# A Pollution Measuring Sensor Network for E-Bikes with Real-TimeData Collection and Data Visualization

## [Team Deliverable G]

## GNG5140
## Engineering Design

Submitted by

**[TEAM PERSONAL ENVIRONMENT A]**

[Mohit Mayurkant Shah, 300269756]

[Rahul Reddy Kotireddy, 300279217]

[Aleesha Maria Abraham, 300326771]

[Bahar Nikkhoo, 300313653]

[Kshitij Deshpande, 300323074]

Mentored By

Dr. David Bruce, University of Ottawa

Dr. Thomas Walker, Carleton University

Jack Wang, WeBike Club, University of Ottawa

Date: March 30, 2023

University of Ottawa

# Abstract

Air pollution is one of the biggest environmental challenges we face today, and its adverse effects on public health and the environment are well documented. In response to this challenge, we present a prototype for an Integrated Sensory Network that provides real-time pollution measurements and data visualization through E-bikes. Our goal is to develop a system that is easily deployable, low-cost, and highly accurate. The system is designed to monitor four major aspects of air pollution, including ozone, temperature, humidity, and PM 2.5 using a network of sensors mounted on an E-bike. The collected data is then transmitted wirelessly to a cloud-based platform for storage, analysis, and visualization. Our prototype design consists of physical and software components. The physical prototype includes a 3D-printed casing, and a circuit constructed using off-the-shelf components such as Arduino Uno Rev3 microcontroller, DHT22 Temperature and Humidity Sensor, MQ131 Ozone Level Sensor, PMS5003 Particulate Matter Sensor, GPS (Neo-6M) module, and an LCD. The software prototype is developed using the Arduino Integrated Development Environment (IDE), and the backend architecture includes a Flask micro web framework for API development and MySQL for database management. The system aligns with the United Nations Sustainable Development Goals (UNSDGs) of Good Health and Well-being, Quality Education, Sustainable Cities and Communities, and Climate Action, and promotes community involvement to tackle the greater issue of environmental pollution. The prototype design considers material usage and recycling, energy consumption, and usability aspects. The usability aspect of the system is evaluated based on the 10 general principles defined by Jason Nielson, which specifies usability guidelines for any system design. Our design has been evaluated based on effectiveness, efficiency, engagement, error tolerance, and ease of learning. In conclusion, our prototype design presents a sustainable solution for air pollution monitoring that promotes community involvement and aligns with the UNSDGs. Our design can be easily scaled up for larger-scale deployment and has the potential to contribute to research, policy-making, and public awareness efforts related to air pollution.

# Table of Contents

# List of Figures

# List of Tables

## List of Acronyms

| Acronym | Definition |
|---|---|
| $CO_2$ | Carbon Dioxide |
| $NO_x$ | Nitrogen Oxides |
| $SO_2$ | Sulphur Dioxide |
| $O_3$ | Ozone |
| VOCs | Volatile Organic Compounds |
| CAAQS | Canadian Ambient Air Quality Standards |
| AQMS | Air Quality Monitoring System |
| $PM_{2.5}$ | Particulate Matter (2.5microns or less in width) |
| UNSDGs | United Nations Sustainable Development Goals |
| LCD | Liquid Crystal Display |
| PLA | Poly-Lactic Acid |
| PCB | Printed Circuit Board |
| IoT | Internet of Things |
| API | Application Programming Interface |
| SQL | Structured Query Language |
| REST | Representational State Transfer |
| IDE | Integrated Development Environment |
| CAD | Computer Aided Design |
| GPS | Global Positioning System |

# 1 Introduction

## 1.1 Background

Because of a variety of natural and man-made phenomena, harmful materials called air pollutants are repeatedly introduced into the atmosphere due to a wide range of chemical and physical processes. These pollutants are then largely distributed due to air patterns and wind cycles gradually spreading throughout the atmosphere. These pollutants can arise naturally like volcanic ash or by virtue of man-made sources like factories, automobiles, aircrafts, landfills, wastewater, maintenance holes, and leaking natural gas infrastructure. These air pollutants predominantly comprise of Ozone, $CO_2$, Carbon monoxide, $NO_x$, sulphur dioxide, VOCs, lead, and oxides of sulphur which can have detrimental effects on human health such as cardiovascular and respiratory diseases, and pulmonary diseases like lung cancer. This makes it essential to study and monitor the distribution of these pollutants in the atmosphere and moreover, identify and analyze the sources of air pollution to curb this worldwide issue.

## 1.2 Motivation

As the old saying goes, modern problems require modern solutions. Therefore, to tackle present-day issues of air pollution, it is important to focus on rapid gathering of data pertaining to the local distribution of air pollutants in the atmosphere. This would help us determine the sources of air pollution and enable us to find ways to control the emission of these hazardous pollutants into the environment.

The Canadian Ambient Air Quality Standards (CAAQS) as established under the Canadian Environment Protection Act 1999 are a key aspect of Air Quality Management System (AQMS) which navigates the improvement of Air Quality across Canada. These standards have been developed for fine particulate matter (PM2.5), nitrogen dioxide (NO2), sulphur dioxide (SO2), and ozone (O3) in particular. Industries, non-governmental and indigenous organizations, using assistance from the federal government, set up air monitoring stations.

Provinces and territories monitor these criteria air pollutants and annually record the air quality to document air quality concerns and trends, and the achievement of CAAQS for each air zone. This is done using measuring instruments called Federal Reference Methods or Reference Monitors.

However, data collected from specific locations (e.g., Local airports and fixed weather stations) are not fully representative of the ambient conditions in local neighborhoods due to variations in elevation, vegetation, and heat/waste coming from automobiles, factories and residential buildings. Moreover, fixed weather stations provide an area-averaged quantitative understanding of pollution levels. In contrast to this, remote sensing provides real-time data on land cover with better spatial resolution and comparatively lower costs, and this is the basis for our project.

## 2. Objectives

This project describes the design and validation of a mobile sensor array to monitor pollution levels with real-time data acquisition and data visualization, using cloud interaction. Our primary aim is to build a mobile pollution monitoring system using low-cost sensors, which would be mounted on electric bikes. Though the data quality may be unknown and variable in the early stages, it poses considerable advantages over fixed monitoring systems. Being portable and inexpensive, it enables monitoring pollution levels across more locations with good spatial resolution. By virtue of its simplicity, this system eliminates the need for highly skilled technicians for operation. Moreover, the frequency of data collection can be automatically regulated to have a spatial resolution of 20-50 meters.

The developed system would help identify pollution hotspots facilitating the identification and further investigation of pollution sources. We are also inclined to raise awareness among the general public about air pollution and its harmful effects, to promote community engagement by enabling users to monitor pollution levels along their daily routes and pathways around the city.

The task at hand is to develop a sensor array to monitor PM2.5, Ozone, and GPS co-ordinates and use this data to co-relate pollution to personal commuting routes. Power and communication integration are expected to be supported using common cellular architectures.

# 3. Literature review

John C. Cassano [1] tested a bicycle-based weather station using the Kestrel 4000 and a Garmin Edge 800 GPS tracker to collect local temperature changes. While the purpose of this device differs from the scope for this project, temperature, humidity, and pressure were relevant metricsmeasured by The Weather Bike. Data from the Garmin Edge 800 and the Kestrel 4000 were correlated to ensure the time stamp matched, allowing for accurate mapping of the sensor data with the GPS locations. Both devices had a resolution of 1 second. Temperature, humidity, and pressure measurements were taken using the Kestrel 4000.

Theo Brandsam et al. [2] looked to model the urban heat islands along a transect in Utrecht. The Elpro TH1 data-logger along with two sensors were used to collect necessary data. The NTC pressure drop sensor is beyond the scope of this project and hence was ignored. A combination sensor capable of measuring temperature and humidity was also used.

Nicholas et al. implemented a bicycle-based measurement system to study the thermal exposure in Ohio. While the majority of the sensors were focused on measuring radiation and similar parameters, other relevant measurements such as temperature, humidity, and pressure were also measured.

Jaime et al. [3] developed a bike-mounted device to measure air quality. Their device incorporated cloud connectivity, to allow for frequent air quality monitoring. Several sensors were used. With a sampling rate of 3 seconds, the device collected the following parameters: date, time, latitude, longitude, altitude, number of satellites in coverage and used, battery voltage and percentage, temperature, humidity, pressure, external-fan rpm, PM1, PM2.5, PM10, OPC-N3 sensor flow rate, NO2 (ppb), O3(ppb), and gas-sensor electrode raw data. The data was stored in an on-board SD card and communicated to the cloud platform using the GPRS/GSM module. Specific data such as the battery status, among other parameters, was also displayed on an LCD screen mounted on the device.

Pedro Lucas et al. [4] collected air pollution measurements from external nano-sensors through the “Data Aggregation Module”. This module receives the data, validates it and sends it to the central server. As the measurements arrive at the central server, they are received by the “Data Loader”, a module to consume data and inserts it into the database.

The NanoSen-AQM allows users to access air quality data via the platform, which can be publicly accessed using a web browser. It can prove useful for users who want to know the air quality at a specific location or for researchers who wish to understand the evolution of an air pollutant over time. Accounts can be created on the platform, allowing users to mark specific clusters as favourites and set up alerts for cases where air quality data exceeds a given threshold. Special users with "sensor owner" privileges, can manage their sensors and clusters and upload adjusting functions. Users can also receive notifications regarding the air quality in specific clusters. Alerts can be created by the user to get email and push notifications (if using a mobile device) whenever the air quality in a certain cluster crosses a user-defined threshold.

Wesseling, J. et al. [5] conducted a study based on the 'Sniffer Project' that aimed to measure the PM2.5 concentration in the province of Utrecht, Netherlands using the data collected from 500 cyclists who used the Sniffer bike sensor kit. The kit consisted of dust, temperature & humidity sensors in addition to a GPS and an accelerometer. The device which was active only while the bicycle moved was mounted to the handlebar. Measurements were done every 10 seconds and sent along with the location details to a server at the company, Civity. The data is then stored in a database, calibrated and then posted back to Civity.

N. Genikomsakis et al. [6] developed a low-cost Air Pollution Monitoring sensor that is microcontroller based and has an Optical PM sensor, along with data logger and GPS module. Microcontroller is programmed such that the time stamped PM sensor measurement is obtained and send to data logger every four seconds. Data is stored in separate .csv format files each day.

## 3.1 Feature and Flaw Analysis of Existing Solutions

### 3.1.1 Solution 1

The NanoSen-AQM project proposes the usage of low-cost nano sensors as the basis for an air quality monitoring platform. It also provides easy access to air quality data, with the aim of improving public health.

**Features:**

The NanoSen-AQM project is a system that collects air pollution measurements using low-cost external nano-sensors. The system has a "Data Aggregation Module" that aggregates the data and sends it to the central server through a Kafka Topic. The data is received by the "Data Loader" module and inserted into the database. The "Data Adjustment Loader" module collects raw measurements, which are then consumed by the adjustment pipeline. The system periodically collects air quality data and inserts it into a Kafka Topic, which is then consumed by the “Fetcher Connector.”

Users can interact with the system using applications created with Ionic. Administrators can manage users, sensors, and other aspects of the platform using the “Administration Application” created with Django. Users can receive notifications about air quality in specific clusters and create alerts for specific locations. The system displays all clusters on a map for better visualization. Sensor owners can create individual sensors, assign them to a cluster, and make their data private or public. Air quality data is publicly accessible using a web browser.

**Flaws:**

- The platform could use more external air quality sources to expand its coverage.
- The clusters of sensors spread around the world are sometimes very distant from each other.
- Future air quality values are not properly forecasted.

**Future work:**

Currently, OpenAQ is the only external source being used, which has many sensor clusters around the world. The platform is also developing air quality forecasting to predict future air quality values.

### 3.1.2 Solution 2

Wesseling, J. et al. conducted a study based on the ‘Sniffer Project’ that aimed to measure the PM2.5 concentration in the province of Utrecht, Netherlands using the data collected from 500 cyclists who used the Sniffer bike sensor kit. The kit consisted of dust, temperature & humidity sensors in addition to a GPS and an accelerometer. The device which was active only while the bicycle moved was mounted to the handlebar. Communication was facilitated by LTE. Measurements were done every 10 seconds and sent along with the location details to a server at the company, Civity. The data is then stored in a database, calibrated, and then posted back to Civity.

The metrices that determine the ‘quality ‘of this device are mainly the PM sensor accuracy, precision of GPS depending on the location of the bikes (e.g., in the woods) and sensor calibration.

**Features:**

One main feature of the Sniffer unit is the sampling time of 10 seconds. Also, the use of low-cost, yet highly unbiased PM sensors due to proper calibration is another plus. These sensors are almost equally effective as compared to government official equipment. This is clear from a graphical representation that shows the same lognormal shape for the concentration of PM 2.5 using both sensors. The device was able to find the average traffic related PM2.5 exposure in the city of Utrecht, which was found to be 2.0 µg/m3. The project was successful in measuring the average value of exposure of cyclists to PM2.5 as 9.9 µg/m3.

