



PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100 Feet Ring Road, Bengaluru, Karnataka, India – 560085

Project Report on USER CENTRIC POLLUTION ALERT SYSTEM

Submitted by

Akella Ramya (01FB16EEC025)

Bhagyashree B R (01FB16EEC068)

Akshaya H N (01FB16EEC352)

Jan – May 2020

under the guidance of

Prof. M Rajasekar

Associate Professor

Department of Electronics and Communication Engineering

PES University

Bengaluru – 560085

FACULTY OF ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PROGRAM B. TECH.



CERTIFICATE

This is to certify that the Report entitled

‘USER CENTRIC POLLUTION ALERT SYSTEM’

is a bonafide work carried out by

Akella Ramya (01FB16EEC025)

Bhagyashree B R (01FB16EEC068)

Akshaya H N (01FB16EEC352)

In partial fulfillment for the completion of 8th semester course work in the Program of Study B. Tech. in Electronics and Communication Engineering, under rules and regulations of PES University, Bengaluru during the period Jan – Apr 2020. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature with date & Seal

Prof. M Rajasekar

Internal Guide

Signature with date & Seal

Dr. Anuradha M

Chairperson

Signature with date & Seal

Dr.Keshavan BK

Dean of Faculty of Engineering

Name and signature of the examiners:

- 1.
- 2.



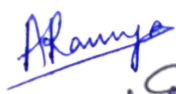
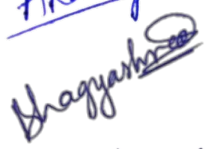
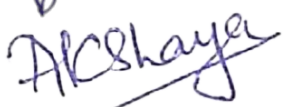
DECLARATION

We, **Akella Ramya, Bhagyashree B R, Akshaya H N**, hereby declare that the project report entitled, '*USER CENTRIC POLLUTION ALERT SYSTEM*', is an original work done by us under the guidance of **Prof. M Rajasekar**, Associate Professor, Dept. of ECE and is being submitted in partial fulfillment of the requirements for completion of 8th Semester course work in the Program of Study B. Tech. in Electronics and Communication Engineering.

PLACE: Bangalore

DATE: 29-07-2020

NAME AND SIGNATURES OF THE CANDIDATES

1.  (Akella Ramya)
2.  (Bhagyashree B R)
3.  (Akshaya H N)



ABSTRACT

In the past few decades, many urban areas in India have suffered from severe air pollution, making air quality an important area of research. Currently, pollution is increasing day-by-day due to various reasons such as industrial growth, development of automobile industries, and chemical industries. The aim of the project is to monitor air pollution and guide users through routes based on the pollution density of the area, along with providing alerts about highly polluted areas. This has been done by setting up a server and creating a mobile application. The pollution levels collected by a monitoring device are stored in the server along with the timestamp and location details. The mobile application uses the data from the server to give notifications to the user and alerts regarding high pollution levels. Also, the app has been integrated with an algorithm which calculates the least polluted route from the starting point to the destination as per the user's choice. The app keeps track of the user's transits and gives daily monthly and yearly graphs of the pollution intake. The pollution data consist of concentrations of various air pollutants such as Sulphur Dioxide(SO₂), Nitrogen Dioxide (NO₂), Carbon Monoxide (CO), particulate matter(PM 2.5 and PM 10), Carbon Dioxide (CO₂) and Ozone (O₃) on an hourly basis. Thus , with these features, any person can keep track of their pollution intake and take well informed decisions.



ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to Prof. M Rajasekar, our project guide for his valuable guidance, enthusiastic encouragement, and helpful critique in the development of this project. We would like to thank the Department of Electronics and Communication Engineering, PES University for giving us the opportunity to carry out this project as part of our final semester course requirements.

Special thanks to our Head of Department, Dr Anuradha M, for her unconditional support and guidance that were needed to complete this project. We would also like to extend our gratitude to Dr. Surya Prasad J, Vice Chancellor, PES University for his encouragement and support. Last but not least, we thank all our friends and family who have provided their valuable support and good wishes for the duration of the project.



Table of Contents

1. INTRODUCTION	6
1.1 What is Air pollution?	6
1.2 What we can do	7
2. PROBLEM STATEMENT	9
2.1 Air Quality Monitoring Systems	9
2.2 The Goal	9
3. LITERATURE SURVEY	11
4. METHODOLOGY	13
4.1 Overview	13
4.2 Requirements	15
4.2.1 Hardware	15
4.2.2 Software	19
i) Android Studio	19
ii) Google Maps API	19
iii) Routing API	23
4.3 Data Model	26
4.4 Server creation and setup	28
4.4.1 Setting up the server	28
4.4.2 Testing the server	29
4.5 Software and App	30
4.5.1 Notifications	30
4.5.2 Features	31
4.6 Least Polluted Route Algorithm	35



4.7 Data Visualization	38
5. RESULTS	39
6. CONCLUSION AND FUTURE SCOPE	42
6.1 Conclusion	42
6.2 Limitations	42
6.3 Applications	43
6.4 Future Scope	43
7. REFERENCES	44
APPENDIX	45



List of Figures

Fig 1.1	Pollutant emissions
Fig 2.1	Overview diagram
Fig 3.1	System Context Diagram
Fig 3.2	Absolute number of deaths from outdoor air pollution
Fig 4.1	Methodology flowchart
Fig 4.2	Pin Diagram of MQ2 Sensor
Fig 4.3	Circuit Diagram of MQ-135 Gas Sensor
Fig 4.4	Specifications of Sharp GP2Y1010AU0F Optical Dust sensor
Fig 4.5	Specifications of GSM SIM800C Module
Fig 4.6	Google Maps frame
Fig 4.7	Current location marker
Fig 4.8	Search bar
Fig 4.9	Red location markers
Fig 4.10	Sample polyline representation
Fig 4.11	Routing API



Fig 4.12	Request url to routing API
Fig 4.13	Routing API response in Json format
Fig 4.14	Raw Json response from routing API
Fig 4.15	Raw data scraped from Air quality API
Fig 4.16	Restructuring of data based on cities
Fig 4.17	Average Pollutant concentration
Fig 4.18	Google Cloud Server setup platform
Fig 4.19	Server created and ready to run
Fig 4.20	Notification showing the pollution level for different users at different location
Fig 4.21	Pollution Threshold Notification
Fig 4.22	App Home page with option buttons
Fig 4.23	a)Screen with one marker showing the pollution level in that place b)Screen with markers representing different places
Fig 4.24	Search bar screen to enter the starting point and destination
Fig 4.25	A map showing the best route to take based on the algorithm's calculations
Fig 4.26	The different routes in latitude and longitude format



Fig 4.27	Pollution levels in different routes
Fig 4.28	Data visualization based on annual data.
Fig 4.29	Data visualization based on hourly basis
Fig 5.1	Different routes possible from source to destination
Fig 5.2	Best route taken by algorithm
Fig 5.3	Colour Representations of AQI levels



1. INTRODUCTION

1.1 What is Air pollution?

Air pollution occurs when harmful or excessive quantities of substances are introduced into Earth's atmosphere. Vehicle discharges, synthetic substances from manufacturing plants, residue, dust and shape spores might be suspended as particles. It may cause diseases, allergies and even death to humans.

