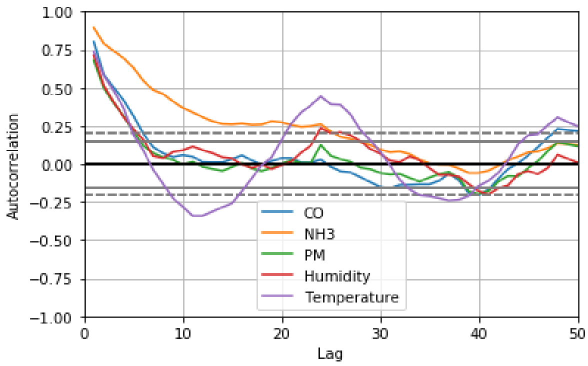


Figure 7: PACF plot of all parameters

Figure 8: ACF plot of all the parameters

Partial Auto Correlation is the correlation between the series and its lag. Here, p is the no. of lags required by a parameter to cross the significance limit (blue shaded region) in figure 7 which is the PACF plot of our parameters. So observing the PACF plot, it becomes clear that CO has a p-value of 2 since it took 2 lags to enter the blue shaded region. For the same reason, the p values for Temperature, Humidity, NH_3 & $PM_{2.5}$ are all 2.

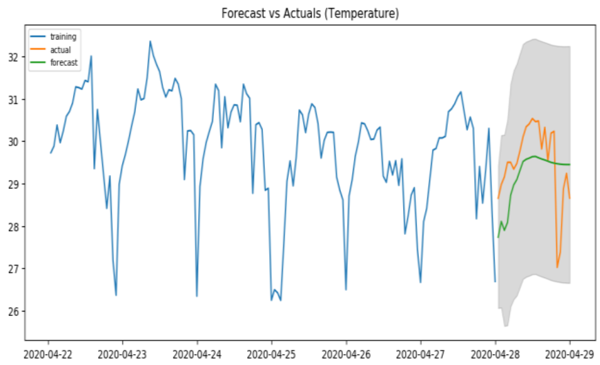
Auto Correlation displays the errors in the lag forecast which is technically the moving average (MA) term. The ACF tells how many MA terms (lags) are required to remove autocorrelation in the stationary series. Here, q is the no. of lags required to remove autocorrelation. So observing the ACF plot in figure 8, it becomes clear that CO has a q value of 2 since it took 2 lags to remove autocorrelation or in other words to enter the region of -0.25 to +0.25. Similarly, the q values for Temperature, Humidity, NH_3 & $PM_{2.5}$ are 1, 1, 3 & 3 respectively.

These were the initial values of p & q. But to tune the model to maximum accuracy, we have to observe the AIC (Akaike Information Criterion) & BIC (Bayesian Information Criterion) values of p & q of each parameter. The lagged value with the lowest combination of AIC & BIC was selected as p-value and q value respectively for each of the parameters. Because the lagged value with the lowest combination produces the most accurate prediction values.

After analyzing the AIC and BIC values the final p, d and q value for Temperature is 5, 1, 1, Humidity is 1, 1, 1, CO is 1, 0, 2, NH_3 is 2, 1, 3 & $PM_{2.5}$ is 3, 0, 1 respectively.

4.3.2 Actual Versus Forecast Values

After getting the d, p & q values for all the parameters, we trained our ARIMA model. We trained our model using 144 hourly observations of 6 days and predicted the next 24 hourly observations. In other words, we trained our model using the first 6 days of data (22nd to 27th April) and predicted the next 1 day (28th April) for all the parameters (CO, NH_3, $PM_{2.5}$, Humidity, and Temperature).