**Flaws:**

- The limitations of the device are mainly the spread of routes in bike routes, the change in bike and local traffic during peak times of the day i.e., morning and evening vs nighttime and the fluctuations in the precision of GPS in areas where trees are dense, like a trail.
- Another drawback to the device was the use of the PM sensor that was effective for PM2.5 but not PM 10. The time period during which the sensor remains active at a time is low as majority of bike rides span for only 15minutes.

**Results:**

The device was used by bike riders (almost 8 million) for a year in the city of Utrecht and almost 68000 valid measurements were taken to arrive at the average PM2.5 concentration value in this study. The data collected was more during the summer months than in winter. Also, the study was conducted during the Covid lockdown and thus, might be undermining the local traffic PM contribution. The study suggests the use of shared Sniffer units or using them while stationery to overcome some of the limitations.

### 3.1.3 Solution 3

Konstantinos N. Genikomsakis et al. worked on the concept of developing a low-cost portable air pollution monitoring system (APMS) for measuring the concentrations of particulate matter (PM), fine particles with a diameter of 2.5 µm or less (PM2.5). This paper presents the on-field testing of the proposed low-cost APMS implementation and demonstrates the intended application of collecting fine-grained spatio-temporal PM2.5 profiles by mounting the developed APMS on an electric bike as a case study in the city of Mons, Belgium.

A single board microcontroller equipped with a data-logging extension, a GPS module, and an off-the-shelf optical PM sensor, namely Laser PM2.5 Sensor SDS011 by Nova Fitness, DHT22 sensor (measures temperature and relative humidity, ±0.5 °C for temperature and ±2% (maximum ±5%) for relative humidity.) and a polypropylene box with dimensions 27× 11× 7 cm and weight 150 g.

**Features:**

The calibration of the developed system was performed with a Mobile Lab equipped with an optical particle sizer. The delay between the measurements of the comparison instrument and the low-cost APMS is caused by the properties and limitations of the specific low-cost PM2.5 sensor that was employed. Moreover, this work examines the potential of a portable APMS with integrated low-cost sensor technologies to perform reliable measurements of PM2.5 concentrations in the ambient air after on-field calibration with a certified optical particle counter instrument.

**Flaws:**

- The accuracy of the system can be improved with further on-field calibration and validation.
- A wider range of PM2.5 concentrations must be accessed to reduce the residual errors observed when measuring higher concentrations.
- There is the risk of accumulating dust inside the PM sensors’ optical chamber, which can affect the accuracy of the measurements.

**Future work:**

Other directions of future work include a comparative analysis with at least one other commercially available low-cost PM sensor and already characterized in other publications, as well as a reproducibility study based on more low-cost PM sensors of the same type.

# 4. Needs and Requirements

## 4.1 Client Needs Identification

The team interviewed all the clients individually through a list of open-ended questionnaires and a dialogue on their short-term and long-term vision for the project. Based on this information, the client needs and the constraints for the implementation of Prototype-I has been identified.The client requirements are classified based on Rigid and Soft requirements.

Table 1. Client Need Identification

| Rigid Requirements | | Soft Requirements | |
|---|---|---|---|
| | **Functional Requirements** | | **Aesthetics and User Motivation** |
| 1. | Spatial Resolution: The device needs to be capable of measuring the data with a minimum spatial resolution in the range of 10m-20m. | 1. | The device to have an aesthetic appeal to motivate the users carry with them/ mount them on their E-bikes. |
| 2. | Real-time Data: The device needs to measure and record the sensor data on a real-time basis | 2. | The Size and Geometry of the device needs to comply with various E-bikes models |
| 3. | Data Anonymity: The data utilized for Analysis and Visualization purpose needs to provide some level of anonymity. | | **Power Consumption** |
| | | 3. | The circuit design needs to consider the power requirements of the device. |
| | **Scalability of the Solution** | | **Trade-off among Requirements** |
| 4. | Scalability: The proposed solution and the methodology used for system design- data measurement, data collection, and data analysis need to be scalable for 200-400 device network. | 4. | The solution needs to be based on a rational trade-off between several parameters such as performance and cost. |
| | | | **Sustainability** |
| | | 5. | The solution needs to be sustainable. |
| | **Data Processing Requirements** | | **Device Protection** |

| | | | |
|---|---|---|---|
| 5. | Sensor Accuracy: The sensor needs to have adequate accuracy levels to support the research use of this device | 6. | Weather Protection and Damping: The mounting/ case needs to protect device from weather conditions such as snow, rain, and sun light. |
| 6. | Data Visualisation: The solution needs to provide a mode of data interpretation to different user groups in form of visualization. | 7. | Theft Protection: The proposed solution needs to consider the scenario for device theft while brainstorming the mount design and data collection methodology. |

## 4.2 Benchmarking and Critical Metrics

The project team studied various sensor utilized by existing solutions during the literature review and determined target specifications for the Prototype-I as highlighted in Table 2.

Table 2. Performance Evaluation Metrics, Benchmarking, and Target System Specifications [1-6]

| | **Performance Criteria** | **PM2.5 Sensor** | | **Temp. Humidity Pressure Sensor** | | **Ozone Sensor** | | **Target SYSTEM Specifications** |
|---|---|---|---|---|---|---|---|---|
| 1. | Size | 41x41x12 mm | 71x70 x2 mm | 19x19x2.8 mm | 14x18x5.5 mm | 15x15x3 mm | 3x2x2.8 cm | ** |
| 2. | System Weight | 26g | 69g | 5g | 2.4g | 6g | 10g | 200-300g |
| 3. | Power Consumption | 4.5-5.5V | 4.7-5.3V | 3.3-5V | 3.3-6V | - | 5V | 5-9V |
| 4. | Cost | 68CAD | 160CAD | 22CAD | 10CAD | 27CAD | 21CAD | 100-200CAD |
| 5. | Sensor Working Range | 0 – 1000µg/m$^3$ | 0 – 999.9 µg /m$^3$ | -40-80°C | -40-80°C | 20ppm <40°C | 10-1000ppm | -30 to 40°C |
| 6. | Sensor Accuracy | - | 70-98% | +/-3%<br>+/-1hPa +/-1°C | +/-2%<br>+/- 0.5°C | 90% | - | ** |
| 7. | Sensor Precision | +/- 10% | ±15%<br>±10 µg | - | 0.1% / 1°C | - | - | ** |
| 8. | Sensor Spatial Resolution | 1s | 1s | - | - | - | - | 20-50m |

## 4.3 Constraints and Risks Study

The execution of the project plan for the Prototype-I contains several constraints and risks. The following study identifies some of the preliminary constraints for the same.

Table 3. Constraints and Risks Study

| | **Constraint** | **Description** |
|---|---|---|
| 1. | Time | There is a limited time for the process of ideation, conceptual design, system construction, and system testing.<br>a. The lead time for the system components (i.e., sensors, Arduino board, and power supply) pose a potential risk of project overrun.<br>b. The final project completion date is on 31st March 2023, which imposes a potential risk of "No Return" once one approach is selected for the system design. The project deadline will limit the<br>opportunity to test multiple alternative solutions to some extent. |
| 2. | Cost | The budget of the project is limited only to CAD 100-200. It may lead to a hard trade-off between the performance parameters for sensors vs. their cost. |
| 3. | Targeted User Group & Future Implications | The long-term goal of the project caters multiple user groups including public, researchers, and governments. Given the specific requirements for each user group, the decisions made on data measurement and storage poses a risk in context of its optimal usability for the future. |
| 4. | Resources | The project team members having more expertise towards mechanical and electrical engineering, the lack of multi-disciplinary knowledge might act as a constraint on the decision-making process for selecting a solution. |
| 5. | Sensor Calibration | For the accuracy of the data and further implications, it is necessary for the sensors to be calibrated. There are various challenges such as cross sensitivity and drift which may affect the sensor calibration. Moreover, different sensors losing calibration over the time at different rate may act as a challenge. |

# 5. Prototype Framework

A detailed literature study of similar existing systems was conducted to define the design and technical requirements for the Integrated Sensor Network. Several sensors were selected to be incorporated in the system by analyzing the cost, performance, and operating conditions of each. These requirements were translated into a conceptual design, which aided the development of an electronic circuit to be mounted on the E-bike for collecting real-time data.

## 5.1 Design Requirements

1. Real-time data collection: The system should be able to collect and transmit data from the various sensors in real-time.
2. Accuracy and precision: The sensors used in the system should be accurate and precise enough to provide reliable data for pollution monitoring.
3. Integration and compatibility: The sensors and devices used in the system should be compatible and integrated with each other, such that the data collected can be easily transmitted and analyzed.
4. User-friendly interface: The system should have a user-friendly interface, such as a LCD display, that can provide real-time data measurements.
5. Low power consumption: The system should be designed to minimize power consumption to maximize the battery life of the E-bike.
6. Durability and portability: The system should be durable and designed to withstand the various weather conditions and vibrations induced while riding an E-bike, while also being portable and easy to install and dismount from the bike.
7. Data security and privacy: The system should include measures to ensure the security, anonymity and privacy of the data transmitted and stored.

## 5.2 Technical requirements

1. Spatial resolution: The device must cumulate the data with a spatial resolution of 20-30 meters.
2. Marginal system dimensions: The overall system dimensions, including the mounting, should be within 20 x 20 x 20cm.
3. System weight: The combined weight of the system should be within 200-300 grams to ensure minimal interference with the performance of the E-bike.

4. Operating conditions: The system should function efficiently in the range of -30°C to 40°C and up to 95% relative humidity.

5. Accuracy: The accuracy of the data collected by the system should be withing the range of ±10%.
6. Response time: The system response time must be within 10-15 seconds.
7. Calibration: The sensors used in the system should be calibrated against cross sensitivity and drift over time, which may affect the accuracy of the collected data.

## 5.3 Conceptual Design

The conceptual design for Prototype-I is presented in the figure below. The entire system can be divided into the following 6 sub-systems.

**Power source**

The system derives power from the E-bike battery. A step-down DC to DC transformer is used to regulate and stabilize the supply of power to avoid any voltage and current fluctuations.

**Arduino**

The system is controlled by an Arduino microcontroller, which serves as the brain of the assembly. It is integrated with the various sensory modules, communication module, and LCD display module to provide user interface.

**Sensor modules**

These modules are responsible for collecting data pertaining to the level of different pollutants. This is done at a rate specified by the Arduino microcontroller to maintain a spatial resolution of a minimum of 20 meters.

The sensor modules used to develop the first prototype are as follows:

a. Temperature and humidity sensor – DHT22

b. PM2.5 Sensor – PMS7003

c. Ozone sensor – Tubatter -131 MQ131 Detection Module Sensor

d. GPS module – NEO-7M GPS Module + EEPROM with Antenna

The 4 above-mentioned sensors work concurrently to collect pollution data, along with the GPS co-ordinates and timestamp.

### Communication module

The communication module, which in this case is the ESP8266 WIFI Module, helps the system interact with the server. It transfers the collected data from the sensor modules to a remote server for storage and further analysis.

Real-time data measurements are displayed on the LCD display module, which is mounted on the main device.

### Data Storage

The data transmitted by the Wi-Fi module is stored on a remote server. This data can be accessed, monitored, and processed according to user needs.

### Data Analysis and Interpretation

The final objective of the project is to analyze and interpret the data to effectively visualize levels of each pollutant throughout the path traversed by the E-bike. This data can then be plotted on a map to generate a visual representation of pollution levels.

Incorporating more such devices and integrating them would help form a network that spreads across all the paths traversed by the numerous E-bikes, providing a visual representation of the pollution levels on every path.

The plotted data would be open for access to the end user for further data interpretation.

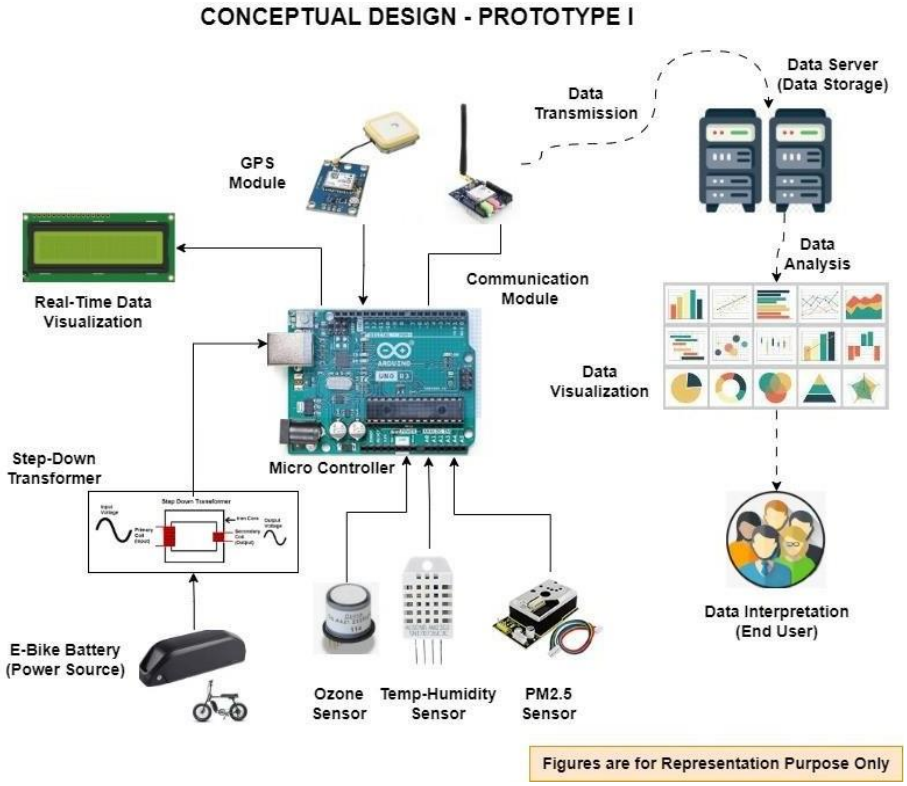


Figure 1. Conceptual Design- Prototype I

Overall, this prototype has the potential to be a valuable tool for environmental monitoring and data analysis.