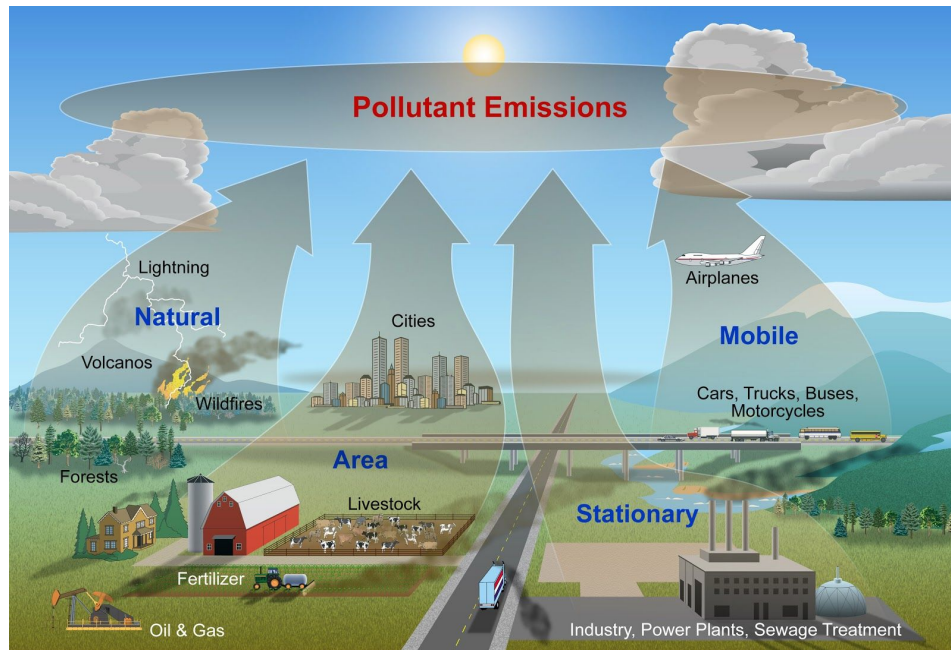


Fig 1.1: Pollutant emissions coming from different sources

In today's world air is contaminated primarily due to vehicular emissions, factory chemical emissions, cigarettes, dust, pollen, natural activities like volcanic eruptions to name a few. From all these emissions there are 7 major constituents namely:

1. **PM 2.5**: These are air particulate matter having a breadth of less than 2.5 micrometres causing constant infection such as asthma, bronchitis, heart and other respiratory problems.



2. **PM 10:** These are very small particles acting like gas and penetrate well into the lungs and bloodstream causing respiratory diseases and premature death.
3. **Nitrogen Dioxide:** Found in cities, comes from motor vehicle exhaust, other sources being petrol and metal refining, burning of fossil fuels, contributing to the formation of photochemical smog having a critical impact on human health.
4. **Ammonia:** Ammonia naturally exists in people and within the environment, utilised in industry, commerce and fundamental to many natural forms. Exposure to this gas in high concentrations causes burning of the nose , throat , respiratory tract, and olfactory weakness.
5. **Carbon monoxide,** an odourless and colourless gas is found in the fumes whenever fuel in cars, trucks or any other carbon material is burned. Any individual exposed to a high concentration of CO can make them unconscious or even killed. Again, infants and elderly individuals with chronic health problems are at high risk.
6. **Sulphur dioxide:** The main source of sulphur dioxide in the air is from industrial actions i.e. from the generation of power from coal, oil, or gas that contains sulphur . The impact of this gas on individuals is felt quite rapidly and its worst side effects such as coughing, wheezing, and tightness are felt around the chest within minutes.
7. **Ozone:** Ozone is a gas which can be "good" or "bad" depending on where it is. The bad ozone occurs at the ground level when pollutants from cars, factories react chemically with the sunlight. It is the main element in smog and can be worse in summer. It has its bout of health complications which can also lead to permanent lung damage.

1.2 What we can do

Many people are not aware of the harmful and long term effects of air pollution. Most of the constituents of air pollution are colourless and odourless gases apart from smoke which makes it seem undetectable. The places where awareness levels about pollution have risen are the places where air pollution has climbed to such suffocating levels where people are falling visibly sick. There are people living in very large groups in cities and villages which are also heavily polluted but unfortunately, they are not well informed of the pollution they are living in. It only reveals that more thought has to be put in while talking about the air quality and the terminology used.

Taking all of this into consideration we have built a system that measures and monitors air pollution at multiple points in the city at frequent intervals of time. Systematically storing, monitoring and analysing this data we have created visualizations to see the effects of pollution.



With these statistics and analysis, we would like to show its effect on a daily basis and how much it is present in the atmosphere where we live.

Having such a system will create awareness among people and help adjust their daily habits. This data in the long term, will allow us to find patterns and hopefully alert the authorities to devise government policies for the same.



2. PROBLEM STATEMENT

Air pollution in large urban areas has a drastic effect on humans and the environment. Ecological issues in India are growing quickly. The quality of air is inferior in metropolitan cities like Kolkata, Delhi, and Mumbai due to a large amount of carbon dioxide and other harmful gases emitted from vehicles and industries.

2.1 Air Quality Monitoring Systems

The primary purpose of a systematic air quality monitoring system is to distinguish between areas where pollution levels violate an ambient air quality standard and areas where they do not. As ambient air quality standards are set at levels of pollutant concentrations that result in adverse impacts on human health, evidence of levels exceeding an ambient air quality standard in an area requires a public air quality agency to reduce the corresponding pollutant. In other words, strategies, technologies, and regulations need to be developed to achieve reduction in pollution. The secondary purpose of a systematic monitoring system is to document the success of this endeavour, either to record the rate of progress towards attaining the air quality standard or to show that the standard has been achieved.

2.2 The Goal

We propose an air pollution monitoring system that predicts the pollution level of different routes to a given destination. It alerts the user if the pollution level is too high and suggests the least polluted route to the user.

“A system to monitor air pollution and guide users through routes based on the pollution density of the area, along with providing alerts about highly polluted areas”



Fig 2.1: Overview diagram of the project

Here the pollution levels are captured by wearable devices and stored in a server. This data is used to show different visualizations and suggest routes to users, through the medium of a user friendly application.

By deploying air quality monitoring devices along busy roads and intersections, cities can track pollution trends and hotspots (the routes and times of the day with the highest concentration of pollutants) to refine their policies accordingly. Industrial emissions are responsible for approximately 50% of all air pollution. The actions of industries have a huge impact on the health of employees and neighboring communities. Continuous monitoring helps pollution control boards as well as the industries to keep pollution levels in check on a real-time basis and to design appropriate emissions cutback initiatives. Air quality monitoring can bring in awareness about the effects of poor air quality and highlight the importance of afforestation.



3. LITERATURE SURVEY

As mentioned in the previous sections air contamination is one of the significant dangers of recent times. It has adverse impacts on humans, particularly on the respiratory system. There have been many research works made in the pollution monitoring systems (like hardware devices, data analysis and visualization) because of the above-mentioned reasons.

The authors in [1] have designed a pollution monitoring system to predict asthma attacks based on each person's pollution intake. For this they have designed a hardware system to constantly detect and monitor pollution levels in the user's vicinity and developed a machine learning model to predict when the asthma attacks can occur. The android device uses ATP (Asthma trigger predictor) and an incident recording module to update the current threshold values. Based on this data their machine learning model can predict if an asthmatic patient will face discomfort due to the pollution level in the area.

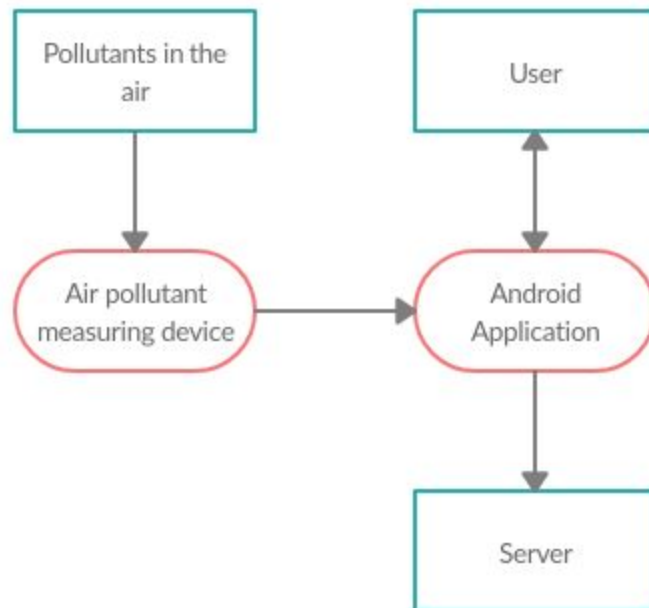


Fig 3.1: Asthma predictor System Context Diagram

Paper [2] shows a systematic air quality monitoring system to identify the areas where pollution levels violate an ambient air quality standard. The authors have prepared an IOT kit with Arduino and gas sensors to monitor the pollution levels. They have also developed an Android application termed IoT-Mobair so that users can access relevant air quality data from the cloud.