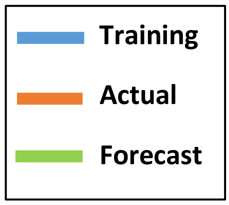


Figure 9: Actual vs Forecast Values of Temp

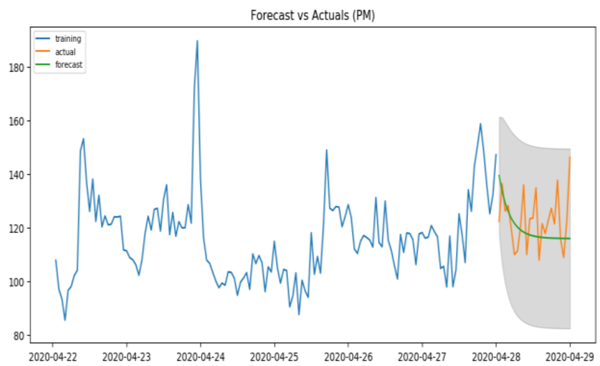


Figure 10: Actual vs Forecast Values of $PM_{2.5}$

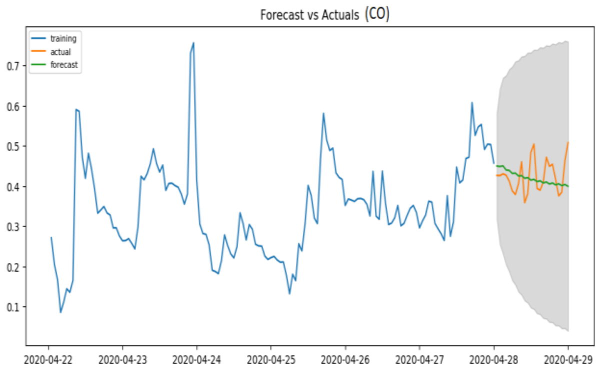


Figure 11: Actual vs Forecast Values of CO

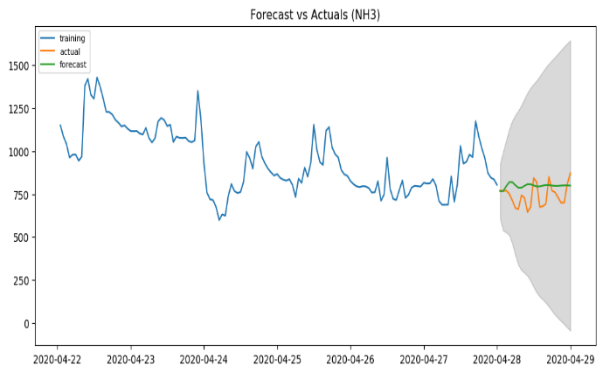


Figure 12: Actual vs Forecast Values of NH_3

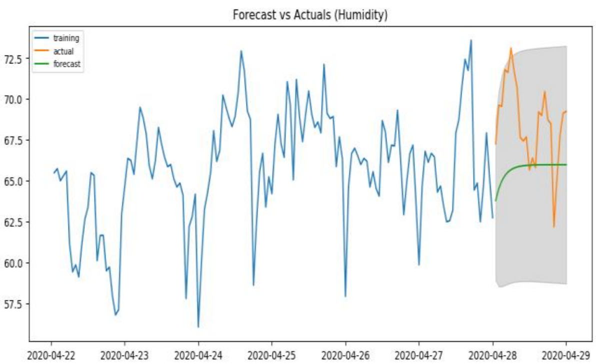


Figure 13: Actual vs Forecast Values of Humidity

Observing the actual vs forecast plots of all the parameters, we can conclude that, due to high accuracy the actual and forecast lines of all the parameters have coincided. Moreover, except for the temperature plot in figure 9, all the other plots have no seasonality. We can observe slight seasonality in the temperature graph. Besides, we can only observe a minute decreasing trend in the NH_3 plot of figure 12. All other plots have no visible trends.

4.3.3 Accuracy Score

Table 7: Accuracy score of forecasted values of all the parameters

Parameter	MAPE (Mean Absolute Percentage Error)
Temperature	**2.82 %**
Humidity	**4.70 %**
$PM_{2.5}$	**6.75 %**
CO	**10.12 %**
NH_3	**10.30 %**

MAPE (Mean Percentage Error) is generally used to determine the accuracy of a model. So we calculated the MAPE score for all of our parameters using the following formula:

$$\text{MAPE} = \frac{\sum_{i=1}^{n} \frac{|forecast - actual|}{actual}}{n} \text{ ; where,}$$

n = the no. of observations,

forecast = forecast values of a parameter,

actual = actual values of a parameter.

Here we see that all the parameters except NH_3 & CO have accuracy above 90%. From the table, we can see that NH_3 & CO have an MAPE score of 10.30% & 10.12% respectively. NH_3 lacks accuracy due to two reasons. First is the high magnitude values of NH_3 compared to other parameters. The other reason is the autocorrelation lag of NH_3 is quite high. On the other hand, CO has only a high autocorrelation lag. If we could take observations for a larger period of time and if we could perform hyperparameter tuning, then the accuracy of both NH_3 & CO would have increased.

Chapter 5 **CONCLUSION**

5.1 Discussion

At the beginning of our project, we had a tough hill to climb with certain objectives. But in the end, we were successful to do so. We have accomplished the task of calculating various pollutant parameters (CO, $PM_{2.5}$, NH_3, temperature & humidity) accurately after properly calibrating the sensors. Besides, we also built a system through which anyone can monitor pollution from anywhere in the world by sending data to the ThingSpeak cloud server via a Wi-Fi module. Finally, we have successfully implemented a model (ARIMA) to train our dataset and have achieved accurate prediction results for our indoor air pollutants.