## 5.4 Potential inadequacies

### 5.4.1 Sensor Calibration

The heavy reliance of the project on the accuracy and precision of the acquired data renders sensor calibration as a crucial aspect of the system.

### 5.4.2 Abnormal spikes in data

Abnormal spikes relate to any value of the data that does not align with the values from the preceding or following data points. This may be caused by temporary stationary sources that the system may be in proximity to, for example, exhaust from a stationary car or truck.

### 5.4.3 Maintaining anonymity of data

As the data provides stamps of time and GPS co-ordinates, it would be important to make sure the data stays anonymous to aid user safety.

## 5.5 Prototype Testing

The Prototype-I functionalities will be tested by performing experimental data measurement followed by data recording and data visualization.

### 5.5.1 Response Time Measurement

The response time of the Temperature/ Pressure sensor will be measured through the stimuli of temperature fluctuation by entering and leaving CBY building.

### 5.5.2 Sensor Data Accuracy Verification

The pollution data to be compared with the daily averages available from the satellites/ weather stations to approximate the accuracy of the data measurements.

### 5.5.3 Modular system testing

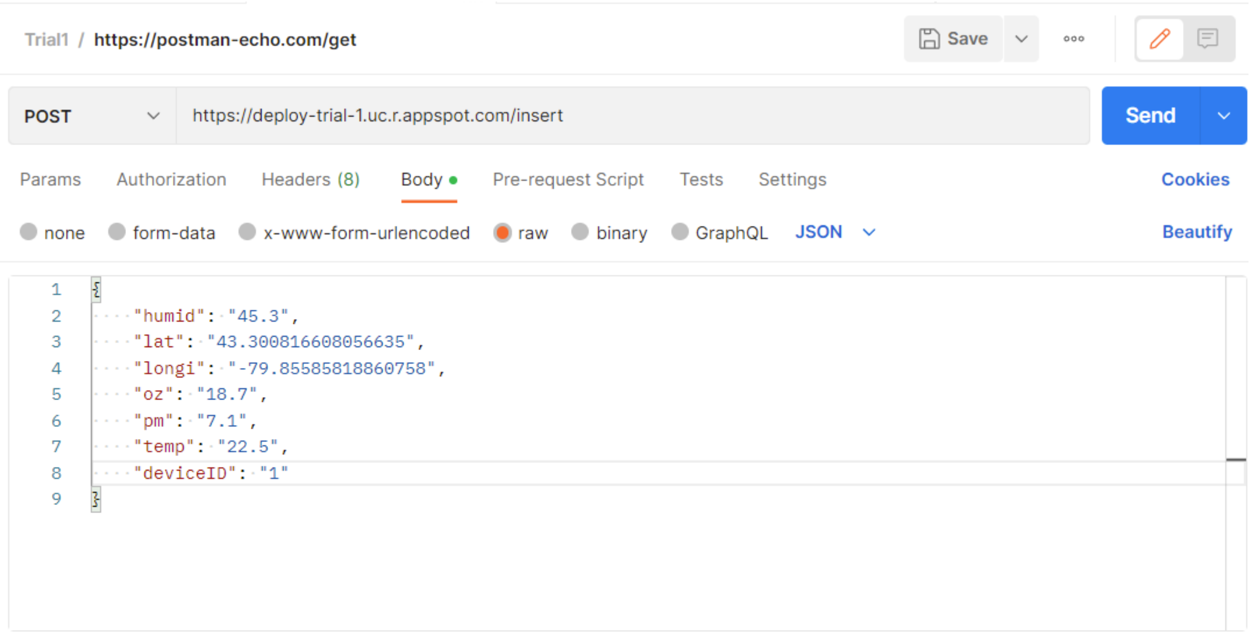
Different parts of the projects were developed simultaneously. As such, the testing was divided into three segments:

1. Data Storage
2. Data Visualization
3. Hardware

The data storage consists of the REST API and the MySQL database. These two were developed in conjunction and the consist of two endpoints:

1. /insert
2. /show

The first endpoint receives data from the Arduino and stores it in the MySQL database. The second endpoint interacts with the visualization code and fetches data based on the request. The “/insert ” endpoint was tested using the POSTMAN tool. This tool lets you send a POST request with a customized request body to any endpoint you want. Using this tool, we were able to send generated data to the endpoint and verify if the data would be stored in the database. The test was successful and the data was being stored successfully.



POST https://deploy-trial-1.uc.r.appspot.com/insert

```
{
    "humid": "45.3",
    "lat": "43.300816608056635",
    "longi": "-79.85585818860758",
    "oz": "18.7",
    "pm": "7.1",
    "temp": "22.5",
    "deviceID": "1"
}
```

Figure 2. Modular Testing of Data Storage using Postman

The visualization code was developed after the REST API and the database were deployed. The code was tested by storing generated data in the database and plotting the data on the map object. This test also proved successful and we were able to generate the map successfully.

## 5.6 Metrics for prototype quality testing

The Table 4 illustrates the performance criteria and target system specifications. These parameters will be used to verify the requirements against the measured system specifications. Following Prototype 1, the following improvements were made and tested.

### 5.6.1 Data Anonymity

Since GPS data is collected through the device, security of the data is of utmost importance even for a prototype. Changes were made to the API code and MySQL database structure to include only Device ID rather than the user information. This acts as an additional layer of protection keeping the identity of the user anonymous. Furthermore, the access credentials for the database were removed from the API and stored offline, giving access only to select individuals from the team. This step further acts a barrier keeping data safe. While testing this implementation is not feasible, as it follows standard industry practices for data protection, data anonymity can be assured.

Table 4 Metrics for Prototype-I Quality Testing

| | **Performance Criteria** | **Target SYSTEM Specifications** | **Measured SYSTEM Specifications** |
|---|---|---|---|
| 1. | Size | Within 20x20x20cm | 14.8cm x 10.5cm x 10.4 cm |
| 2. | System Weight | 200-300g | 168g + electronics |
| 3 | Power Consumption | 5-9V | |
| 4 | Cost | 100-200CAD | ≈175 CAD |
| 5 | Operating Conditions | 30°C to 40°C | |
| 6 | Sensor Accuracy | Up to 95% | |
| 7 | Sensor Precision | ** | |
| 8 | Sensor Spatial Resolution | 20m | |
| 9 | Sample Rate | 10s | |

### 5.6.2 Sensor Calibration, Map API and REST API integration

The defective sensors were replaced and the calibration for the new sensors is pending. This will be done in conjunction with the integration of other sensors into the system.

Another critical step is the integration of the map API and the REST API. Defining the queries that fetch the data as requested to be overlayed on the map is a critical part of the project and is a key deliverable for the final prototype. Connecting the Arduino Uno to WiFi and establishing the connection between the map API and REST API is done, the entire pipeline can be tested.

### 5.6.3 Improvement Plan for Prototype-II

Due to defects in prototype-I, the casing for prototype-II has been redesigned and printed. The casing consists of three layers with holes for airflow. In prototype-I, the holes on the first and third layers were on the upper half, while the second layer had holes on the lower half. However, this design did not function effectively. To improve airflow, we added more holes to each layer, offsetting the holes on the second layer to prevent rain from entering. We also included additional holes on the back panel for further airflow and adjusted the casing dimensions to ensure a perfect fit.

We replaced two malfunctioning sensors and tested them. The PM2.5 sensor is functioning correctly, but the Wi-Fi sensor is not working properly.

A good way to solve the Wi-Fi problem is to replace it with a board that has built-in Wi-Fi or to change the Wi-Fi shield. Another option is to switch to Bluetooth for communication, but this would require a lot of time. As we don't have much time left at this stage, this is not an option for us.

## 5.7 Updated Problem Statement and SMART Goals

### 5.7.1 Updated Problem Statement

"Prototyping and Development of an Integrated Sensory Network for Real-Time Pollution

Measurement and Data Visualization on E-Bikes"

### 5.7.2 SMART Goals:

**Specific**

Develop a wireless sensor network that can collect and transmit real-time data on ozone, temperature, humidity, PM 2.5, and GPS location from an E-bike, with data storage and visualization capabilities.

**Measurable**

Achieve the spatial resolution with an accuracy of 20 meters for the collected data by testing the system at various bike speeds.

Calibrate the pollution sensors under operating conditions to measure precise and accurate levels of pollutants under consideration.

**Achievable**

Develop a working prototype-I of the system that can be installed and operated on an E-bike within 1 month (March 15, 2023) using off-the-shelf components and open-source software.

**Relevant**

Create a pollution monitoring and visualization system that provides a quantified understanding of pollution levels at the local level to be used by researchers, or environmental analysts to determine and inspect sources or pollution.

Concurrently meet the needs of urban commuters who want to monitor and reduce their exposure to air pollution and contribute to a cleaner environment.

**Time-bound**

Test the prototype-I system by March 24th, 2023; verify its performance and usability and identify the areas for further development and mass production.

# 6. Prototypes – I & II

For the implementation of the first prototype, several sub-systems were made and the code for the various subsystems were compiled and run to test the functionality of different sensors. The physical and software prototypes are described as follows:

## 6.1 Physical Prototypes

The circuitry for the system includes the Arduino Uno Rev3 module, DHT22, MQ131, PMS5003, GPS (Neo-7M) module, LCD, and the Wi-Fi (ESP-8266) module. The power source (the e-bike battery is not considered in the first prototype) for the sensors is supplied from the Arduino itself.

The basic circuit diagram was developed using the KiCAD software and it is as follows:

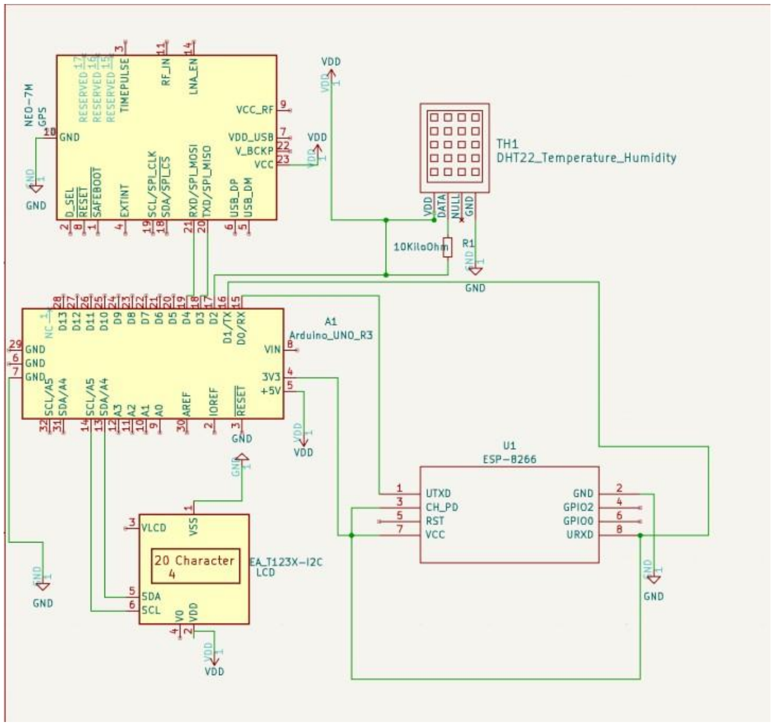


Figure 3 Circuit Diagram

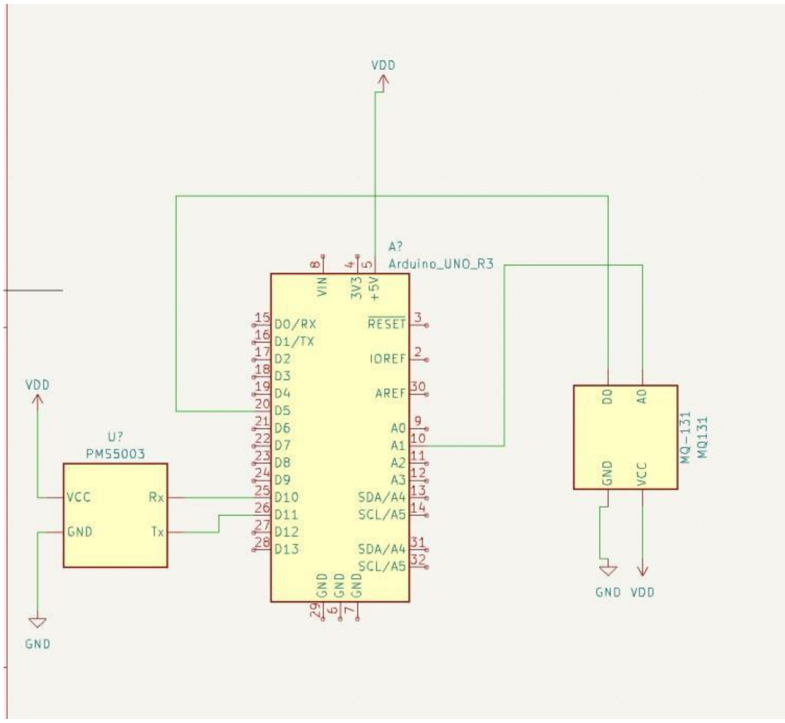


Figure 4 Circuit Diagram

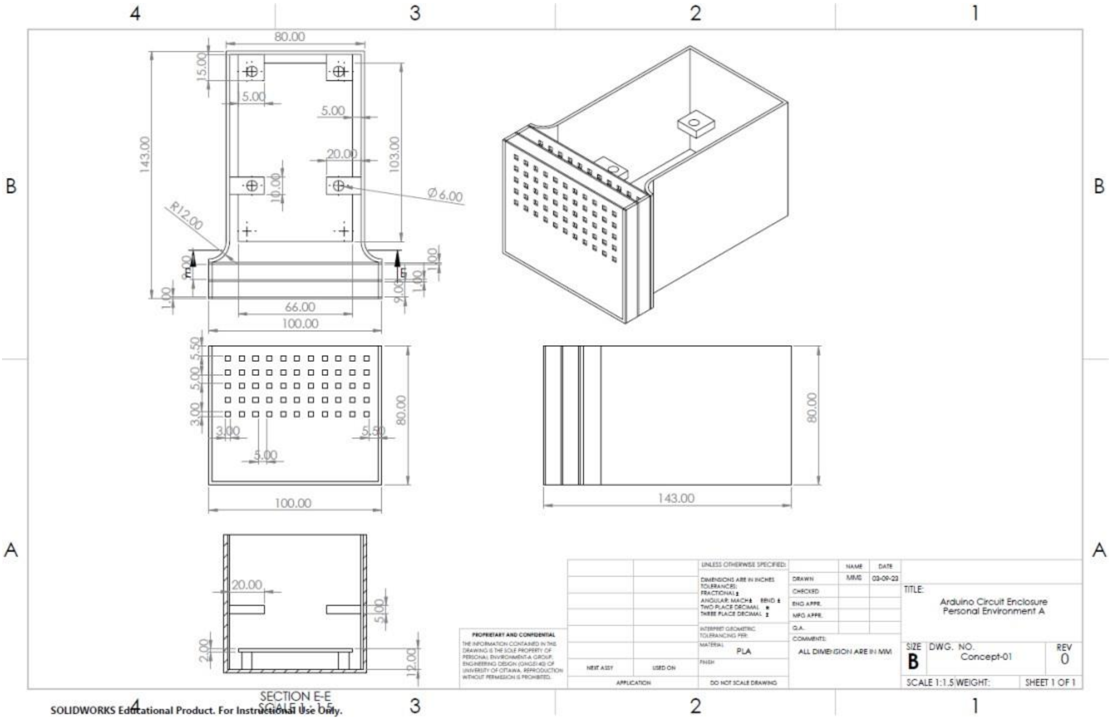


Figure 5 Arduino Circuit Enclosure Design (Prototype-I)

The Arduino casing was designed in SolidWorks keeping in mind the features such as good air flow for the proper functioning of sensors and circuit's safety from the external environmental factors like rain or snow.

Based on these consideration, the subsequent design changes were made to accommodate the circuitry with ensuring good air flow. The prototype-II for the outer casing can be visualized as given in Figure 6-7.

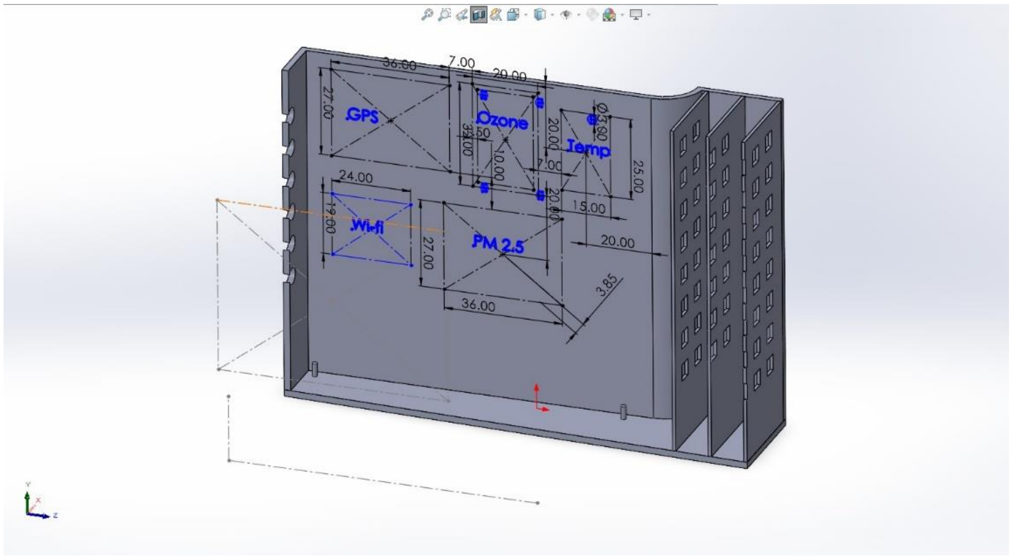


Figure 6 Arduino Circuit Enclosure Design- Sensor Arrangement (Prototype-II)

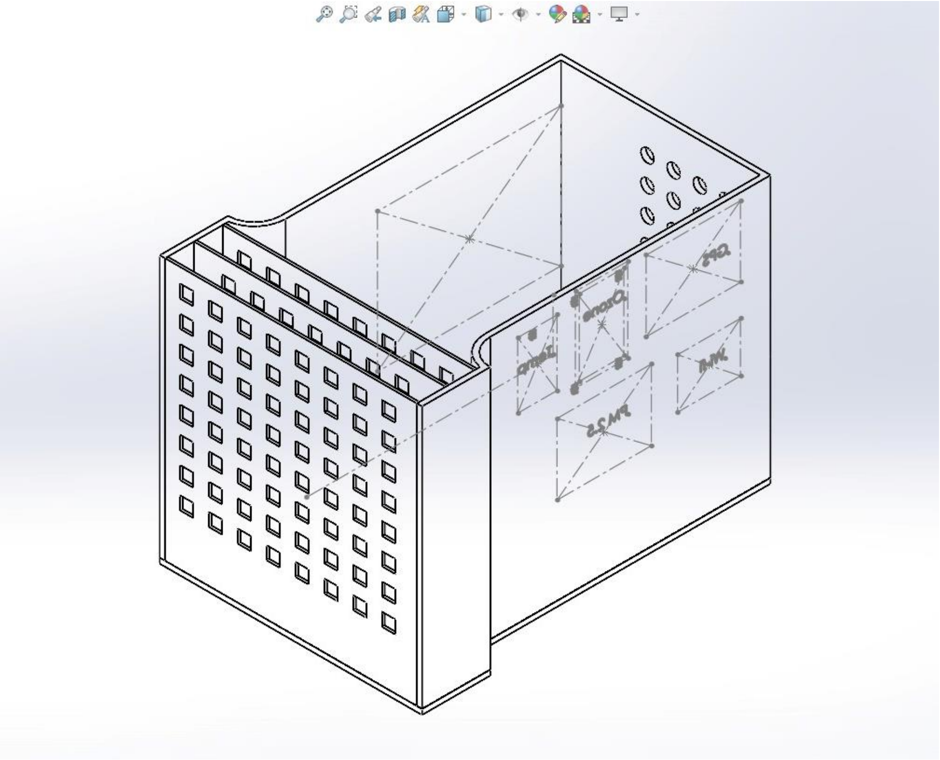
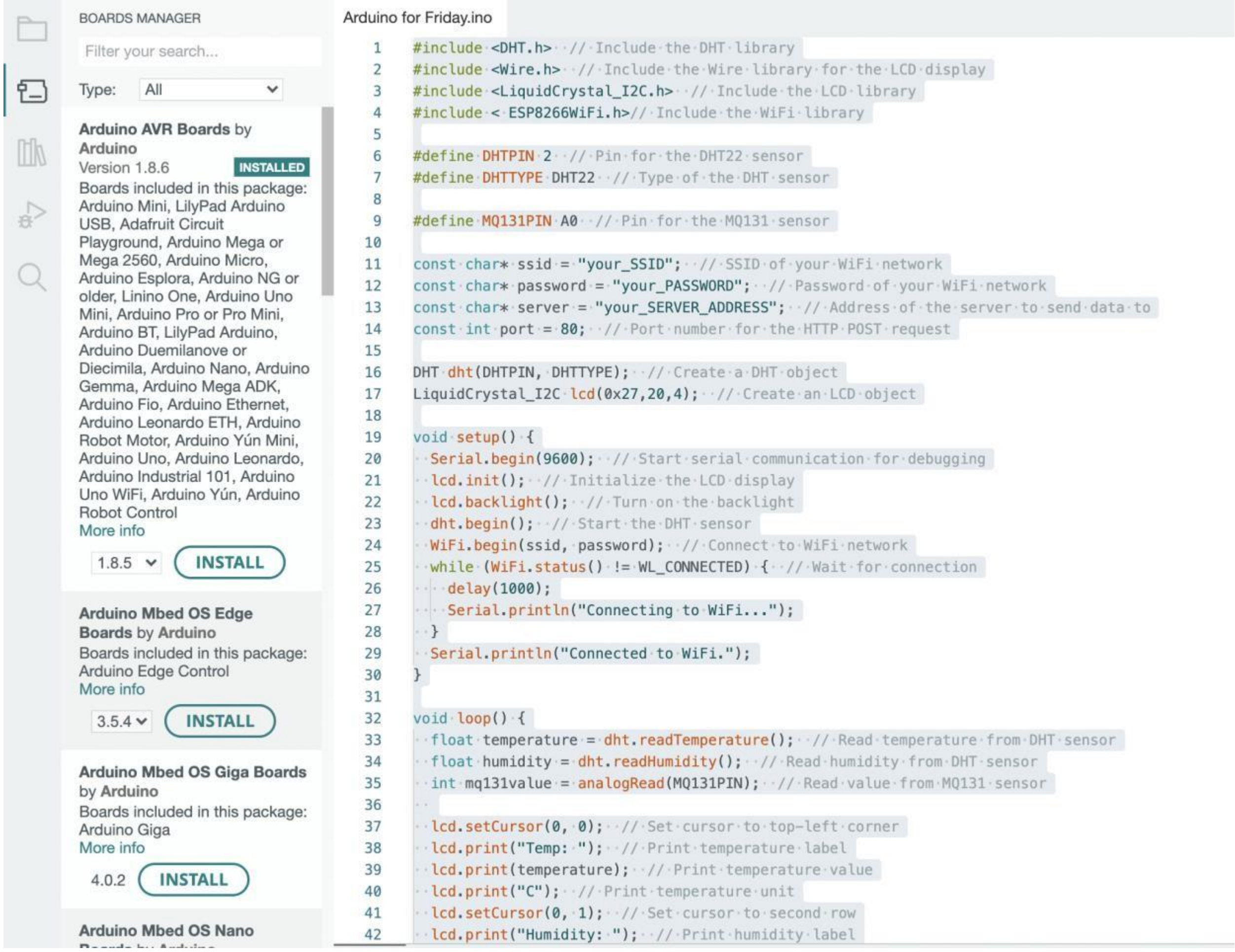


Figure 7 Arduino Circuit Enclosure Design (Prototype-II)

## 6.2 Software Prototypes

The Arduino IDE is used for loading the code into the Arduino module. The required libraries were installed, and the program was compiled and run to check the functionality of individual sensors and the prototypes. Sample of the code is shown in the figure below.



