ECS782 COURSEWORK REPORT

180529469

AIRDUINO

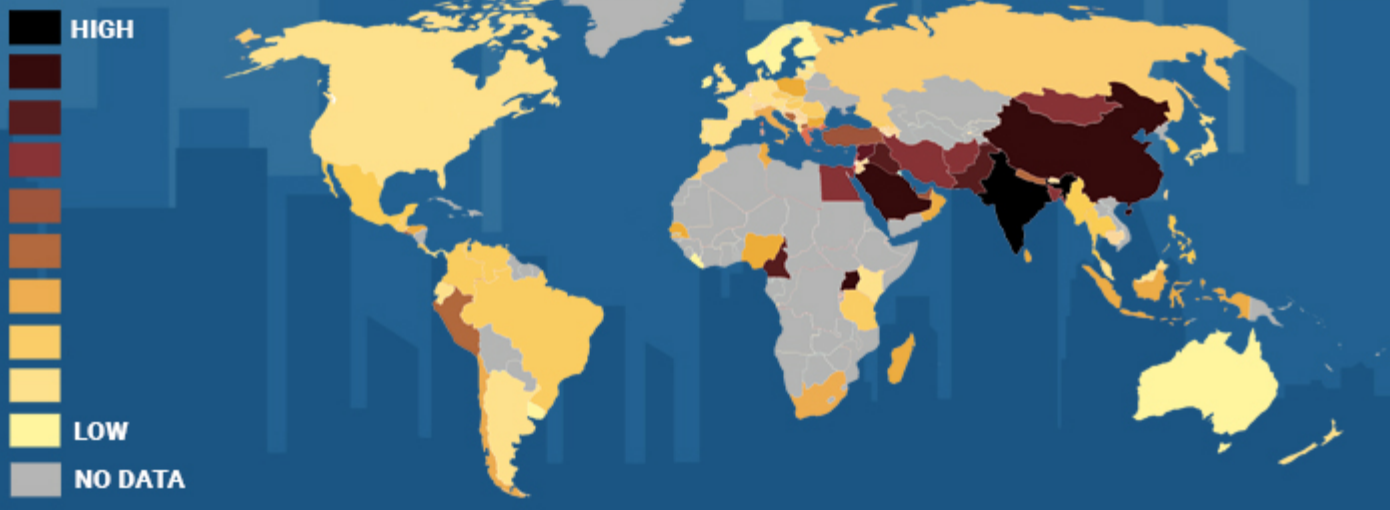
AIR QUALITY VISUALIZATION

Contents

1. INTRODUCTION

Air quality is a global challenge for governments, regulators, city administrators and citizens. Air pollution levels in many cities exceed legal and World Health Organization (WHO) limits for particulate matter and gaseous pollutants which can be found in concentrations that are hazardous to health. Poor air quality is causing a public health problem, since breathing polluted air increases the risk of debilitating and deadly diseases such as lung cancer, stroke, heart disease and chronic bronchitis.
Air pollution is now the world's fourth-leading fatal health risk, reported as causing one in ten deaths in 2013.

Many governments are investing multi-billion dollar sums in policies and solutions to improve air quality and they are empowering cities to tackle air pollution locally.
In order to implement effective policies and interventions there is an increasing focus on understanding the levels and causes of air pollution.



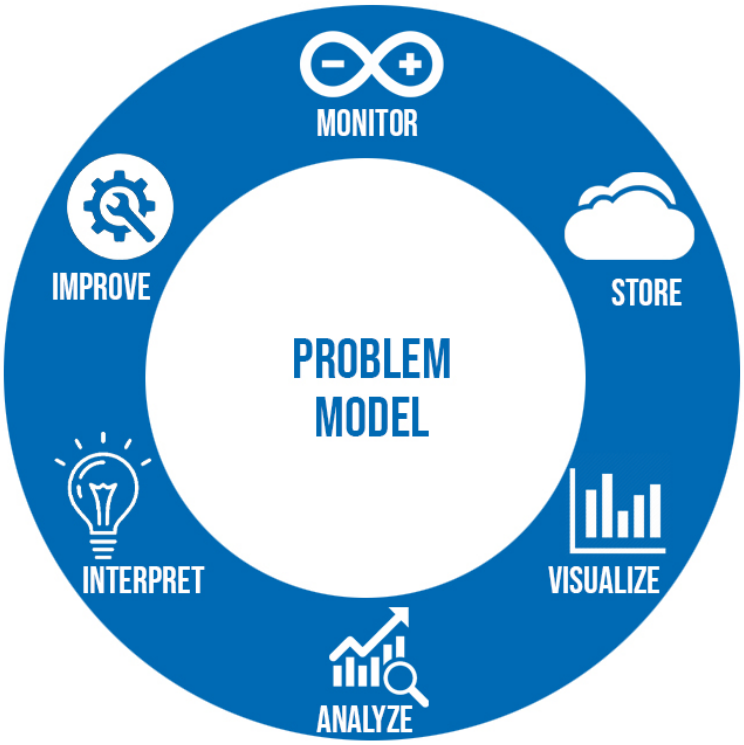
GLOBAL AIR POLLUTION LEVELS

2. System Specifications

2.1 PROBLEM STATEMENT

Advances in sensors, IoT platforms and mobile communication technologies have enabled the development of smaller, portable, low cost, mobile-enabled solutions that can measure and report air quality in near real time.
Big Data capabilities, such as analytics and machine learning, can then be applied to this data and related data sets, such as weather and traffic, to understand the causes and fluctuations in air pollution.