This system is analogous to Google Traffic or the Navigation application of Google Maps. Furthermore, air quality data can be used to predict future air quality index (AQI) levels.

The authors in Paper[3] have created a model to predict the pollution levels (Air Quality Forecasting using LSTM RNN and Wireless Sensor Networks). A hardware device was used to measure the pollution levels across different locations, along with data from a few websites. Using this data, the model was trained to predict the pollution levels, a few hours ahead of time. Although our project focuses on pollution alerts and not prediction of pollution levels, this paper gave insights into the pollution monitoring hardware.

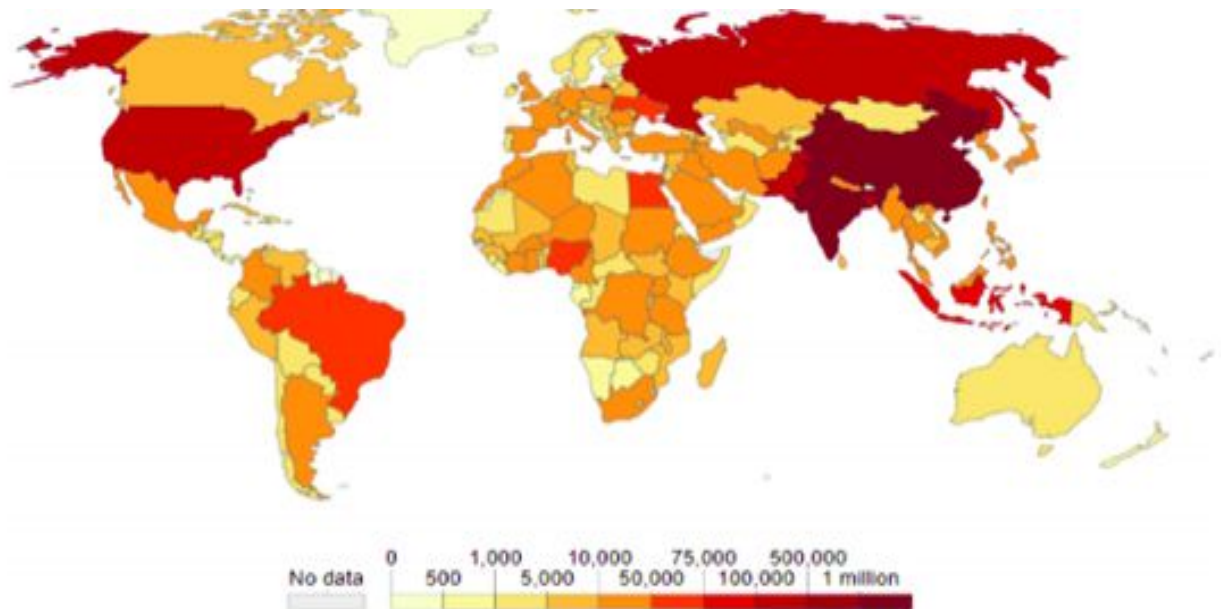


Fig 3.2:Absolute number of deaths from outdoor air pollution

The authors of [4] have used Wireless sensor networks and designed a system to track the user's activity across different places and the pollution intake at each point. This way the user is informed about the pollution level exposed to at each place, at every point of time. They have also created a heat map showing the absolute number of deaths from outdoor air pollution.

However, our proposed system shows the best and least polluted route from one location to another and suggests the route with least pollution, to the user. The system also warns about highly polluted routes and shows the daily and monthly intake of pollution. Also the hardware devices will be mobile and attached to the user to give a more precise reading of AQI, rather than the hardware device being placed in a remote location.



4. METHODOLOGY

4.1 Overview

An elevated level perspective on our strategy is depicted in this section. The overall flow of our methodology is demonstrated below.

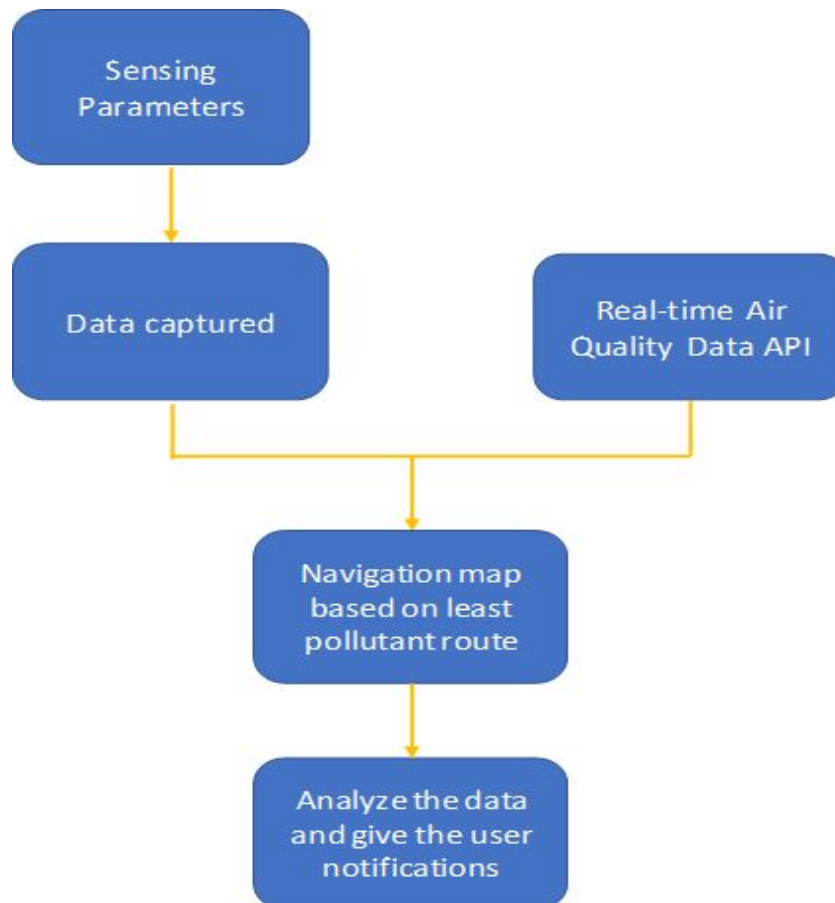


Fig 4.1: Project Flow diagram

The role played by each block is as follows:

- **Sensing Parameters:** It consists of a network of sensor modules. These sensors measure the pollutant levels. It captures the data and stores them.
- **Real Time air quality API:** The central and various state pollution control boards gather



real time data of various pollutant's concentration from various cities. This real time data is made available hourly on data.gov.in and accessible through a Data API.

1. Scraping: The data from the Real time air quality API is collected twice every hour and stored onto the server
 2. Cleaning: The data captured is then cleaned and reordered in the server based on different pollutants and areas.
-
- **Navigation based on least pollutant route:** The data captured is stored in the server based on AQI levels. When our app is used for navigation the route is shown based on AQI levels and the distance, giving the best possible route.
 - **Analyze the data and notify the user:** The data captured from the route is captured in the server. We use the same data and give the user a personal notification of the pollution levels, so that the user can take the necessary steps to reduce their pollution intake and keep themselves safe.