**Arduino AVR Boards** by Arduino
Version 1.8.6
Boards included in this package:
Arduino Mini, LilyPad Arduino USB, Adafruit Circuit Playground, Arduino Mega or Mega 2560, Arduino Micro, Arduino Esplora, Arduino NG or older, Linino One, Arduino Uno Mini, Arduino Pro or Pro Mini, Arduino BT, LilyPad Arduino, Arduino Duemilanove or Diecimila, Arduino Nano, Arduino Gemma, Arduino Mega ADK, Arduino Fio, Arduino Ethernet, Arduino Leonardo ETH, Arduino Robot Motor, Arduino Yún Mini, Arduino Uno, Arduino Leonardo, Arduino Industrial 101, Arduino Uno WiFi, Arduino Yún, Arduino Robot Control
More info

**Arduino Mbed OS Edge Boards** by Arduino
Boards included in this package:
Arduino Edge Control
More info

**Arduino Mbed OS Giga Boards** by Arduino
Boards included in this package:
Arduino Giga
More info

**Arduino Mbed OS Nano**

```
#include <DHT.h>  // Include the DHT library
#include <Wire.h>  // Include the Wire library for the LCD display
#include <LiquidCrystal_I2C.h>  // Include the LCD library
#include < ESP8266WiFi.h>// Include the WiFi library

#define DHTPIN 2  // Pin for the DHT22 sensor
#define DHTTYPE DHT22  // Type of the DHT sensor

#define MQ131PIN A0  // Pin for the MQ131 sensor

const char* ssid = "your_SSID";  // SSID of your WiFi network
const char* password = "your_PASSWORD";  // Password of your WiFi network
const char* server = "your_SERVER_ADDRESS";  // Address of the server to send data to
const int port = 80;  // Port number for the HTTP POST request

DHT dht(DHTPIN, DHTTYPE);  // Create a DHT object
LiquidCrystal_I2C lcd(0x27,20,4);  // Create an LCD object

void setup() {
  Serial.begin(9600);  // Start serial communication for debugging
  lcd.init();  // Initialize the LCD display
  lcd.backlight();  // Turn on the backlight
  dht.begin();  // Start the DHT sensor
  WiFi.begin(ssid, password);  // Connect to WiFi network
  while (WiFi.status() != WL_CONNECTED) {  // Wait for connection
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi.");
}

void loop() {
  float temperature = dht.readTemperature();  // Read temperature from DHT sensor
  float humidity = dht.readHumidity();  // Read humidity from DHT sensor
  int mq131value = analogRead(MQ131PIN);  // Read value from MQ131 sensor
  
  lcd.setCursor(0, 0);  // Set cursor to top-left corner
  lcd.print("Temp: ");  // Print temperature label
  lcd.print(temperature);  // Print temperature value
  lcd.print("C");  // Print temperature unit
  lcd.setCursor(0, 1);  // Set cursor to second row
  lcd.print("Humidity: ");  // Print humidity label
```

Figure 8. Sample Code

## 6.3 Back-End Architecture

### 6.3.1 Data Storage

The purpose of the backend infrastructure is to store the information so that it is readily available when it needs to be viewed. This requires an API to handle the communications, and a database to store the sensor data.

API stands for Application Programming Interface. An API is a software that allows other software to communicate with each other. For this project, the Arduino Uno communicates with the API which would then communicate with the device requesting the sensor data. There are several types of APIs such as SOAP, RPC, WebSocket API, and REST API. This project uses a REST API because it is a simple and lightweight approach that can easily transfer data in a standardized format such as JSON over HTTP protocol. Additionally, REST APIs provide flexibility in terms of the types of requests and responses that can be made, allowing for efficient and scalable communication between devices and servers.

An API is a software and so can be created using any programming language of choice. There are several available frameworks for each language that can be used to create an API to receive and store sensor data from an Arduino UNO. For this project, the API was created using the Flask micro web framework, which is written in Python. The Arduino UNO sends a POST request with the sensor data stored in JSON format in the request body. This is routed to a specific endpoint that stores this database.

The other crucial component is the database. MySQL was selected because there is an abundance of literature, and it is easy to setup. MySQL uses queries to store, retrieve, alter, or view information from a database. When the endpoint receives the sensor data, a function is called which sends a query to the MySQL database to store the sensor data received. When the data needs to be viewed, a client would send a request to a different endpoint which would trigger a different function. This function queries the database to retrieve specific information based on the filtering applied.

The API and the database are hosted on the Google Cloud Platform. The Arduino UNO sends the data to the API which stores it in the MySQL database. When the data needs to be viewed on a map, the client requests the API for the data. The API retrieves it from the database and sends it back in JSON format. This is illustrated in the image below.

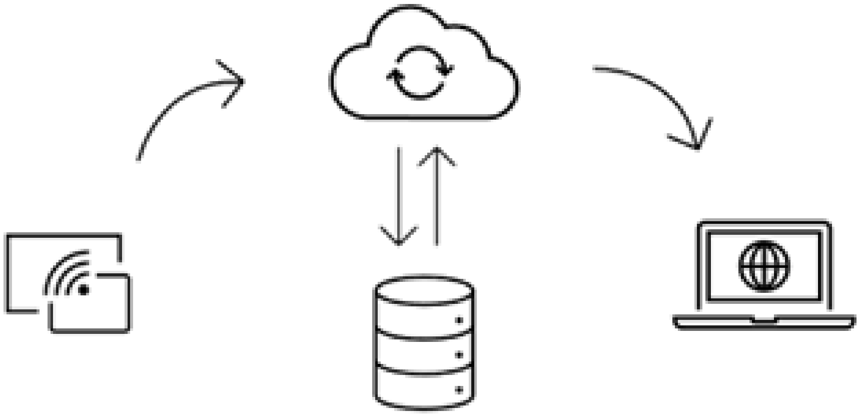


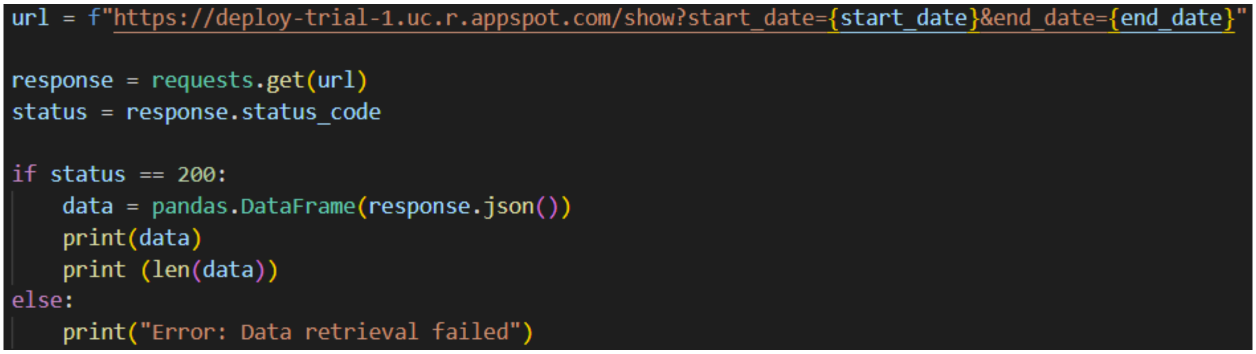
Figure 9 Back-End Architecture

### 6.3.2 Data Visualization

This segment of the project retrieves the data from the MySQL database hosted on Google Cloud Platform and overlays the data on a map. Python was elected to do this as there are several readily available libraries. Three libraries were used to retrieve the data and overlay it. There libraries are as follows:

1. Requests
2. Pandas
3. Folium

Requests is a library that provides functions and objects that can be used to handle HTTP communication. In this code, the library is used to send a GET request to the REST API. This request sends a get request to the specific endpoint with two time and date values. These time and date values correspond to 12 AM from the previous day and 11:59 PM of the previous day. This is so that the GET request retrieves data entries made during the whole of the previous day. This can be seen in the following section of the code where the data attached in the request body of the response to the GET is stored as DataFrame variable.



```
url = f"https://deploy-trial-1.uc.r.appspot.com/show?start_date={start_date}&end_date={end_date}"

response = requests.get(url)
status = response.status_code

if status == 200:
    data = pandas.DataFrame(response.json())
    print(data)
    print (len(data))
else:
    print("Error: Data retrieval failed")
```

Figure 10. Code to send get request to “\show” endpoint

The next library is Pandas. This library is used for data manipulation and analysis. It contains functions that can be used to handle data in different forms. This is particularly useful for this project because the data is stored in KEY-VALUE pairs in JSON format. By using Pandas, we can store the data in the request body in a table format (DataFrame) and extract data from each row and plot it accordingly. The third library is Folium. This library provides map objects with multiple customization options. It also allows us to define markers based on certain user specified criteria. Another useful feature that is used in the project is the ability to define a legend. By using these features in conjunction with the libraries described earlier, we can request data between a specific date range from the database, convert the data from JSON format to DataFrame format, and finally plot the data on a map object with customized markers that correspond to different datapoints. By leveraging different aspects of these three very different libraries, we created the following map below.

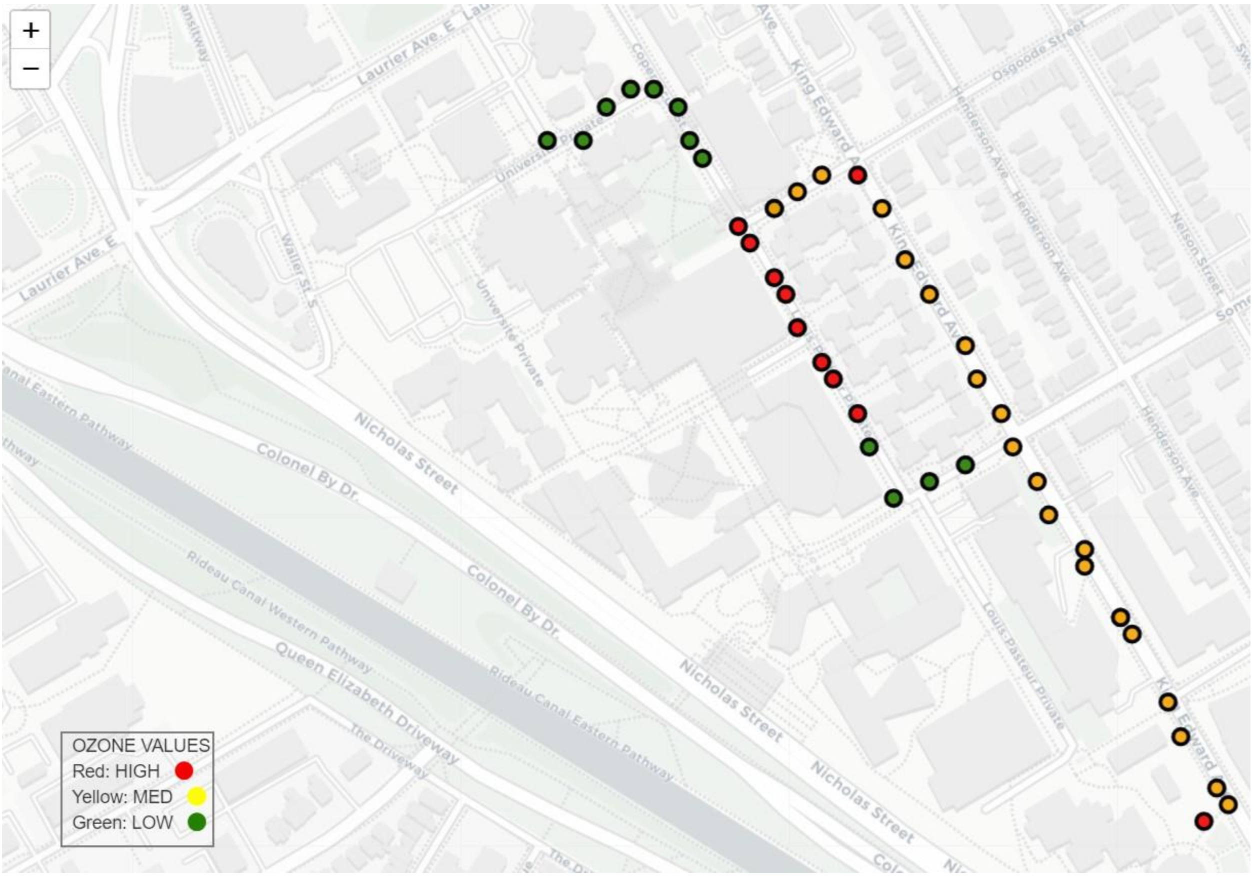


Figure 11. Data Visualization on Map

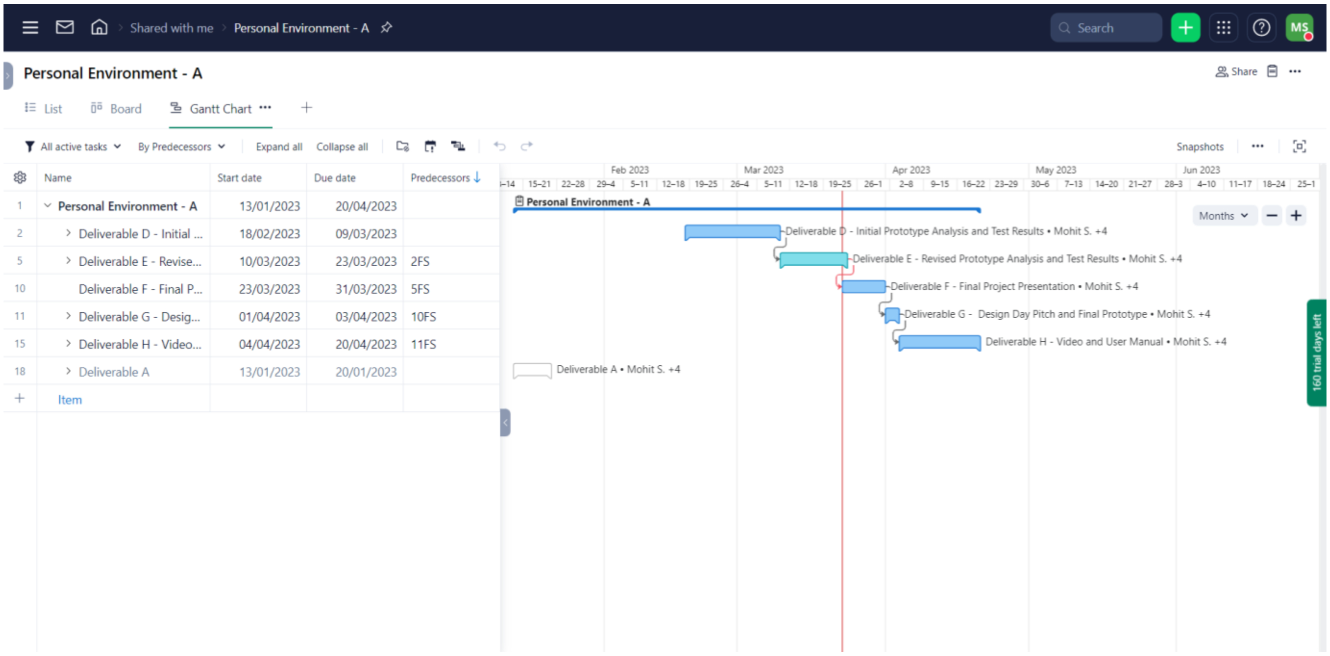
# 7. Detailed Bill of Materials (BOMs) for the Project