The aim of this project is to develop an IoSD that is capable of monitoring and collecting weather and air quality data, storing it in an accessible location such as the cloud in order to visualize, analyze and interpret trends in the data. Such a low cost, easy-to-deploy IoSD solution will provide enhanced visibility, situational awareness and early indications of pollution hotspots by giving insight into the levels and causes of air pollution.



2.2 System Requirements

- **MONITORING**

  Input : The system must be able to sense information regarding its environment. Sensors are needed to measure air quality parameters such as temperature, humidity and concentration of pollutants in the sensing environment.

  Controller : A low-cost and accessible microcontroller capable of interfacing with the sensors and implementing communication protocols is required.

  Communication : Serial UART to interface with the sensors, wireless capability for network access (Wi-Fi) and data communication protocol such as HTTP/MQTT.

  Storage : Storage requirements are minimal in the monitoring phase because the data is to be sent to the cloud. The device only needs flash memory to store code.

  Power : Portable power supply is needed as the system has to be easily moved between different environments.

  Software : The system should be programmable according to the above requirements, mainly being able to interface with the sensors and implementing the communication protocols.

- **STORAGE**

  The system must be able to seamlessly store the sensor data at an ideal sampling Interval to a networked storage solution such as the cloud. The storage service has to support the data communication protocol used by the monitoring device and should be able to organize the data such that it could be easily visualized.

- **VISUALIZATION**

  The collected data has to be visualized in a practical manner that is helpful in analyzing trends and patterns. The visualization tool must be able to present the data in a graphical way such that comparisons can be made between different datasets collected from different sensing environments.

3. Design & Architecture

3.1 Hardware Design

CONTROLLER

Arduino Uno R3

The Arduino was chosen because it is a low-cost device and easily programmable. It is effortlessly compatible with the rest of the design choices made for the IoSD. There were readily available libraries for the sensors, WiFi module and the Cloud platform.

SENSORS

a. Grove - Temperature & Humidity Sensor

This Temperature & Humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It gives the temperature values in degree Celsius and the Relative Humidity percentage.

b. Grove - Air Quality Sensor v1.3

This sensor is designed for comprehensive air quality monitoring. It is responsive to a wide scope of harmful gases, such as carbon monoxide, alcohol, acetone, thinner, formaldehyde and so on. Due to the measuring mechanism, this sensor does output specific data to describe target gas concentrations quantitatively. It instead gives a qualitative value which is indicative of the pollutant levels. A qualitative sensing approach was chosen to simplify the metrics for data visualization.

c. seeed Base Shield V2

The purpose of using the Base Shield was to help get rid of the breadboard and jumper wires. With the rich grove connectors on the base board, both the grove sensor modules could be connected to the Arduino Uno conveniently. This aids in the portability factor, making sure that the wires do not get disconnected.

COMMUNICATION

Arduino WiFi Shield

The Arduino WiFi Shield allows an Arduino board to connect to the internet using the 802.11 wireless specification (WiFi). It is based on the HDG204 Wireless LAN 802.11b/g System in-Package. An AT32UC3 provides a network (IP) stack capable of both TCP and UDP.

Android WiFi Hotspot

The mobile hotspot enables an Android phone to share its cellular data connection by creating a Wi-Fi network. The Arduino can then access that Wi-Fi network and use the phone's cellular data network to connect to the internet.

3.2 Software Design

The code for programming the Arduino is written in C. The file is attached with the report.

The program uses the AirQuality and DHT libraries in order to use the sensor modules. The WiFi library is also used to enable the use of the Arduino WiFi shield.

The program runs as follows:

- **Sensor initialization**
  The air quality sensor takes 20 seconds to initialize. The temperature sensor also takes a couple of seconds before it reads accurate values. The first block of code Initializes both the sensors.

- **Scanning and connecting to the WiFi hotspot**
  The Arduino searches for the Android mobile hotspot and connects automatically using the hardcoded wireless credentials defined in the program.

- **Obtaining the sensor values**
  The sensor values are read from the respective ports and stored in variables to transfer to the Thingspeak platform.

- **Sending data to Thingspeak**
  The channel number and API key from the Thingspeak account is hardcoded in the program. The Thingspeak.h library enables the use of the setField function which sends the sensor values to the Thingspeak platform over the HTTP protocol. MQTT was not used because the communication is one-to-one and HTTP did not impose any performance issues. The sensing and sending steps were looped with a delay of **20 seconds**.