4.2 Requirements

4.2.1 Hardware

For the system to gather data a pollution monitoring device can be strapped on by the user. This wearable device can collect data and can further be stored in the server.

The following hardware components are required for the hardware device to measure the pollution. Their specifications are mentioned as follows.

1. **MQ-2 Gas Sensor:** The Gas Sensor (MQ2) is useful for the detection of gas leakage and is used at homes and industries. It is suitable for detecting Hydrogen (H₂), Liquefied Petroleum Gas (LPG), Methane (CH₄), Carbon Monoxide (CO), Alcohol, Smoke or Propane. It has a high sensitivity to the above mentioned gases and a quick response time, due to which measurements can be taken faster than the other alternatives. The sensor's sensitivity is variable and can be regulated by using a potentiometer.

1	Vcc	This pin powers the module, typically the operating voltage is +5V
2	Ground	Used to connect the module to system ground
3	Digital Out	You can also use this sensor to get digital output from this pin, by setting a threshold value using the potentiometer
4	Analog Out	This pin outputs 0-5V analog voltage based on the intensity of the gas

Fig 4.2: Pin Diagram of MQ2 Sensor

2. **MQ-135 Gas Sensor:** MQ-135 is a multi purpose air quality sensor. It is used for detecting and observing a wide amount of gases, including Ammonia (NH₃), Nitrogen oxides (NO_x), Carbon Dioxide (CO₂), Alcohol, Benzene and Smoke. It is used in offices and factories. MQ-135 gas sensor has a high sensitivity to Sulphide, Ammonia and Benzene steam.

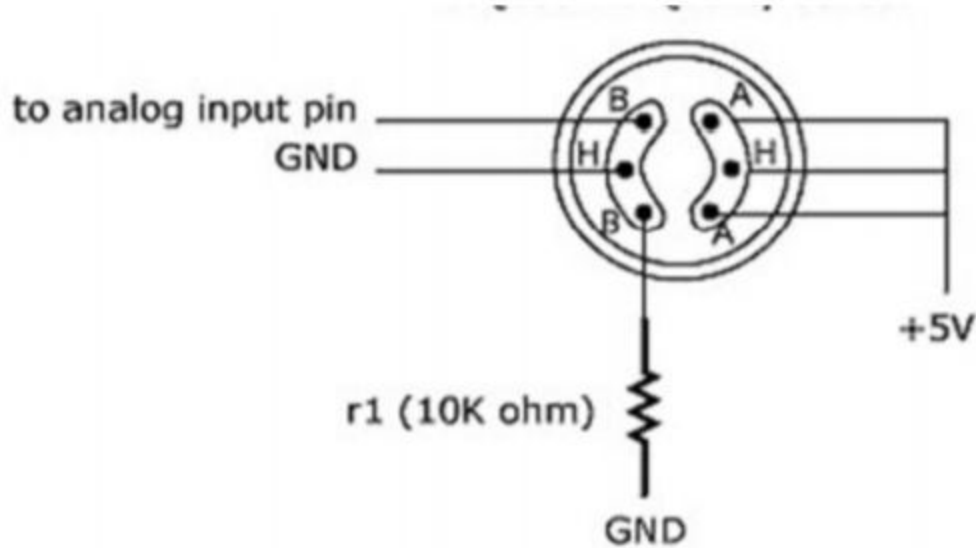


Fig 4.3: Circuit Diagram of MQ-135 Gas Sensor

3. PM (Particulate Matter) 2.5 Dust Sensor: Sharp GP2Y1010AU0F sensor is an optical sensor, which is used to sense dust particles in air. The principle of working of the sensor is as follows:

An infrared emitting diode and a photo transistor are arranged on opposite ends into this device, and this observes the reflected light of the dust particles in air. It is highly sensitive and can detect very small particles such as cigarette smoke. The sensor has a very low current consumption (20mAmax, 11mA), and can be powered up with voltages upto 7V DC. The sensor's output is an analog voltage which is in proportion to the dust density (which is being measured), and has a sensitivity of 0.5V/0.1mg/m³.



Sensitivity	0.5V/(100μg/m³)
Measurement range	500μg/m³
Power	2.2V-5.5V
Operating current	20 mA (max)
Operating temperature	-10°C~65°C
Storage temperature	-20°C~80°C
Lifetime	5 years
Dimension	63.2mm×41.3mm×21.1mm
Mounting holes size	2.0mm
Air hole size	9.0 mm

Fig 4.4: Specifications of Sharp GP2Y1010AU0F Optical Dust sensor

1. **4. GSM Module:** GSM is an acronym for “Global System for Mobile communications” The module that we have selected for our purpose is the SIM800C model. SIM800C is a small module which supports features such as sending and receiving SMS messages and voice calls. It also allows sending and receiving of data packets over the GPRS network. It is low cost, has small power requirements, and has quad band frequency support which makes this module an apt solution for any work that requires good connectivity and long range. After connection, the power module lights up, it looks for cellular networks and logs in. On board, the LED displays the state of the connection of the module which is useful for debugging purposes - (no network coverage is fast blinking, logged in is slow blinking).



	SIM800C
GSM	850,900,1800 and 1900MHz
BT	Support
FLASH	24Mbit
RAM	32Mbit

Fig 4.5: Specifications of GSM SIM800C Module

5. **Arduino Uno:** The Arduino Uno board is a widely used microcontroller board based on the ATmega328P. It is open source. It contains groups of analog and digital input and output pins that can be used to interface to various boards and circuits. It can be programmed with the Arduino integrated development environment (IDE) through an USB cable. It can be powered by the same USB cable, a 5V adapter or takes a supply of 7-12V through a battery or an external source.



4.2.2 Software

i) Android Studio

We have used Android studio, an IDE to build and develop the app. The frontend and different screens of the app have been programmed in XML. The backend, routing algorithm and other features of the app have been coded in Java. We have used the emulator in Android Studio to test the application. All the screens have been explained further in the upcoming chapters.

ii) Google Maps API

The google maps API facilitates the usage of the maps screen and a few other features of the maps. The features given by this API are free till a certain amount of API calls, beyond which they are paid. Also, there are many features of the API which are paid from the first API call, which includes the Directions API, Geocoding API etc.

Taking this into account we have not used the google directions API for routing, however we have used the other features of this API to plot the route that was received from the routing API , showing markers for different locations ,etc.

The usage requires an API key with a registered account. To use it in Android Studios we add the API key in the Gradle file. With the help of the API, we create a maps entity and use it as a variable. We can show the Map on the screen, or place pointers in different places and plot polylines on the map.

The features of the API that we used are:

1. **Maps view:** This can be viewed as a screen on the app, it gives the entire view of the world map along with the names of the places and clear view of the roads and other aspects. This includes the zoom in and zoom out features



Fig 4.6: Google Maps frame

2. **Current location details** (Fig 4.7): This can keep track of the user's current location details, with the help of GPS on the phone. This can be used after the user gives a few permissions to the app to track the location status. Based on this location from the maps, the app can give notifications to the user about the pollution levels in that area and precautions to be taken based on that