Table 5 Detailed Bill of Materials (BOMs)

| Part no. | Description | Model | Supplier | Quantity | Cost Per Piece ($) | Total Cost($) | Comments |
|---|---|---|---|---|---|---|---|
| 1 | Ozone Sensor | Tubatter – 131 MQ131 | Amazon | 1 | 28 | 28 | - |
| 2 | Temperature Humidity Sensor | DHT22 | Simcoe DIY | 1 | 10 | 10 | - |
| 3 | PM2.5 Sensor | PMS7003 | Amazon | 1 | 27 | 27 | Replaced |
| 4 | Arduino Microcontroller | UNO R3 | Makerspace | 1 | 0 | 0 | - |
| 5 | Arduino Cable |  | Makerspace | 1 | 0 | 0 | - |
| 6 | DC to DC Step Down Transformer |  |  | 1 | 17 | 17 | - |
| 7 | LCD Display |  |  | 1 | 16 | 16 | - |
| 8 | GPS Module | NEO-6M GPS Module + EEPROM with Antenna |  | 1 | 19 | 19 | Replaced |
| 9 | Wi-Fi Module | ESP8266 |  | 1 | 7 | 7 | - |
| 10 | 9V Battery |  | Amazon | 4 | 15 | 15 | - |
| 11 | Battery Connector |  |  | 1 | 2 | 2 | Not acquired |
| 12 | Resistor (5.1KΩ) | - | Makerspace | 10 | 0 | 0 | - |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 13 | Resistor (10KΩ) | - | Makerspace | 10 | 0 | 0 | - |
| 14 | Jumper Cable(MF) | - | Makerspace | 15 | 0 | 0 | - |
| 15 | Jumper Cable (MM) | - | Makerspace | 15 | 0 | 0 | - |
| 16 | Jumper Cable(FF) | - | Makerspace | 15 | 0 | 0 | - |
| 17 | Breadboard | GS-400 | Makerspace | 1 | 0 | 0 | - |
| 18 | Shipping Cost | | | | | 30 | |
| | Total | | | | | 171 | |
| | Total (incl. tax) | | | | | 194 | |

# 8. Updated Project Plan



| | Name | Start date | Due date | Predecessors |
|---|---|---|---|---|
| 1 | Personal Environment - A | 13/01/2023 | 20/04/2023 | |
| 2 | Deliverable D - Initial ... | 18/02/2023 | 09/03/2023 | |
| 5 | Deliverable E - Revise... | 10/03/2023 | 23/03/2023 | 2FS |
| 10 | Deliverable F - Final P... | 23/03/2023 | 31/03/2023 | 5FS |
| 11 | Deliverable G - Desig... | 01/04/2023 | 03/04/2023 | 10FS |
| 15 | Deliverable H - Video... | 04/04/2023 | 20/04/2023 | 11FS |
| 18 | Deliverable A | 13/01/2023 | 20/01/2023 | |
| + | Item | | | |

Figure 12. Updated Project Plan-WRIKE

# 9. Sustainability Aspect

## 9.1 United Nations Sustainable Development Goals (UNSDGs)

The system aligns with the UN sustainability goals [7] of Good Health and Well-being as it gives the end-users an idea of the air quality around them. Since the device and the data collected may be used for research purposes, it promotes quality education. Two other goals that the device achieves are Climate Action and Sustainable cities and Communities respectively.

3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
11 SUSTAINABLE CITIES AND COMMUNITIES
13 CLIMATE ACTION

Figure 13. UN Sustainability Goals

While providing a solution for prototyping and development of an Integrated Sensory Network for real-time pollution measurement and data visualization, this project also looks at the bigger picture of achieving objectives listed under the United Nations Sustainable Development Goals (UNSDGs).

The developed system aligns with the below-listed UNSD Goals and while simultaneously providing end-users with a visual representation of the air quality around them.

Goals listed under UNSDG that are addressed in our project:

### 9.1.1 Good Health and Well-Being – GOAL 3

**[3.9]** This goal focuses on significantly reducing the number of deaths and illnesses due to the presence of harmful chemicals causing pollution and contamination of air, water and soil by the year 2030. Goal 3.9.1 is attributed to mortality rate pertaining to household and ambient air pollution.

Our project reinforces this goal by providing a network of sensors to monitor air pollution levels locally and visualize it on a map so end-users can commute using alternative routes along which air pollution levels are minimum, consequently reducing the adverse impact of pollution.

### 9.1.2 Quality Education – GOAL 4

**[4.7]** This goal ensures providing knowledge and skills needed to promote sustainable development through education and awareness through a culture of global citizenship and appreciation of culture’s contribution to sustainable development.

The developed system, if promoted for use within schools, colleges and universities would raise awareness regarding air pollution through community engagement and inclusion.

### 9.1.3 Sustainable Cities and Communities – GOAL 11

**[11.6]** The prime focus of this goal is to reduce the population weighted environmental impact of cities by looking closely at air quality and waste management. Goal 11.6.2 specifically looks at annual levels of fine particulate matter in cities.

The developed system allows us to closely monitor local pollution levels accurately with a resolution of 20-50 meters.

### 9.1.4 Climate Action– GOAL 13

**[13.2]** This goal (13.2) looks to integrate climate change measures into national policies, strategies and planning. 13.2.2 specifically looks at total annual greenhouse gas emissions.

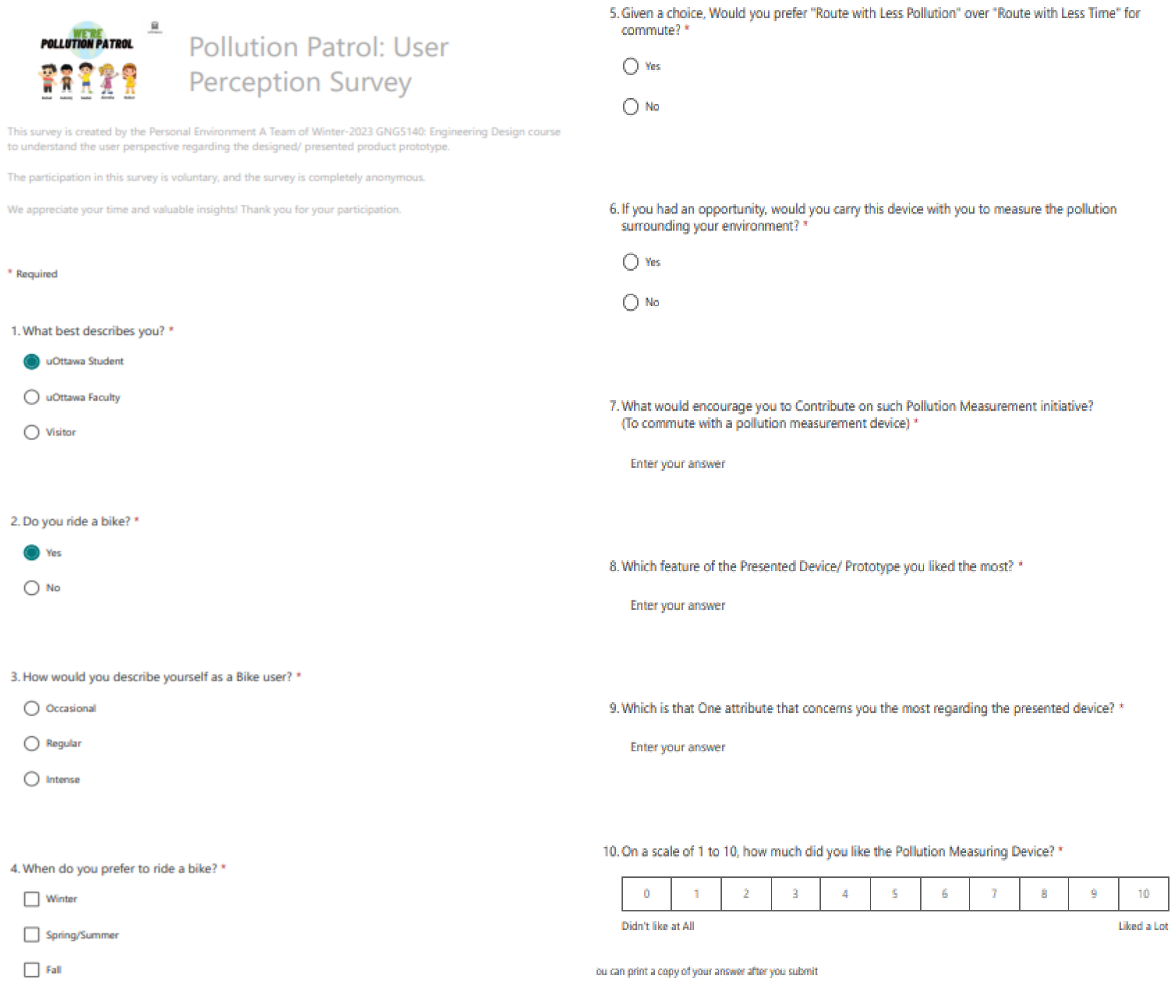
**[13.3]** This goal focuses on improving education and awareness on climate change mitigation, adaptation, impact reduction and warning. 13.3.1 converges on the importance of education for sustainable development.

Since the system provides a detailed representation of pollution levels, it also provides a basis for analyzing sources of pollution, which in turn allows us to draft environmental policies and also curb sources responsible for a high degree of environmental pollution.

## 9.2 Community involvement

The project would ensure community involvement to tackle the greater issue of environmental pollution by motivating users to track and monitor pollution levels on routes used by general public for daily commute.
The project emphasizes the “people as partner” model, according to which, the impact can only be created by closely associating communities with a sustainability initiative.

To acquire user-perspective feedback, the project team carried out a survey to identify the aspects critical to customers and the motivation factors which would drive this initiative.



Pollution Patrol: User Perception Survey

This survey is created by the Personal Environment A Team of Winter-2023 GNG5140: Engineering Design course to understand the user perspective regarding the designed/ presented product prototype.

The participation in this survey is voluntary, and the survey is completely anonymous.

We appreciate your time and valuable insights! Thank you for your participation.

* Required

1. What best describes you? *

● uOttawa Student

○ uOttawa Faculty

○ Visitor

2. Do you ride a bike? *

● Yes

○ No

3. How would you describe yourself as a Bike user? *

○ Occasional

○ Regular

○ Intense

4. When do you prefer to ride a bike? *

□ Winter

□ Spring/Summer

□ Fall

5. Given a choice, Would you prefer "Route with Less Pollution" over "Route with Less Time" for commute? *

○ Yes

○ No

6. If you had an opportunity, would you carry this device with you to measure the pollution surrounding your environment? *

○ Yes

○ No

7. What would encourage you to Contribute on such Pollution Measurement initiative? (To commute with a pollution measurement device) *

Enter your answer

8. Which feature of the Presented Device/ Prototype you liked the most? *

Enter your answer

9. Which is that One attribute that concerns you the most regarding the presented device? *

Enter your answer

10. On a scale of 1 to 10, how much did you like the Pollution Measuring Device? *

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

Didn't like at All — Liked a Lot

ou can print a copy of your answer after you submit

Figure 14. User Perception Survey Form

The project also allows the collection of useful data accurate in time and space to be used by researcher for further study in this domain.