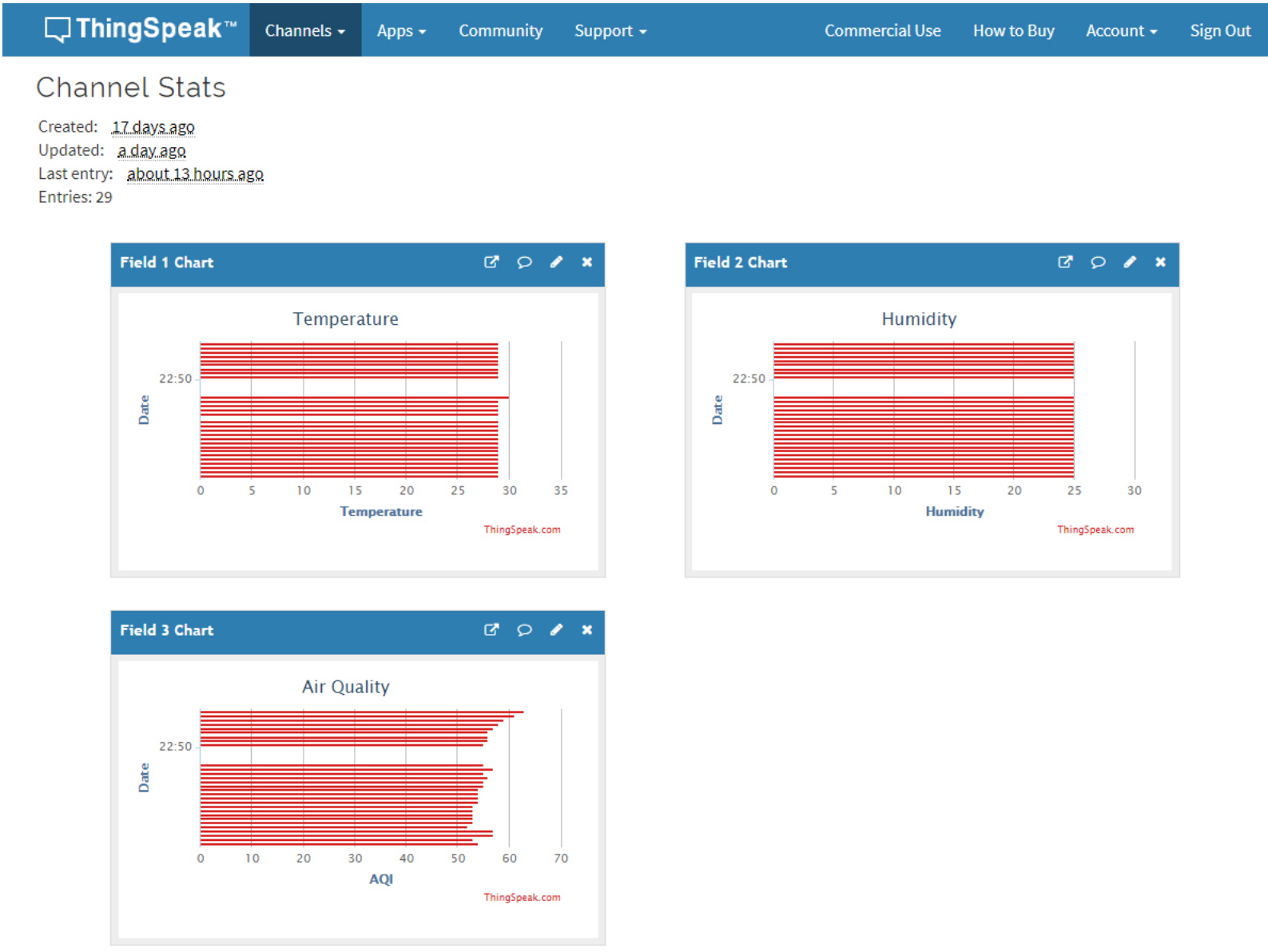
ThingSpeak IoT Platform

ThingSpeak is an open IoT platform with MATLAB analytics and support for Arduino. It was chosen as the Cloud platform to store the sensor data because of the relative ease in setting up a live data channel through readily available libraries and functions. The platform also has integrated MATLAB analytics which proved useful in visualizing the data.

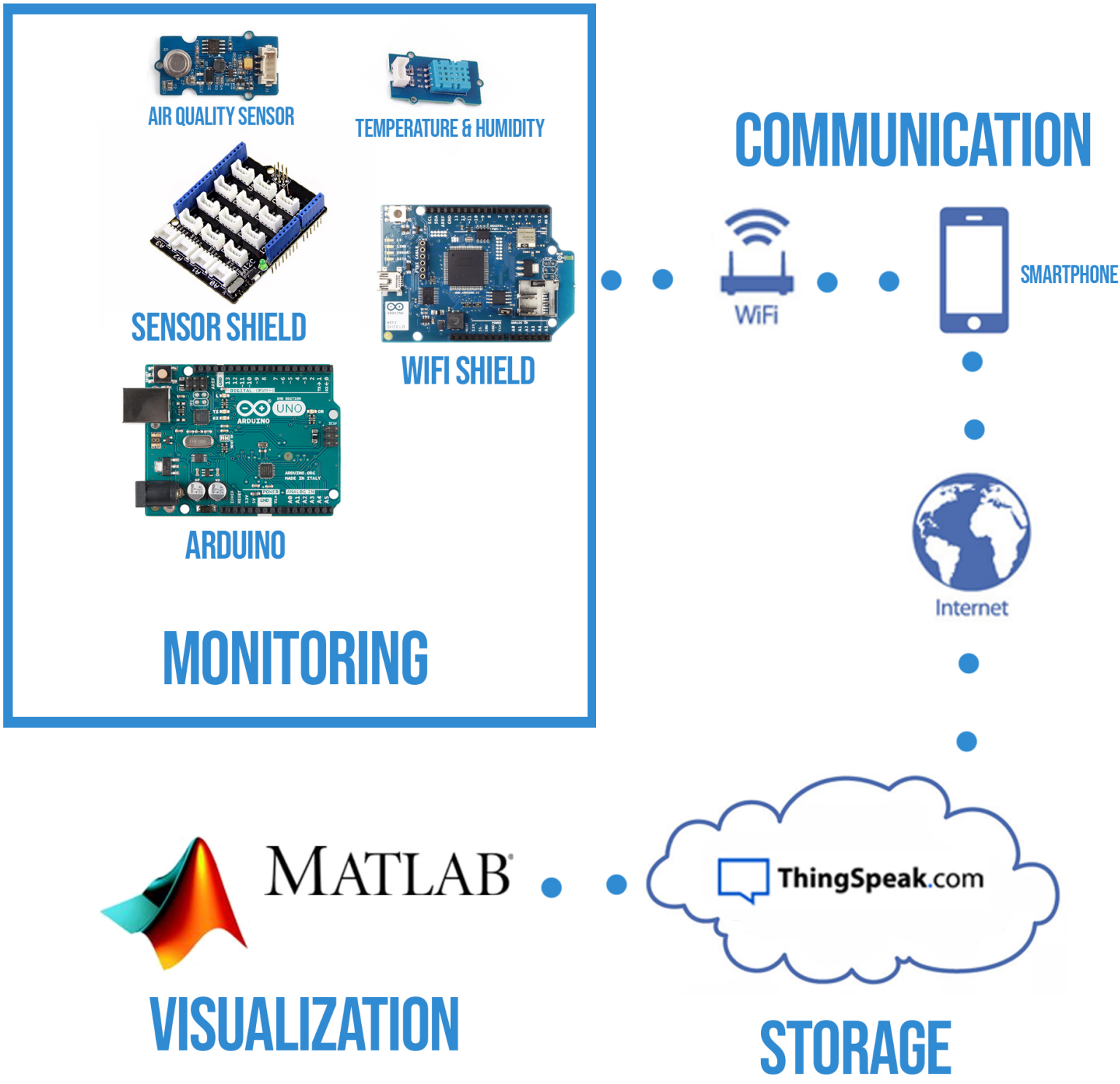
The steps undertaken to use the ThingSpeak IoT platform were as follows:

- Account creation using existing QMUL Mathworks ID
- Channel creation which generated a Channel ID and Write API Key
- Field creation for the three sensor values; Temperature, Humidity and Air Quality.

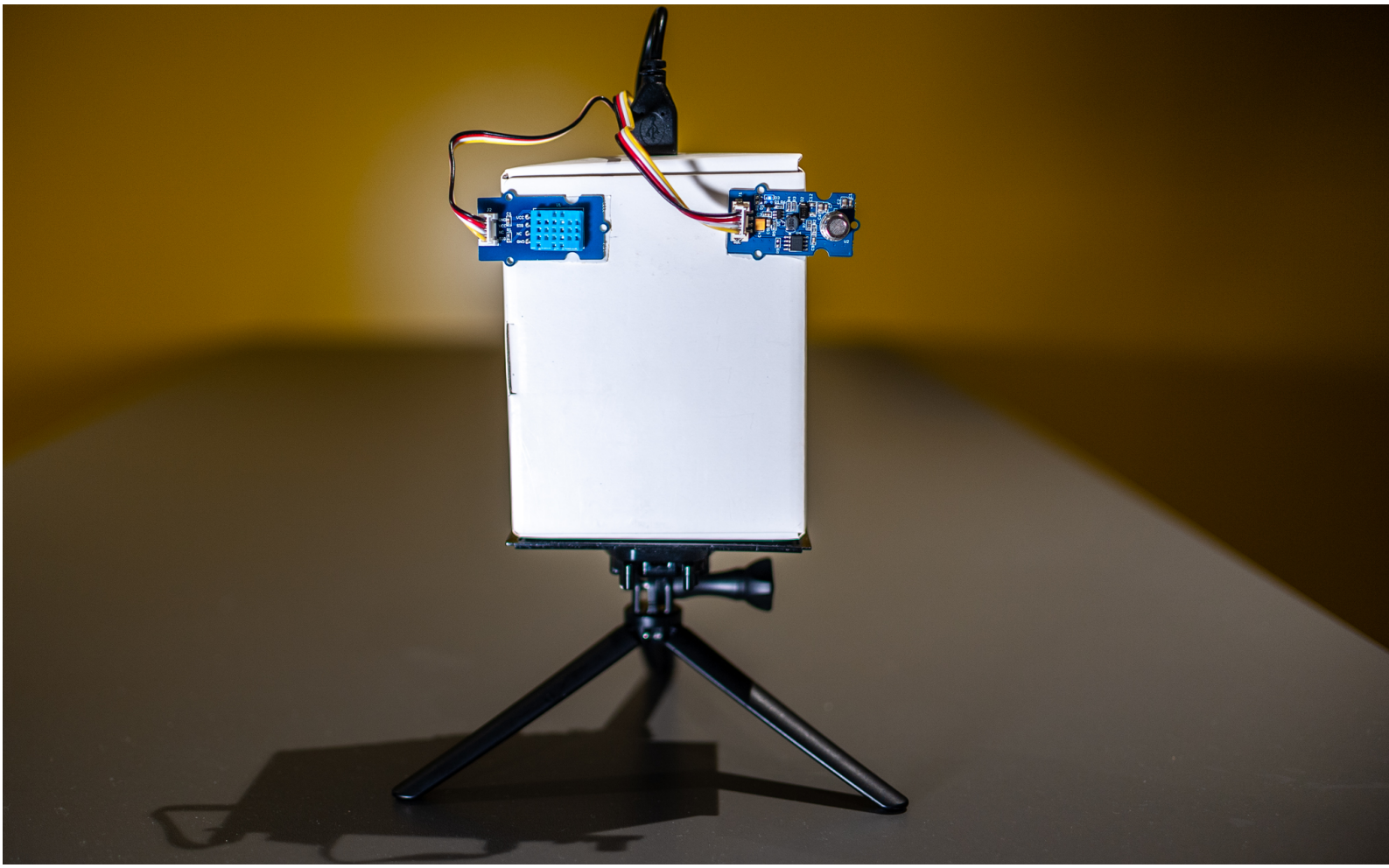
The firmware of the Arduino WiFi shield had to be updated to Version 1.1.0 in order to interface the Arduino with ThingSpeak. Line graphs of the collected data were readily available using the MATLAB Analytics integration.