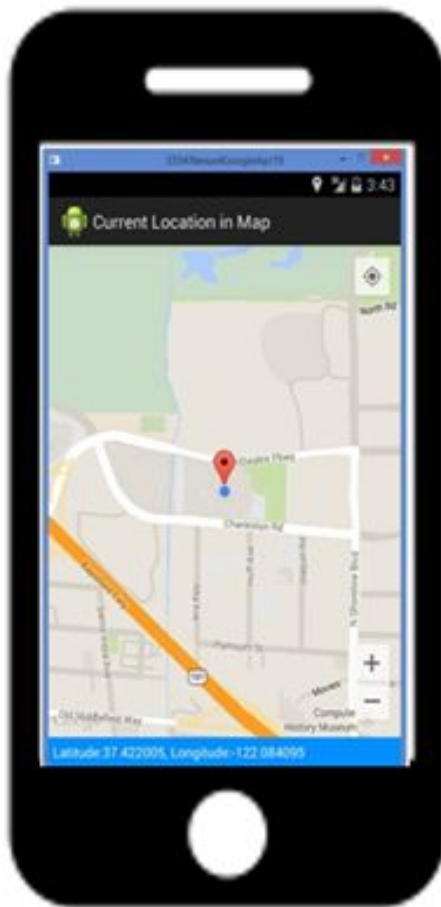


Fig 4.7: Current location marker

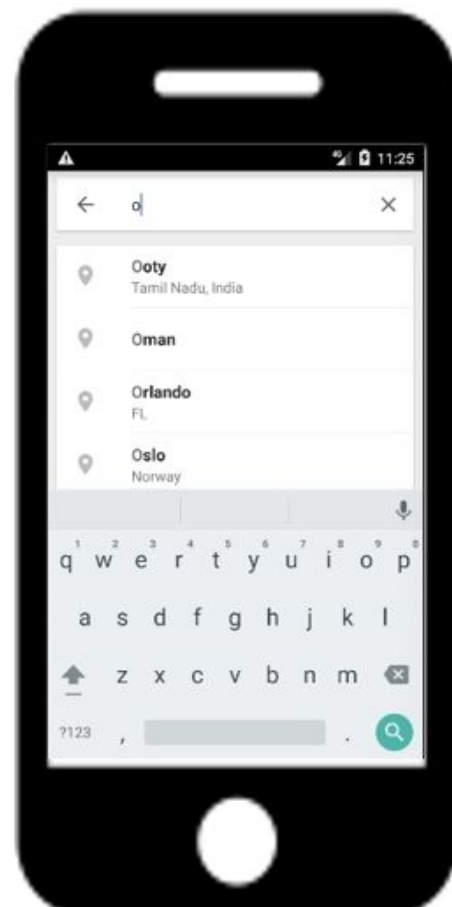


Fig 4.8:Google Search bar

3. **Google maps search bar** (Fig 4.8): The search bar is used to search for different locations on the map, it gives suggestions to the places that exist and keeps track of all the locations. After the user enters the location, it finds that place and converts them to latitude and longitude points. This search bar has been used for the user to enter the source and destination locations for the route generation.
4. **Markers** (Fig 4.9): These markers are used to identify a location on the map entity. Also, by clicking the marker the pollution level of that place is shown. The pollution information is taken from the server. These markers can be plotted on to the Map based on the latitude and longitude given.

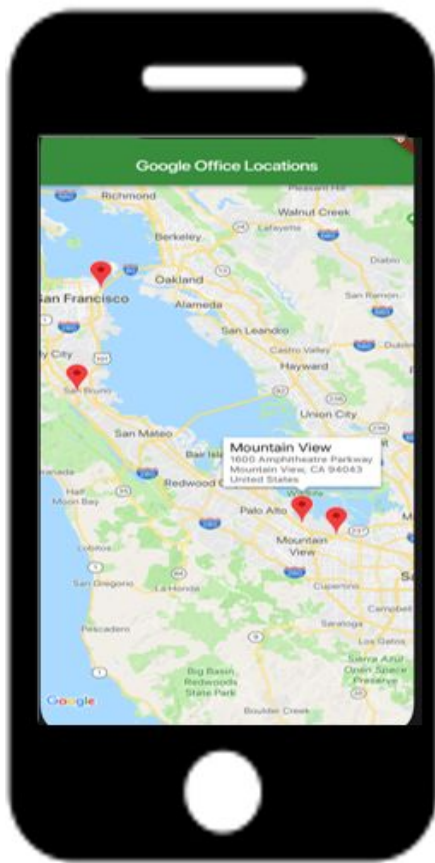


Fig 4.9: Red location markers



Fig 4.10: Polyline representing route

5. **Polylines to plot the directions** (Fig 4.10): A polyline is a line drawn between two or more given consecutive points, with an adjustable width (by default value 10), onto a map entity. Multiple of these polylines together represent a route between two places. These lines can be represented in any colour required. We used these polylines to plot the best route determined by our algorithm onto the maps.



iii) Routing API

For the Directions API, we have used TomTom, a location-based API i.e. Directions API to get different routes from a source to a destination.

Routing

GET /routing/{versionNumber}/calculateRoute/{locations}/{contentType} **Calculate Route**

Calculates a route between an origin and a destination.

Parameters

Name	Description
versionNumber * required integer (path)	Service version number. The current value is 1.
locations * required string (path)	Locations through which the calculated route must pass. Example: 52.50931,13.42936:52.50274,13.43872
contentType * required string (path)	The content type of the response structure. If the content type is jsonp, a callback method can be specified in the query parameters. Default value: xml

TRY IT OUT

Fig 4.11: Routing API

We use a URL link to call this API. This URL is standard, and we add the source and destination latitude and longitude points, along with an option to use only paved roads.

Request URL

```
https://api.tomtom.com/routing/1/calculateRoute/52.50931%2C13.42936%3A52.50274%2C13.43872/json?avoid=unpavedRoads&key=*****
```

Server response

Fig 4.12: Request URL to routing API

A Json request is made through the app to this URL generated. The API responds with a Json object consisting of details regarding the route like the distance between them, the time



taken for each route and the 4-5 different routes in the format of an array of latitude and longitude points. Based on the routes sent by this API the best route is calculated by the algorithm

JSON	Raw Data	Headers
Save	Copy	Collapse All Expand All Filter JSON
formatVersion: "0.0.12"		
▼ routes:		
▼ 0:		
▼ summary:		
lengthInMeters: 2141		
travelTimeInSeconds: 501		
trafficDelayInSeconds: 0		
departureTime: "2020-04-20T16:56:43+05:30"		
arrivalTime: "2020-04-20T17:05:04+05:30"		
▶ legs: [-]		
▶ sections: [-]		
▶ 1: {}		
▶ 2: {}		
▶ 3: {}		

Fig 4.13: Routing API response in Json format

Fig 4.14: Raw Json response from routing API



4.3 Data Model

An API generates the Air Quality data on “data.gov.in” twice an hour. A script makes requests to this API to scrape that data and store it in the server.

(To use this script, go to data.gov.in, sign up and generate an API key.

Once you have an API key replace the placeholder <API KEY> in the data URL with your API key and run)

Another script will run in parallel along with the current code to check whether the data captured twice in an hour is the same. If data is the same we delete the file. This process helps us reduce the data duplication.

The raw data scraped from the API is shown in the figure below.