## 9.3 Material usage and Recycling

The primary casing for the device is 3D printed using poly-lactic acid (PLA) which is a biodegradable thermoplastic made from sustainable resources like corn-starch and sugarcane. Since it is a thermoplastic, it can also be melted and used to make other products using injection molding or 3D-printing.

The Arduino microcontroller and level sensors used in the device contain electronic components which can be recycled using electronic waste recycling programs or facilities. The plastic elements of the sensors can be recycled using plastic recycling programs. Additionally, the electronic components may contain hazardous materials like lead or mercury which would require special handling and disposal procedures.

## 9.4 Energy Consumption

Table 6 Power Consumption of Electronic Components

| Component | Voltage (max.) | Current (max.) | Power |
|---|---|---|---|
| Arduino UNO Rev2 | 5 V | 200-250 mA | 1 – 1.25 W |
| GPS Sensor | 2.7 - 3.6 V | 45 mA | 0.12 – 0.16 W |
| Ozone Sensor | 4.5 – 5 V | 150 – 165 mA | 0.675 – 0.825 W |
| PM2.5 Sensor | 4.5 - 5.5 V | 100 mA | 0.45 – 0.55 W |
| Temperature-Humidity Sensor | 3.3 – 6 V | 1 – 1.5 mA | 0.003 – 0.09 W |
| - | - | - | Total: 2.25 – 3 W |

The system uses low-powered sensors to reduce the power consumption of the overall system.
By reducing energy consumption, we can reduce greenhouse gas emissions and mitigate
climate change, which is an important consideration for environmental sustainability.

# 10. Usability Aspect

Jason Nielson describes 10 general principles to define the usability of interactive systems called “heuristics”.  They broadly specify usability guidelines for any system design.

1. Visibility of system status
   The system uses an LCD screen mounted on the top section of the device. This provides the user with real-time instantaneous values for the level of each pollutant.
   However, for future scope, the back-end code fed to the microcontroller can be modified to display any sort of system failure on the LCD screen.

2. <u>Match between the system and the real world</u>

   Since most elements of the system are automated, the two fundamental aspects of user interaction would be i. LCD screen mounted on the device, and ii. Data visualization on a map

   The LCD screen prints out values for the level of each pollutant which is easy for the user to understand.

   The map used for visualization provides a dropdown "selector" on the top right corner which lets the user select the pollutant to be represented on the map using layers. It also uses a legend view to make the data points more comprehensible for users.

   Clicking on the data points further provides a pop-up with the value corresponding to that particular point for each pollutant.

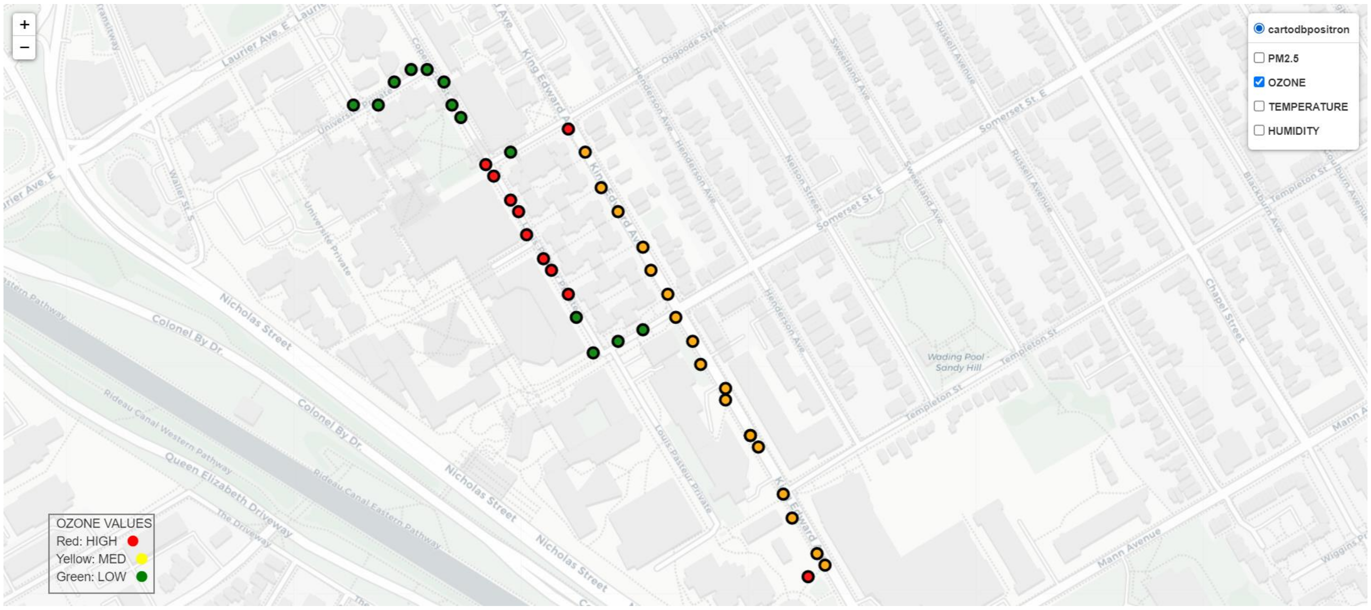
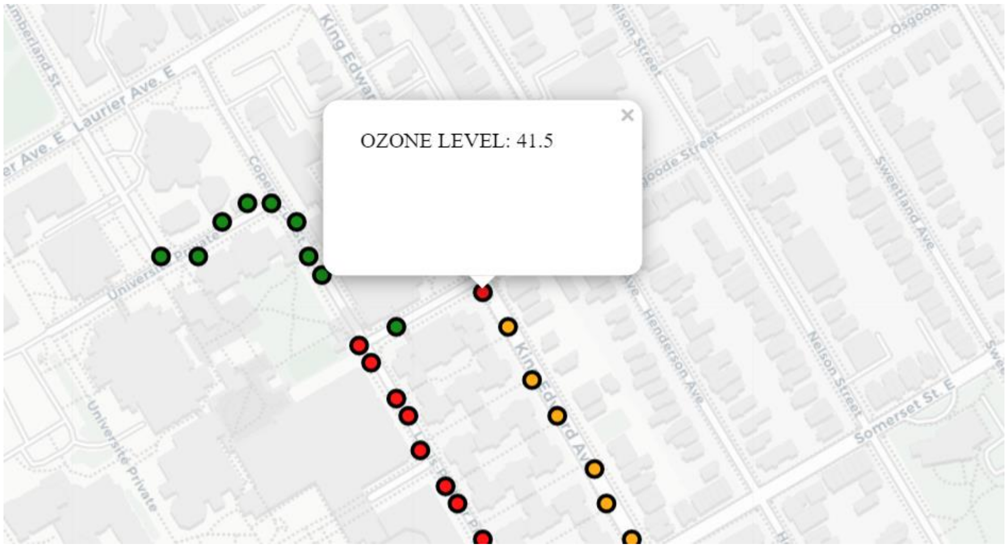


Figure 15. Visualizing pollution level data on map



OZONE LEVEL: 41.5

Figure 16. Text pop-up displaying value corresponding to data-point

3. User control and freedom
   The minimalist design demands minimum user engagement while providing a detailed representation of the collected data. This minimizes/eliminates the scope for errors while using the application.

4. Consistency and standards
   The user-interface uses self-explanatory and non-technical jargon to make the application easy-to-use for the end user. Once the testing phase is complete, some changes would be made in the back-end codes to provide the user with units for each level of pollutant.

5. Error prevention
   The system has been designed in a way to prevent error-prone conditions for the user in the first place. The user-interface uses helpful constraints and good defaults while also eliminating memory burdens.

6. Recognition rather than recall
   The map uses clearly visible options to be selected, which makes interaction easier for the user. The minimalist design further helps to minimize the users' memory load and simplify information recognition.

7. Flexibility and efficiency of use
   The LCD screen directly presents values for individual pollutants making data interpretation easier for end users. The color-coded markers incorporated in the map, along with a legend view aids efficient and prompt user interpretation.

8. Aesthetic and minimalist design
   The visual design focuses only on the essentials and does not consist of unnecessary elements that may distract users from the information they require.
   For future scope, changes can be made to the 3D printed casing to make it look more aesthetically pleasing.

9. Help users recognize, diagnose, and recover from errors
   For future scope, we plan to make changes to the back-end code fed to the micro-controller which would provide any instructions for corrective measure to the user through the LCD screen. For example, notifying the user that the battery needs charging.

10. Help and documentation
    The current system requires very little documentation providing instructions for use. The heavy automation of the system functionality minimizes the user load.

The project uses the above 10 principles as a framework to achieve the following usability goals:

1. Effectiveness: Can the user complete their goals with a high degree of accuracy?
2. Efficiency: How fast can the user get the job done?
3. Engagement: Does the user Find The product pleasant and gratifying to use?
4. Error Tolerance: Can the user effectively recover from an error?
5. Ease of Learning: How quickly can the user learn to use the system?

# 11. Scalability Aspect

## 11.1 Scalability of Production

The prototype uses off-the-shelf sensors that are readily available in large quantities. With regards to design for manufacturability, the casing of the design can be optimized. The prototype casing is made of PLA and manufactured using 3D printers. While this manufacturing technique is ideal for complex designs and rapid prototyping, the manufacturing time makes it a less desirable choice for mass manufacturing. A good manufacturing technique for mass production of the casing is injection moulding. This is a manufacturing technique that injects molten plastic into a mould cavity, which then sets to yield the completed part. This manufacturing technique allows for a high production rate and delivers parts with good accuracy. Additionally, the use of plastic ensures that the casing keeps water away from electronics.

Another key component of designing for production is designing for ease of assembly. The prototype design already uses standoffs to isolate the Arduino Board from any stagnant water in the casing that hasn't drained away. These standoffs act as a place holder making it easy to position the board when assembling. Furthermore, the sensors are currently being held in place using double sided tape, this could be replaced by carefully designing slots in the casing design that fit the sensors perfectly. The slots could be designed so that sensors that fit within align with screw holes, reducing the needed to fix the sensors in place.

The breadboard connecting the Arduino and the sensors is another key component that can be drastically simplified from an assembly standpoint. Breadboards are ideal for prototype as they are cheap and easy to work with. However, when designing for production, the breadboard would have to be replaced with a custom PCB. Once the electronic components and the circuit diagram for the device is constructed, a custom PCB can be designed for the device. Custom PCB manufacturers are easy to find in today's market and for a significantly low cost, the design can be simplified by eliminating the breadboard and all the associated wiring with a custom PCB.

## 11.2 Scalability of Functionality

Replacing the breadboard and the associated wiring with a custom PCB also improves the functionality of the device as it removes numerous failure modes in the device. Replacing a breadboard and wiring with a custom PCB can impact functionality positively. A custom PCB not only provides a more permanent and stable solution but also eliminates numerous failure modes in the device. By implementing a custom PCB, we can ensure that the device is optimized for functionality, reliability, and performance.

The backend infrastructure is another part of the project that has many failure modes. However, unlike the hardware components, the failure modes can be eliminated by implementing error handling components. This includes anticipating, detecting, and resolving errors on the software side of the system. Additionally, failure modes in the hardware components can also be temporarily mitigated to ensure that the whole system does not fall apart if one sensor fails. For example, if the temperature sensor fails on the Arduino, the Arduino code can be altered to incorporate a component that automatically adds a null value to the JSON data.

### 11.3 Scalability of Users

The current design is catered towards the stakeholders directly involved with the project. They are:

i. Community
ii. Government, policy makers, and city planners
iii. Researchers and environmentalists

The prototype was designed to be of a small form factor so that it can be carried around easily. The data transmission was also selected to work well in a metropolitan infrastructure where the device could leverage open Wi-Fi networks. Decisions made with regards to these two factors cater directly to the needs of those in the community and the policy makers in Ottawa. Expanding the user base for this device would require re-evaluating the needs for a larger user base and altering the design decisions for these two factors.

Firstly, because this device was designed to be mounted onto an E-Bike, being light enough so that it wouldn't affect the centre of mass of the bike was an important factor. As such, the thickness of the casing and the infill density when 3D printing the design was also reduced. This reduced the weight from ---- to ---- while still maintaining the structural rigidity to hold the sensors in place. This however would not be a factor of much importance if the target vehicle was a public transport vehicle. The design would change to account for the higher velocities and vibrations that come with mounting the sensor on a bus.

Similarly, because the communication method was optimized to work well in the city, Wi-Fi was selected. This wouldn't be a suitable option if it were to be used in rural areas. An alternative for this would be to use a cellular shield that would work with Arduino. This could be paired with an IoT sim provided by many carriers in Canada allowing for adequate coverage even in remote areas. Another cost-effective alternative that can be used in the city, which wouldn't rely on open Wi-Fi networks, is using LoRaWAN. This was initially considered as the solution for the prototype but the lack of LoRaWAN gateways in Ottawa meant that the coverage would not be great and often limited in most areas. However, when considering the design decisions for mass producing a device such as this, the capital needed to install the infrastructure such as gateways in Ottawa would provide much better coverage for a lower overall cost. This would significantly affect the design decisions when trying to cater for a larger user base.