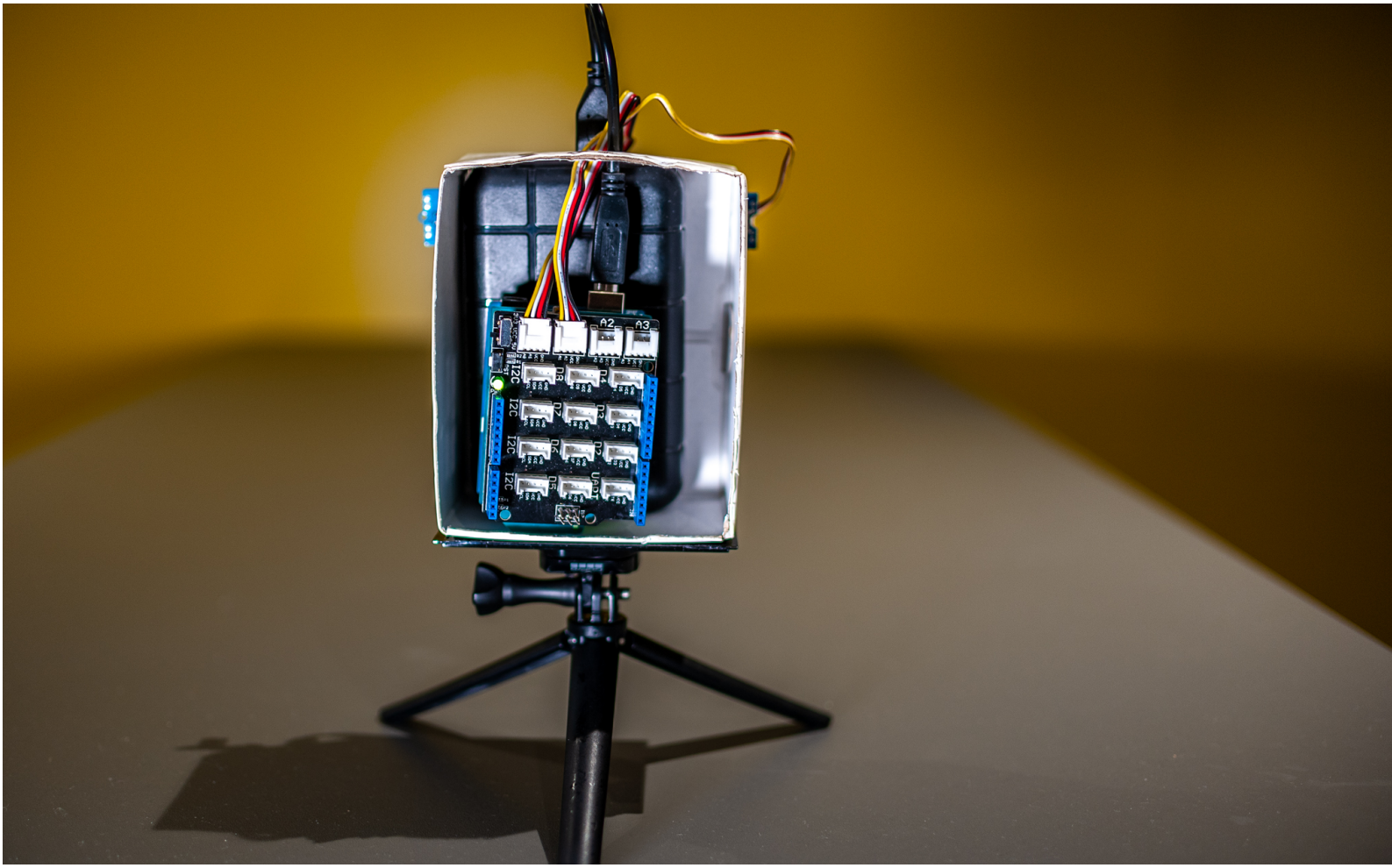


3.3 System Architecture



The system architecture follows the hardware and software design choices.
A key design choice in the architecture was the sampling time of **20 seconds**.
ThingSpeak does not allow a faster sampling rate, and the chosen rate was deemed optimum to generate enough data points in a limited time frame while not flooding the plots with redundant data.