	A	B	C	D	E	F	G	H	I	J	K	L
	id	country	state	city	station	last_updat	pollutant	pollutant	pollutant	pollutant	pollutant	unit
1	1	India	Andhra_Pr	Amaravati	Secretaria	#####	PM2.5	27	98	65	NA	
2	2	India	Andhra_Pr	Amaravati	Secretaria	#####	PM10	41	105	76	NA	
3	3	India	Andhra_Pr	Amaravati	Secretaria	#####	NO2	4	16	9	NA	
4	4	India	Andhra_Pr	Amaravati	Secretaria	#####	NH3	2	2	2	NA	
5	5	India	Andhra_Pr	Amaravati	Secretaria	#####	SO2	5	31	19	NA	
6	6	India	Andhra_Pr	Amaravati	Secretaria	#####	CO	18	36	24	NA	
7	7	India	Andhra_Pr	Amaravati	Secretaria	#####	OZONE	18	60	24	NA	
8	8	India	Andhra_Pr	Rajamaher	Anand Kal	#####	PM2.5	27	82	54	NA	
9	9	India	Andhra_Pr	Rajamaher	Anand Kal	#####	PM10	48	101	64	NA	
10	10	India	Andhra_Pr	Rajamaher	Anand Kal	#####	NO2	1	17	6	NA	
11	11	India	Andhra_Pr	Rajamaher	Anand Kal	#####	NH3	2	7	3	NA	
12	12	India	Andhra_Pr	Rajamaher	Anand Kal	#####	SO2	4	20	10	NA	
13	13	India	Andhra_Pr	Rajamaher	Anand Kal	#####	CO	17	50	27	NA	
14	14	India	Andhra_Pr	Rajamaher	Anand Kal	#####	OZONE	26	94	52	NA	
15	15	India	Andhra_Pr	Tirupati	Tirumala,	#####	PM2.5	40	70	53	NA	
16	16	India	Andhra_Pr	Tirupati	Tirumala,	#####	PM10	69	107	82	NA	
17	17	India	Andhra_Pr	Tirupati	Tirumala,	#####	NO2	17	107	53	NA	
18	18	India	Andhra_Pr	Tirupati	Tirumala,	#####	NH3	6	8	7	NA	
19	19	India	Andhra_Pr	Tirupati	Tirumala,	#####	SO2	4	26	11	NA	
20	20	India	Andhra_Pr	Tirupati	Tirumala,	#####	CO	14	27	18	NA	
21	21	India	Andhra_Pr	Tirupati	Tirumala,	#####	OZONE	72	162	88	NA	
22	22	India	Andhra_Pr	Vijayawad	PWD Grou	#####	PM2.5	27	98	53	NA	
23	23	India	Andhra_Pr	Vijayawad	PWD Grou	#####	PM10	38	98	71	NA	
24	24	India	Andhra_Pr	Vijayawad	PWD Grou	#####	NO2	10	17	13	NA	
25	25	India	Andhra_Pr	Vijayawad	PWD Grou	#####	NH3	1	2	1	NA	

Fig 4.15: Raw data scraped from Air quality API

Then we restructured the data based on states, cities and pollutants and their average values and stored them on to the server. Using this AQI level data, the pollution at different places were captured.



Name	Date modified	Type	Size
Bengaluru	20-01-2020 09:40	File folder	
Chikkaballapur	20-01-2020 09:40	File folder	
Hubballi	20-01-2020 09:40	File folder	
Kalaburagi	20-01-2020 09:40	File folder	
Kalaburgi	20-01-2020 09:40	File folder	

Fig 4.16: Restructuring of data based on cities

ate Backup Plus Drive (E:) > Dataset_Restructured > Karnataka > Bengaluru > Bapuji Nagar, Bengaluru - KSPCB >			
Name	Date modified	Type	Size
CO	15-02-2020 15:03	File folder	
NH3	20-01-2020 09:40	File folder	
NO2	20-01-2020 09:40	File folder	
OZONE	20-01-2020 09:40	File folder	
PM2.5	20-01-2020 09:40	File folder	
PM10	20-01-2020 09:40	File folder	
SO2	20-01-2020 09:40	File folder	

Fig 4.17: Average Pollutant concentration



4.4 Server creation and setup

4.4.1 Setting up the server

- An instance is a virtual machine (VM) hosted on Google's infrastructure and created using the Google Cloud Console.
- When an instance is created in a project, the zone, operating system, and machine type of that instance should be specified.
- When an instance is deleted, it is removed from the project.
- We Created a Linux virtual machine instance in Google Compute Engine using the google cloud console.

VM instances

[CREATE INSTANCE](#) [IMPORT VM](#) [REFRESH](#) [START](#) [STOP](#) [RESET](#) [SHOW I](#)

Filter VM instances Columns

Name	Zone	Recommendation	In use by	Internal IP	External IP	Connect
<input type="checkbox"/> pollutionserver	asia-south1-a			10.160.0.3 (nic0)	None	SSH

Related Actions

View Billing Report
 View and manage your Compute Engine billing

Monitor Stackdriver Logs
 View, search, analyse and download VM instance logs

Set up Firewall Rules
 Control traffic to and from a VM instance

Manage Quota
 View or request instance quota

Fig 4.18: Google Cloud Server setup platform

- The machine properties of the instances can be chosen from a list provided, such as the number of virtual CPUs and the amount of memory, by using a set of predefined machine types or by creating our own custom machine types.
- Connect to the instance using Secure Shell (SSH) to configure applications on the instances.



- A virtual machine instance on Google Compute Engine can be managed like any standard Linux server.
- Deployed a simple Apache web server on the virtual machine instance.

4.4.2 Testing the server

Once the VM instance is started, we navigate to the external IP through which the server can be accessed. Connection is established to the virtual machine instance via the ssh. Interaction with the virtual machine instance is done using the terminal window. The server is set up and it stores the values of pollution levels of various places. The server gets this data which is in the format of latitude, longitude and pollution level from the hardware, once the devices have been deployed across the city. It also serves a console webpage that displays the data being sent by a node in real time.

At the end of each hour, the server finds the average, minimum and maximum of all the pollutant concentrations received from a given node and saves it in a CSV file. It then makes these values available to the app through an API. The app calculates the least polluted route based on these values and suggests this route to the user and alerts are notified based on the pollution level accordingly.

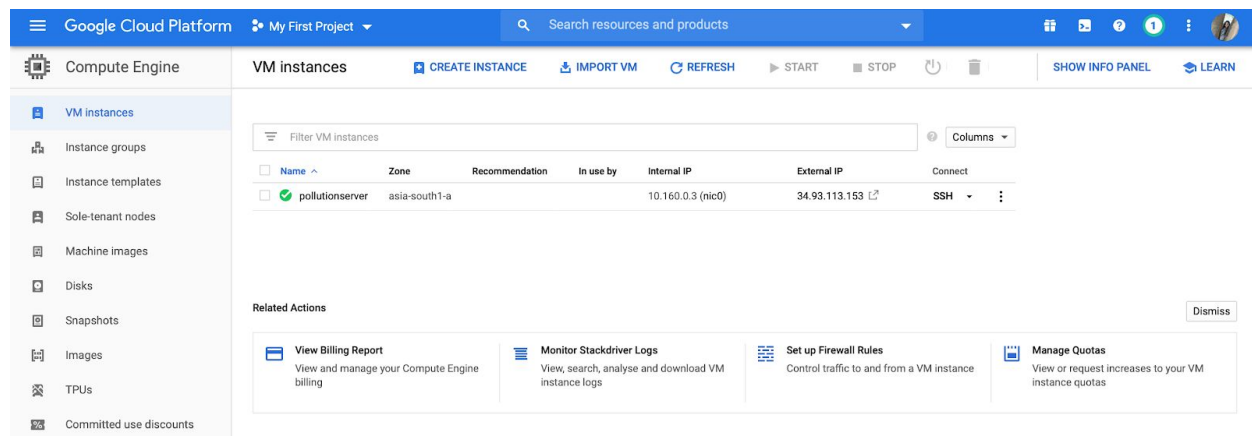


Fig 4.19: Server created and ready to run



4.5 Software and App

4.5.1 Notifications

Once the application is launched, a user login page appears. The user needs to enter their details and create an account. The login is essential so that each user's details are stored separately, and this data can be used to calculate pollution levels and thresholds specifically for that user.