5.2 Future Improvements

No matter how good a project is, there is always room for improvement. Our project is no exception. Below are the possible improvements that we would like to add to our project in the future.

5.2.1 Software Improvements

The first improvement that we would like to do is build a website along with a mobile application so that users can visualize the pollution levels and their health impacts conveniently at any time. In our current system to monitor pollution levels, the user has to go to the ThingSpeak cloud server. Besides, the user can only observe graphs of the parameters. For this reason, we would like to allow the user to observe not only graphs but the parameter values and AQI of the past 7 days on our website and mobile app. Moreover, we would also like to show the predicted values of the next 7 days.

Next, we want to further test our prediction model (ARIMA) rigorously to enhance its prediction capabilities. In our current system, we experimented on a 7-day time frame due to time constraints and got a decent MAPE score of below 10% for all parameters except NH_3 (10.3%) & CO (10.12%). Moreover, we couldn't observe clear seasonality and trend in the pollutant parameters. So in the future, we would like to take a larger time frame like a month or possibly 6 months which would allow us to use other models with ARIMA i.e. hybrid model like ARIMA with LSTM to not only further increase the accuracy of the model but also to observe a definite seasonality and trend.

5.2.2 Hardware Improvements

The hardware we used to measure pollutants was affordable, reliable, and quite accurate. But to ensure higher accuracy in the future, we would like to use Nova PM Sensor instead of a Sharp dust sensor to measure $PM_{2.5}$ values. Besides, in our current system, we have used 4 sensors to calculate 5 parameters which are temperature, humidity, carbon monoxide, ammonia, and $PM_{2.5}$. Our future objective is to include more sensors to measure ozone (O_3), carbon dioxide (CO_2), sulfur dioxide (SO_2), and PM_{10}.

Lastly, we also would like to commercialize our device by making it portable, affordable, and user-friendly so that it can be conveniently used in homes, offices, industries, and factories as an indicator of air quality and make people conscious of the environment.

REFERENCES

[1] “Air Pollution,” *Who.int.* [Online]. Available: https://www.who.int/health-topics/air-pollution#tab=tab_1 [Accessed: 25th Feb’2020]

[2] “Clean and Resilient Growth in Bangladesh,” *WorldBank.org*. [Online]. Available: https://www.worldbank.org/en/news/feature/2018/09/16/clean-and-resilient-growth-in-bangladesh [Accessed: 25th Feb’2020]

[3] “বায়ুদূষণে ক্ষতি সোয়া লাখ কোটি টাকা,” *ProthomAlo.com*. [Online]. Available:https://www.prothomalo.com/bangladesh/article/163950 [Accessed: 25th Feb’2020]

[4] Q. P. Ha, S. Metia and M. D. Phung, "Sensing Data Fusion for Enhanced Indoor Air Quality Monitoring," in *IEEE Sensors Journal*, Apr. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8950378 [Accessed: 25th April’2020]

[5] Andrew Rebeiro-Hargrave, Naser Hossein Motlagh, Samu Varjonen, Eemil Lagerspetz, Petteri Nurmi and Sasu Tarkoma, “MegaSense: Cyber-Physical System for Real-time Urban Air Quality Monitoring,” in *15th IEEE Conference on Industrial Electronics and Applications (ICIEA2020)*, Jun. 2020. [Online]. Available: https://researchportal.helsinki.fi/en/publications/megasense-cyber-physical-system-for-real-time-urban-air-quality-m [Accessed: 25th April’2020]

[6] JunHo Jo, ByungWan Jo, JungHoon Kim, SungJun Kim and WoonYong Han, “Development of an IoT-Based Indoor Air Quality Monitoring Platform,” in *Journal of Sensors*, Jan. 2020. [Online]. Available: https://www.hindawi.com/journals/js/2020/8749764/ [Accessed: 25th April'2020]

[7] Press Information Bureau, Ministry of Environment, Forest and Climate Change, Government of India, "National Air Quality Index (AQI) launched by the Environment Minister AQI is a huge initiative under 'Swachh Bharat'," 2014. [Online]. Available: https://pib.gov.in/newsite/PrintRelease.aspx?relid=110654 [Accessed: 25th Feb'2020]

[8] Central Pollution Control Board, Ministry of Environment, Forests & Climate Change, Government of India, "National Air Quality Index," 2014. [Online]. Available: http://www.indiaenvironmentportal.org.in/files/file/Air%20Quality%20Index.pdf [Accessed: 25th April'2020]

[9] "Uttara, Dhaka Air Quality and Weather Daily Forecast," *IQAir.com*. [Online]. Available: https://www.airvisual.com/bangladesh/dhaka/uttara [Accessed: 25th April'2020]