A2
A3
I2C
D6
D7
D8
D9
D10
D11
D12
D13

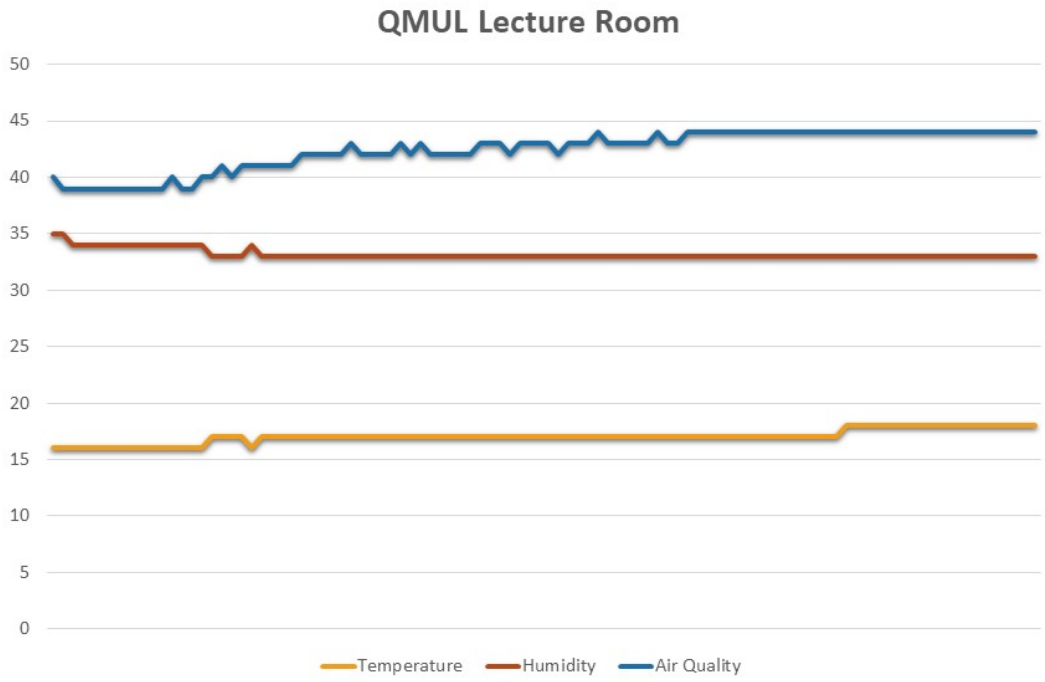
4. Implementation & Analysis

The data was collected in **four different environments** over a **30-minute time period**, which generated **100 data points**. The data was plotted using the integrated MATLAB functions in Thingspeak and the following observations were made.

- **Indoor Test - Apartment Bedroom**

- The test was carried out in an apartment bedroom which was a controlled indoor environment.
- The temperature and humidity values stayed relatively constant throughout the test period.
- The air quality sensor returned low values in the range of 40 - 45 which is an indicator of fresh air, implying very low concentrations of any pollutants.

- **Indoor Test - University Lecture Room**

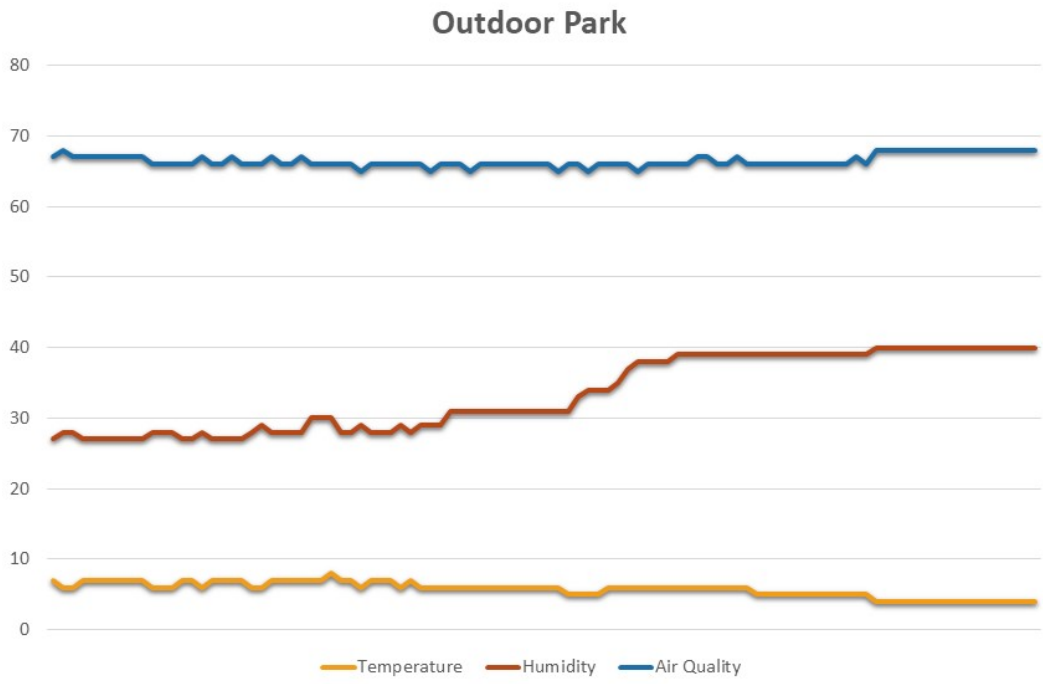


- The test was carried out in a QMUL Lecture Room and the trends in the data were Identical to the previous test, staying constant throughout the test period.

- Although, it was noted that the classroom was colder and more humid in comparison.