The user receives a notification regarding the pollution details at that location at that point of time

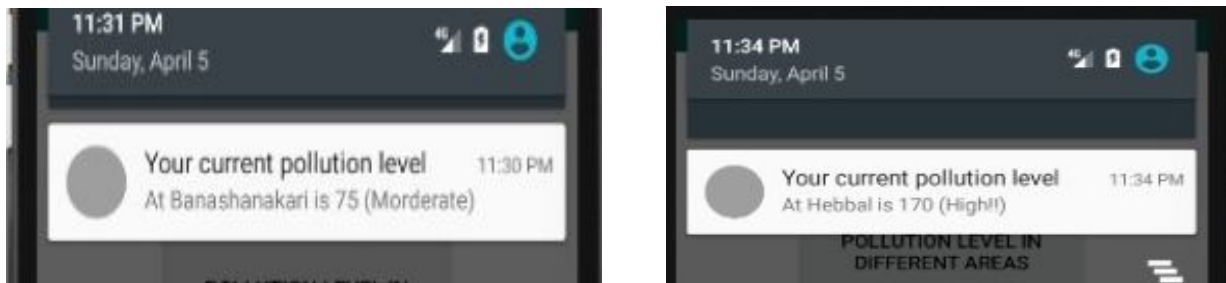


Fig 4.20 (a) (b): Notification showing the pollution level of different users in different locations

The current location of the user is tracked by the google maps API through the app and converted to latitude and longitude points. A request for these latitude and longitude points is made to the server and the corresponding pollution level is returned. This pollution level along with the current area they are in is made available to the user in the form of a notification.

The app keeps track of the pollution that the user encounters on a daily basis. If the exposure to pollution is high then, a notification is sent to inform the user that the threshold to pollution has been reached, and it is suggested that the user avoids polluted areas.

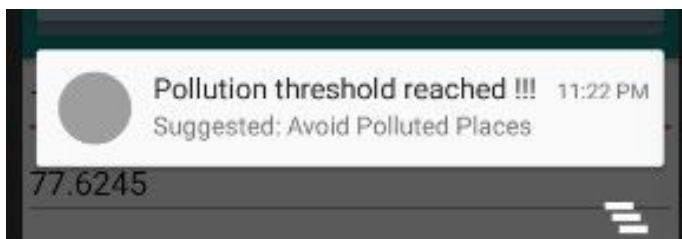


Fig 4.21 : Pollution Threshold Notification



4.5.2 Features

- 1) Once the registration or login is complete the following screen appears, showing the features that can be accessed by clicking on these buttons (Fig 4.22)

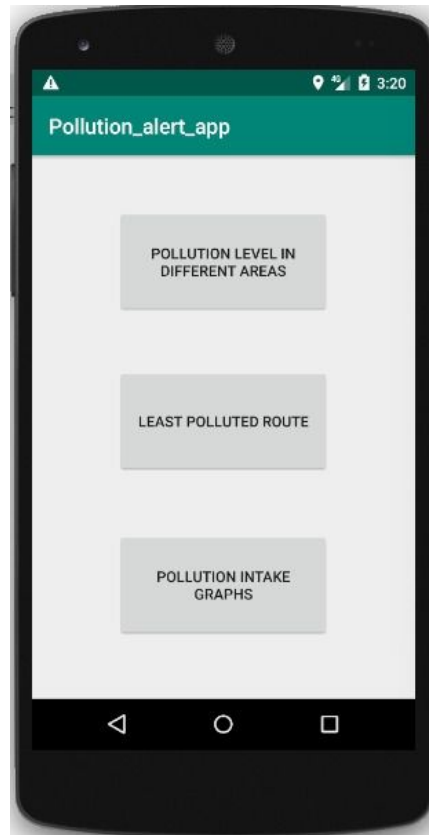


Fig 4.22: App Home page with option buttons

- 2) This screen(Fig 4.23) shows the user the pollution levels in different areas of Bangalore. There are different markers placed in different parts of Bangalore and shown in a Map representation. Clicking on each of these markers shows the pollution level at that location, along with whether that amount of pollution is tolerable or unhealthy for that user.

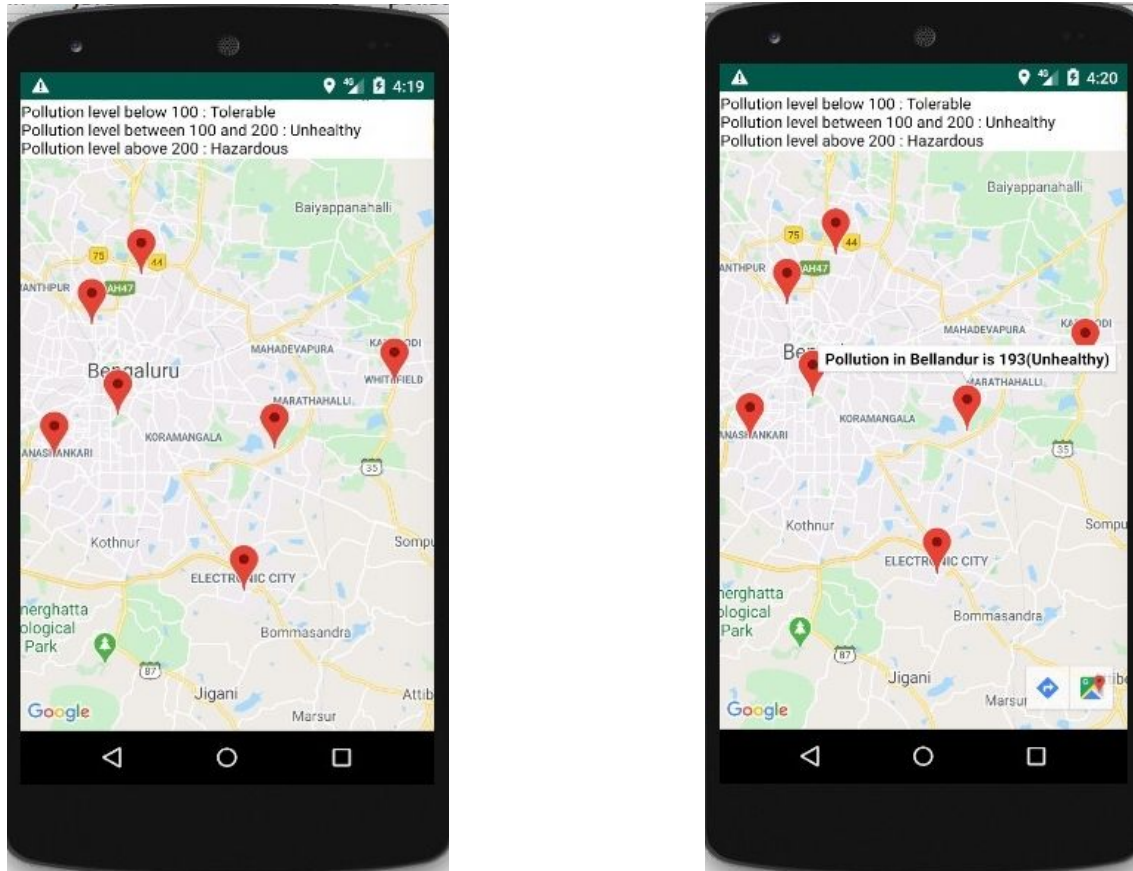


Fig 4.23 (a),(b):Screen with markers representing different places,
Screen showing the pollution in a location after clicking on one of the markers

- 3) The latitude and longitude of around 10 different locations are stored in the app. Requests are made to the server to get the pollution level of these latitude and longitude points. These values are stored. Using the latitude and longitude points the markers are placed in those locations on the map along with the pollution data and the location data. Also, along with the pollution level a warning is given based on the intensity of pollution.
 - Level below 100: Tolerable
 - Levels between 100 and 200: Unhealthy
 - Levels above 200: Hazardous
- 4) This feature shows routes which are less polluted as compared to the other routes from a given location to the destination. This screen shows the fields to be entered i.e. the source



and the destination fields. After entering the values, the user clicks on the GET DIRECTIONS button to get the route.