## 12. Quality Aspect

There are several methods that can be used to evaluate the quality of our design. Some of the commonly used tools are:

1. Failure mode and effects analysis (FMEA)
2. Total Quality Management (TQM)
3. Kaizen
4. Quality Function Deployment (QFD)
5. Six Sigma

The kaizen will be used to evaluate the performance and quality of our prototype. The Kaizen process uses the 5S process to standardize and improve on the existing system. These are:

1. Sort
2. Set in Order
3. Shine
4. Standardize
5. Sustain

By applying these principles to the various components of the system, waste can be identified and systematically eliminated in future iterations of the design. First, by carefully evaluating the sensors in the system and the user needs, the temperature and humidity sensor can be identified as a component that adds an extra feature to the device but does not add any value to the stakeholders involved.

Another component that has a significant impact on the quality of the device is the accuracy of the sensors. Since the purpose of the device is to measure pollution data along with GPS location, any delay with regards to the measurement of any of the pollutants would result in a miscorrelation of the data. The MQ- 131 Ozone sensor contributed towards this miscorrelation the most. The needs to heat up and reach it’s optimal working temperature before it can take measurements at constant intervals. The minimum time between measurements before the optimal temperature is reached is closer to 30 seconds. When operating at the optimal working temperature, this reduced to 18 seconds. The special resolution specified in the user needs is 20 - 50 meters. An E-Bike traveling would have to travel no more than 10 Km/h to be able to collect data with the spatial resolution specified by the stakeholders. This is very slow compared to the speeds that a typical E-Bike would travel at. This is a significant deficiency in the quality of the prototype and alternate sensors need to considered for future iterations.

# 13. References

[1] J. J. Cassano, “Weather Bike: A Bicycle-Based Weather Station for Observing Local Temperature Variations,” *Bull. Am. Meteorol. Soc.*, vol. 95, no. 2, pp. 205–209, Feb. 2014, doi: 10.1175/BAMS-D-13-00044.1.

[2] T. Brandsma and D. Wolters, “Measurement and Statistical Modeling of the Urban Heat Island of the City of Utrecht (the Netherlands),” *J. Appl. Meteorol. Climatol.*, vol. 51, no. 6, pp. 1046–1060, Jun. 2012, doi: 10.1175/JAMC-D-11-0206.1.

[3] J. Gómez-suárez, P. Arroyo, R. Alfonso, J. I. Suárez, E. Pinilla-gil, and J. Lozano, “A Novel Bike-Mounted Sensing Device with Cloud Connectivity for Dynamic Air-Quality Monitoring by Urban Cyclists,” *Sensors 2022, Vol. 22, Page 1272*, vol. 22, no. 3, p. 1272, Feb. 2022, doi: 10.3390/S22031272.

[4] P. Lucas *et al.*, “NanoSen-AQM: From Sensors to Users,” *Int. J. Online Biomed. Eng.*, vol. 16, no. 04, pp. 51–62, Apr. 2020, doi: 10.3991/IJOE.V16I04.11871.

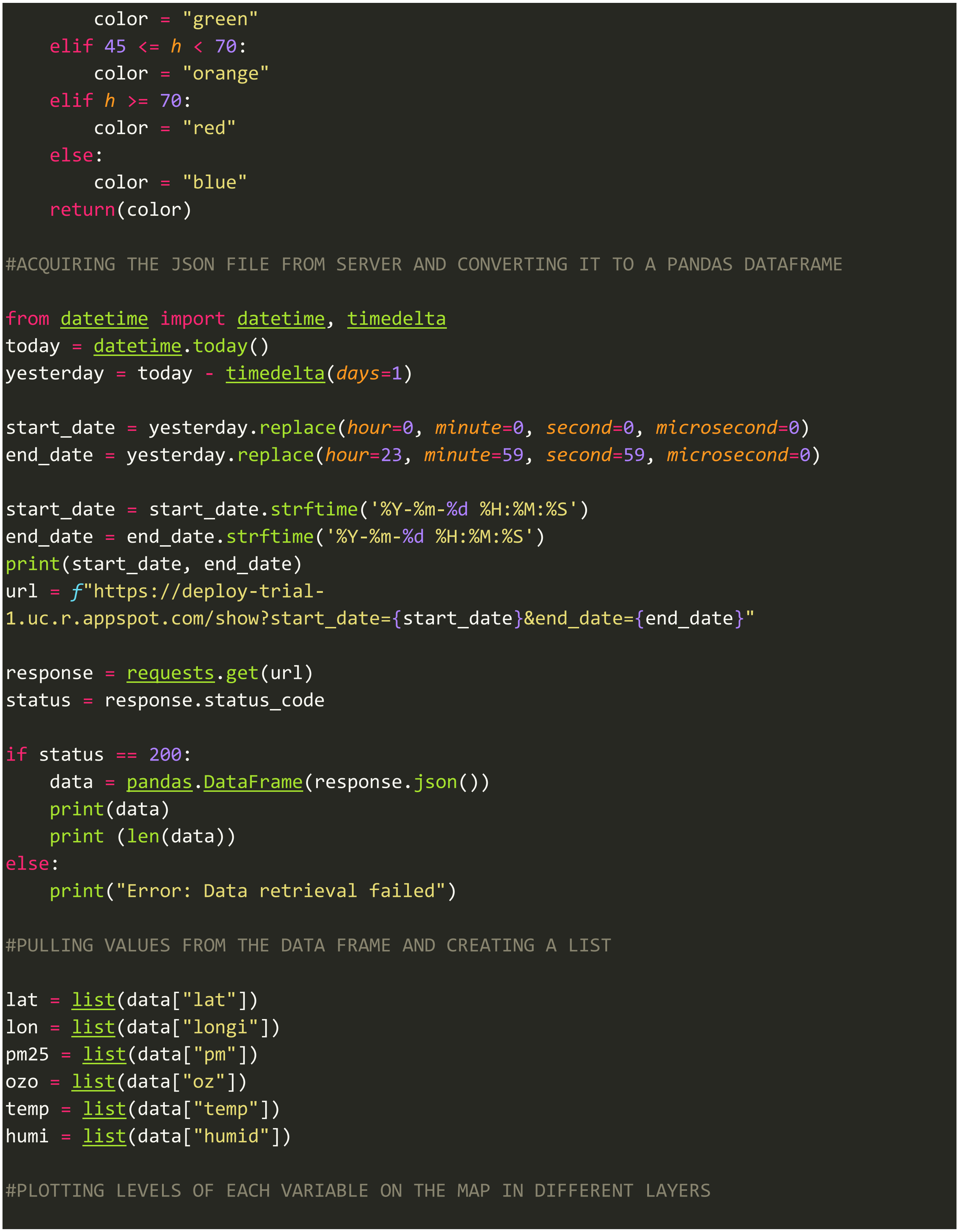
[5] J. Wesseling *et al.*, “Development and Implementation of a Platform for Public Information on Air Quality, Sensor Measurements, and Citizen Science,” *Atmos. 2019,Vol. 10, Page 445*, vol. 10, no. 8, p. 445, Aug. 2019, doi: 10.3390/ATMOS10080445.

[6] K. N. Genikomsakis, N. F. Galatoulas, P. I. Dallas, L. M. C. Ibarra, D. Margaritis, and C.S. Ioakimidis, “Development and On-Field Testing of Low-Cost Portable System forMonitoring PM2.5 Concentrations,” *Sensors (Basel).*, vol. 18, no. 4, Apr. 2018, doi: 10.3390/S18041056.

[7] United Nations Sustainable Development Goals - Website

# APPENDIX - I : Visualization Code

```
#IMPORT NECESSARY LIBRARIES

import requests
import pandas
import folium

#FUNCTIONS TO GENERATE COLORS BASED ON LEVELS: PM2.5, OZONE, TEMPERATURE, HUMIDITY

def gen_color_pm25(p):
    if 1 < p < 4:
        color = "green"
    elif 4 <= p < 7:
        color = "orange"
    elif p >= 7:
        color = "red"
    else:
        color = "blue"
    return(color)

def gen_color_ozon(o):
    if 0 < o < 20:
        color = "green"
    elif 20 <= o < 40:
        color = "orange"
    elif o >= 40:
        color = "red"
    else:
        color = "blue"

    return(color)

def gen_color_temp(t):
    if -30 < t < -5:
        color = "green"
    elif -5 <= t < 25:
        color = "orange"
    elif t >= 25:
        color = "red"
    else:
        color = "blue"
    return(color)

def gen_color_humi(h):
    if 10 < h < 45:
```



```
        color = "green"
    elif 45 <= h < 70:
        color = "orange"
    elif h >= 70:
        color = "red"
    else:
        color = "blue"
    return(color)

#ACQUIRING THE JSON FILE FROM SERVER AND CONVERTING IT TO A PANDAS DATAFRAME

from datetime import datetime, timedelta
today = datetime.today()
yesterday = today - timedelta(days=1)

start_date = yesterday.replace(hour=0, minute=0, second=0, microsecond=0)
end_date = yesterday.replace(hour=23, minute=59, second=59, microsecond=0)

start_date = start_date.strftime('%Y-%m-%d %H:%M:%S')
end_date = end_date.strftime('%Y-%m-%d %H:%M:%S')
print(start_date, end_date)
url = f"https://deploy-trial-
1.uc.r.appspot.com/show?start_date={start_date}&end_date={end_date}"

response = requests.get(url)
status = response.status_code

if status == 200:
    data = pandas.DataFrame(response.json())
    print(data)
    print (len(data))
else:
    print("Error: Data retrieval failed")

#PULLING VALUES FROM THE DATA FRAME AND CREATING A LIST

lat = list(data["lat"])
lon = list(data["longi"])
pm25 = list(data["pm"])
ozo = list(data["oz"])
temp = list(data["temp"])
humi = list(data["humid"])

#PLOTTING LEVELS OF EACH VARIABLE ON THE MAP IN DIFFERENT LAYERS
```

```
map_poll_levels = folium.Map(location = [data.iloc[0]['lat'], data.iloc[0]['longi']],
zoom_start= 15, tiles="CartoDB positron", attr="BigMapTiles")

fgp = folium.FeatureGroup(name= "PM2.5")
html_p = "PM2.5 LEVEL: %s"
for lt, ln, pm in zip(lat, lon, pm25):
     iframe = folium.IFrame(html = html_p %(pm), width= 200, height= 100)
     fgp.add_child(folium.CircleMarker(location= [lt, ln], radius = 6, fill_color
=gen_color_pm25(float(pm)), color="black", popup= folium.Popup(iframe),
fill_opacity=0.9))

fgo = folium.FeatureGroup(name= "OZONE")
html_o = "OZONE LEVEL: %s"
for lt, ln, oz in zip(lat, lon, ozo):
     iframe = folium.IFrame(html = html_o %(oz), width= 200, height= 100)
     fgo.add_child(folium.CircleMarker(location= [lt, ln], radius = 6, fill_color
=gen_color_ozon(float(oz)), color="black", popup= folium.Popup(iframe),
fill_opacity=0.9))

fgt = folium.FeatureGroup(name= "TEMPERATURE")
html_t = "TEMPERATURE LEVEL: %s"
for lt, ln, tp in zip(lat, lon, temp):
     iframe = folium.IFrame(html = html_t %(tp), width= 200, height= 100)
     fgt.add_child(folium.CircleMarker(location= [lt, ln], radius = 6, fill_color
=gen_color_temp(float(tp)), color="black", popup= folium.Popup(iframe),
fill_opacity=0.9))


fgh = folium.FeatureGroup(name= "HUMIDITY")
html_h = "HUMIDITY LEVEL: %s"
for lt, ln, hd in zip(lat, lon, humi):
     iframe = folium.IFrame(html = html_h %(hd), width= 200, height= 100)
     fgh.add_child(folium.CircleMarker(location= [lt, ln], radius = 6, fill_color
=gen_color_humi(float(hd)), color="black", popup= folium.Popup(iframe),
fill_opacity=0.9))


#LEGEND VIEW FOR ALL POLLUTANTS

legend_html_VALUES = """
<div style="position: fixed; top: 100px; left: 100px; width: 120px; height: 90px;
             border:2px solid grey; z-index:9999; font-size:14px;
             ">&nbsp; VALUES <br>
             &nbsp; Red: HIGH &nbsp; <i class="fa fa-circle fa-1x"
style="color:red"></i><br>
```

```
             &nbsp; Yellow: MED &nbsp; <i class="fa fa-circle fa-1x"
style="color:yellow"></i><br>
             &nbsp; Green: LOW &nbsp; <i class="fa fa-circle fa-1x"
style="color:green"></i><br>
</div>
"""
# add the legend for layer 1 to the map object
map_poll_levels.get_root().html.add_child(folium.Element(legend_html_VALUES))


#ADDING THE LAYER CONTROL TO DISPLAY ONE VARIABLE AT A TIME

map_poll_levels.add_child(fgp)
map_poll_levels.add_child(fgo)
map_poll_levels.add_child(fgt)
map_poll_levels.add_child(fgh)
map_poll_levels.add_child(folium.LayerControl())

map_poll_levels.save("Map-Pollution-Levels.html")
```