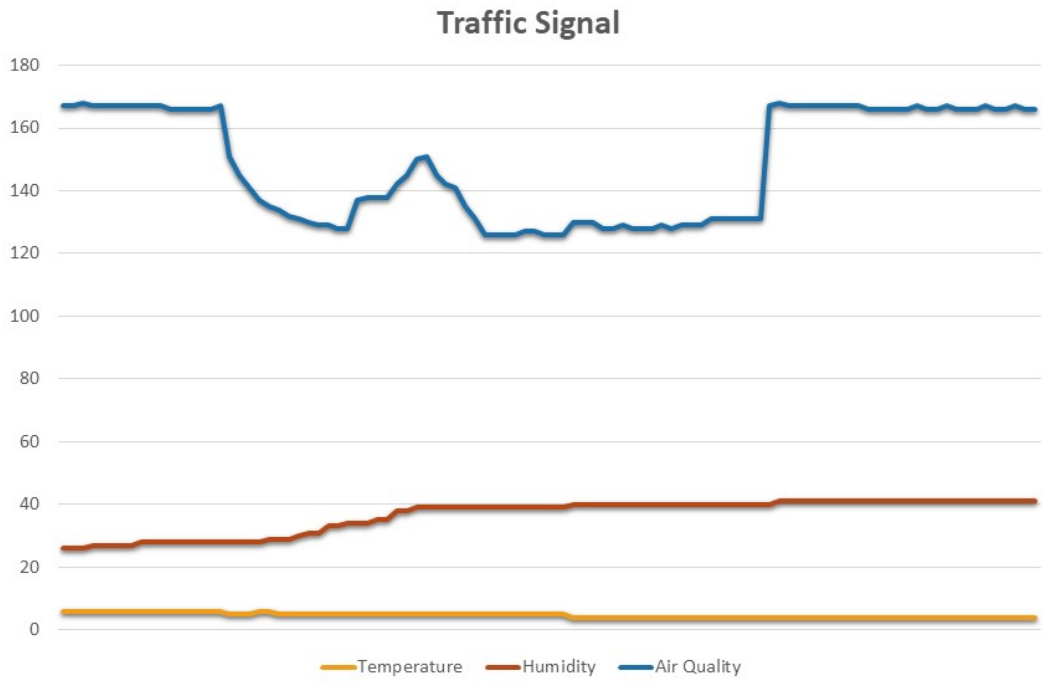
- The air quality values were similar and in the expected lower range indicating fresh air.

- **Outdoor Test - Mile End Park**



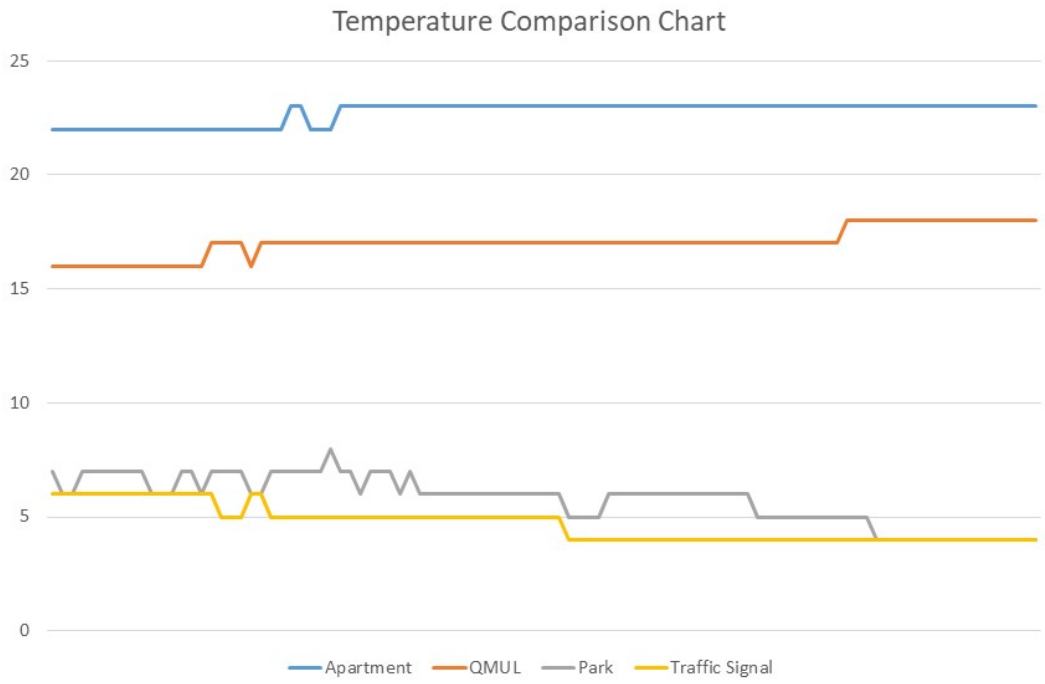
- The first outdoor test was carried out in an open park at Mile End.
- The temperature values were sensed accurately, dropping to almost 4 degrees towards the end of the test period.
- The humidity was found to be relatively higher than the indoor conditions, hitting a maximum of 40%.
- The air quality was observed to be worse compared to the indoor tests, with values in the range of 60 -70 indicating a slightly higher concentration of pollutants.

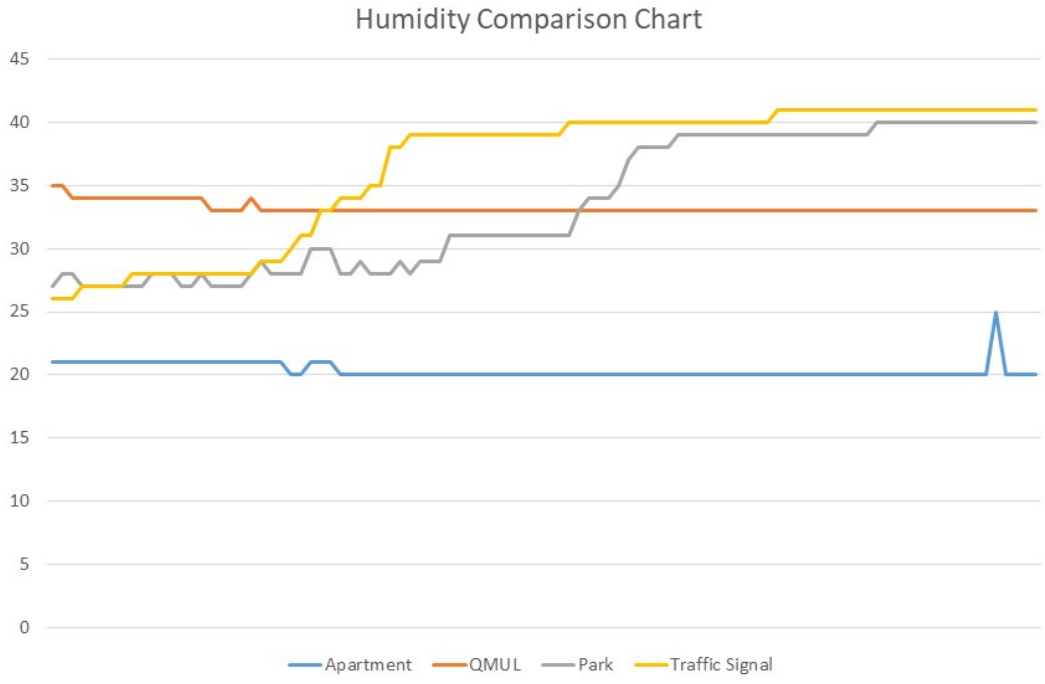
- **Outdoor Test - Bow Traffic Signal**

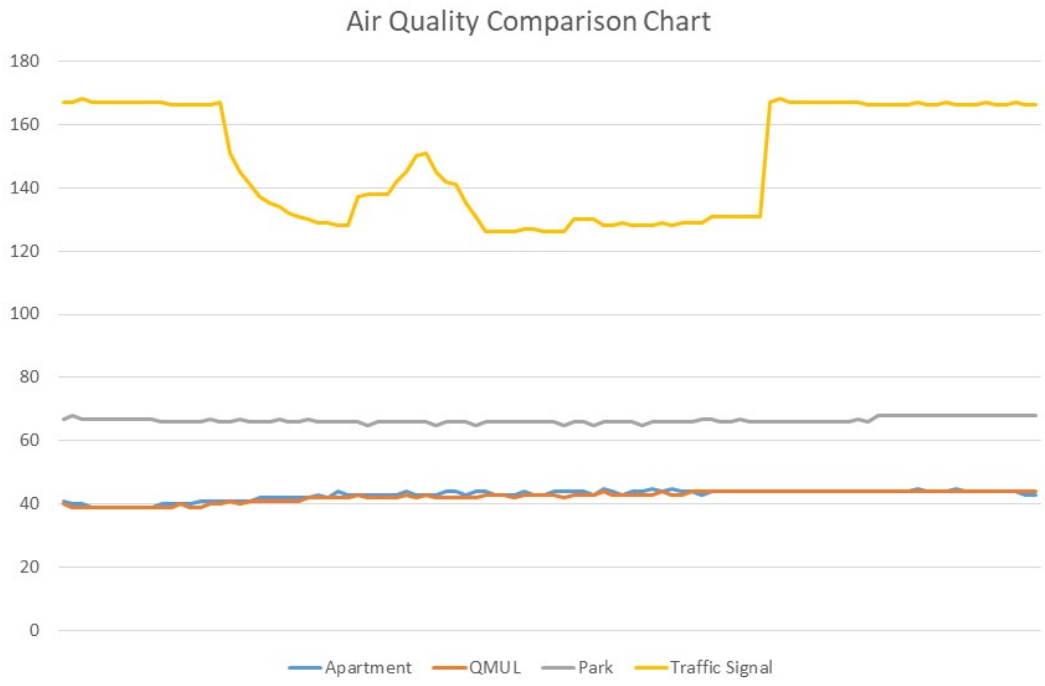


- Lastly, the airduino was tested at a busy traffic signal near the Bow Roundabout.
- The temperature dropped even further because the test was carried out in the evening.
- The humidity values were comparable to the results obtained from the previous test.
- The air quality sensor returned high values in the range of 120 - 180 which is an indicator of strong concentration of air pollutants in the sensing environment.
- It was further noted that there were spikes in the air pollution data, presumably indicating periods when vehicles were idling at the traffic signal during red lights.

Comparison Charts






Air Quality Comparison Chart
180
160
140
120
100
80
60
40
20
0
Apartment
QMUL
Park
Traffic Signal

5. Conclusions

The conclusions reached through the development and implementation of this project are as follows:

- **The device gives reliable results only when stationary.**

  The Airduino was also tested by mounting it on a bike. The sensors failed to give reliable readings because they are sensitive to vibrations. It is important to be able to operate the device while moving, because it helps determine pollution hotspots. A solution could be to use shock-resistant sensors.

- **The qualitative nature of the sensors give mixed results.**

  Although the results obtained were as expected, better sensors could be used to give exact quantitative data like the concentration of specific gases, or particulate matter.

- **The system is dependent on the mobile WiFi hotspot.**

  The device can be made truly autonomous by implementing direct cellular connectivity using a GSM module. There is also the security risk of hardcoding the WiFi credentials in the Arduino code.

- **A single device cannot give a true sense of the data.**

  The device can be implemented as a node in a large wireless sensor network installed in strategic locations which can be thought of as part of a Smart City to make relevant observations that can further be used to implement innovative and informed solutions to combat air pollution.

Demonstration Video

https://youtu.be/4teq_e8GdWg

6. References

1. https://openknowledge.worldbank.org/handle/10986/25013
2. https://www.gsma.com/iot/iot-big-data/smart-london-air-quality-monitoring-big-data/
3. http://wiki.seeedstudio.com/Grove-Air_Quality_Sensor_v1.3/
4. http://wiki.seeedstudio.com/Grove-TemperatureAndHumidity_Sensor/
5. https://uk.mathworks.com/help/thingspeak/use-arduino-client-to-publish-to-a-channel