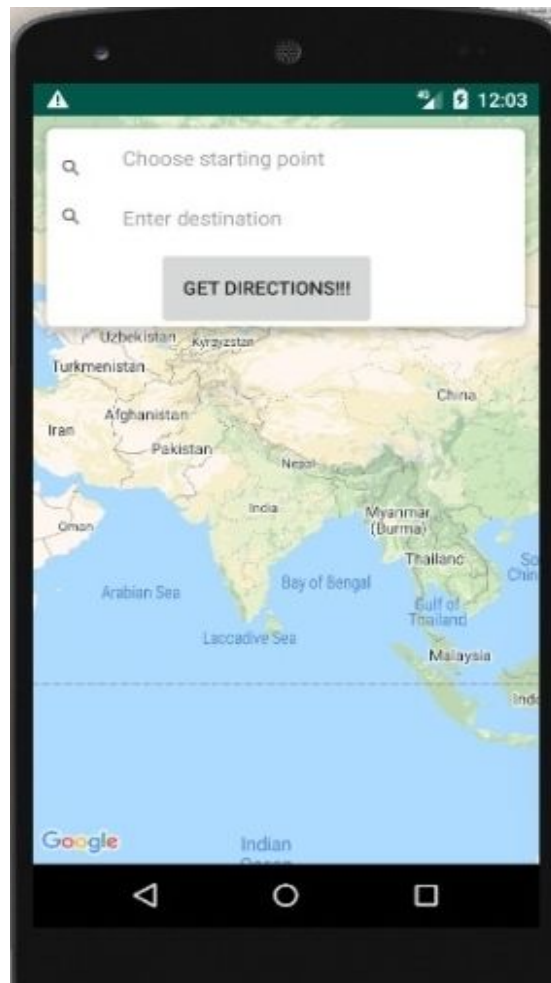


Fig 4.24: Search bar screen to enter the starting point and destination

- 5) Based on the locations given by the user the app calculates the latitude and longitude points. The app sends these latitude and longitude points as a request to the Routing API. The return value is an array of latitude and longitude points indicating the route. The pollution in the route is calculated by taking the pollution levels from the server. The algorithm selects the best route out of these and plots that route onto the Map interface in the app. More in depth about the Least polluted route algorithm in the next section.



- 6) The route is visible to the user with different colour coding to show the different pollution levels in that part of the area.
- Blue for low levels of AQI
 - Yellow for moderate levels of AQI
 - Red for high levels of AQI

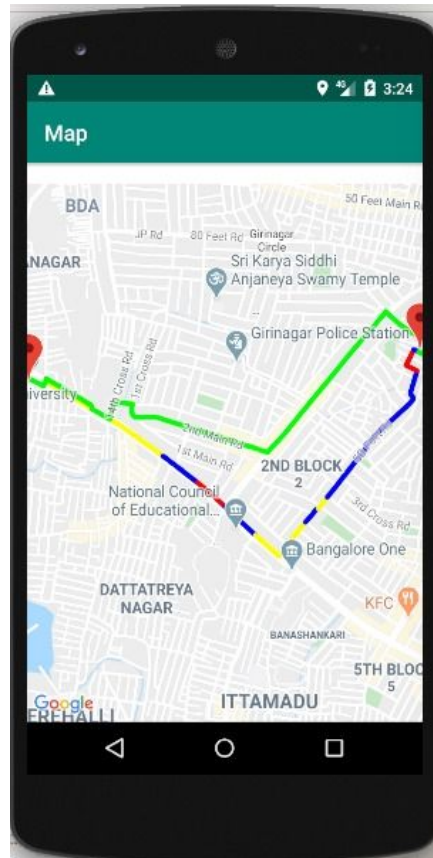


Fig 4.25: A map showing the best route to take based on the Least polluted route algorithm



4.6 Least Polluted Route Algorithm

This algorithm that we have written computes the best route or the route with the least pollution based on AQI level data from the server and the directions API.

This algorithm plots the route onto the google maps frame with the help of polylines (an object of the google maps API).

How the algorithm works:

1. The user enters the names of the source and destination of the route
2. These points are mapped onto the map frame by the app and then converted to latitude and longitude points. Automatically converting to latitude and longitude points is most important since only these will be used instead of location names, while plotting on the map frame or calling the routing API or creating polylines to show the route.
3. These are stored in variables and are used to make the API call to the routing API by the app
4. The routing API returns a set of four to five possible routes from the source to the destination

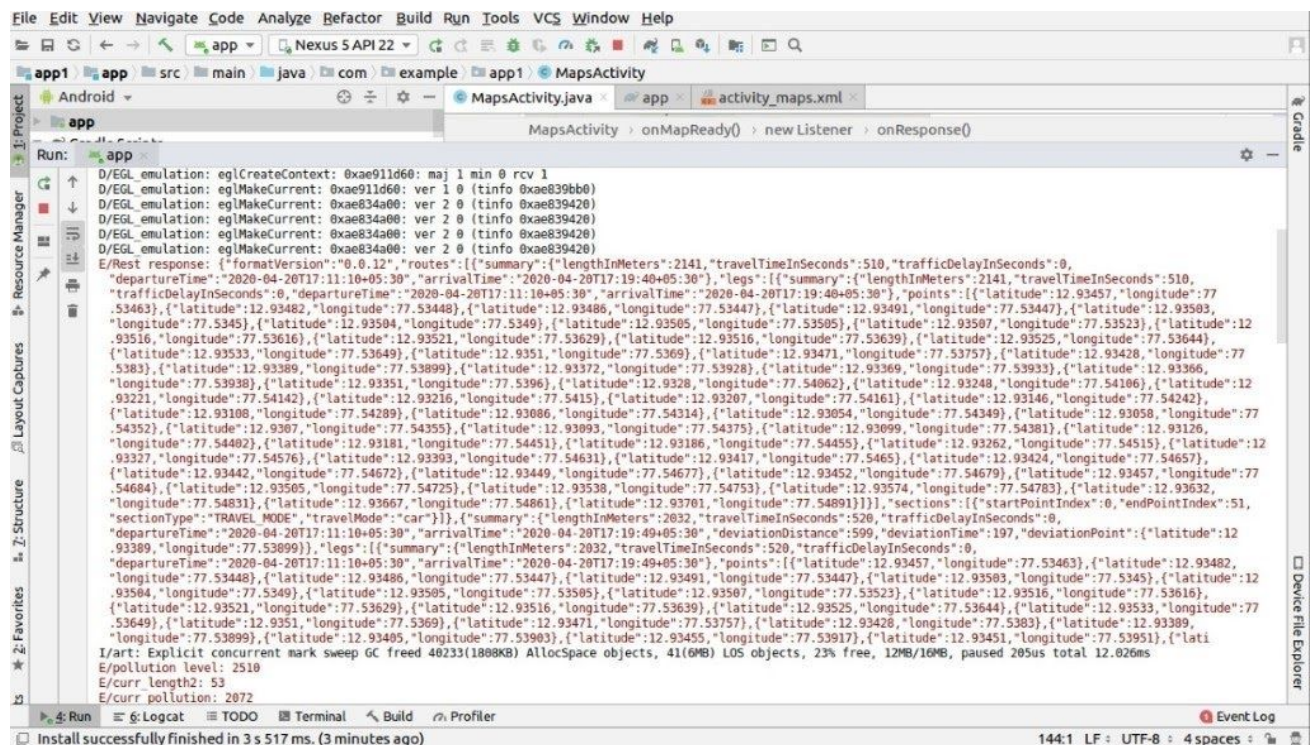


Fig. 4.26: The different routes in latitude and longitude format



5. All of these are in the format of an array of latitude and longitude points.
6. For each of these arrays the total pollution through the route is calculated. Each set of adjacent points in the array represents a small part of the route. For each of these sets of points the AQI level is taken from the server based on the latitude and longitude points in the server
7. Since the AQI levels have been recorded for each of the set of points, the areas with AQI levels more than 200 are identified and tagged as red
8. Similarly, for AQI levels between 100 to 200 are tagged as yellow and levels below 100 are tagged as blue
9. Based on these tags created for each route, the best one is chosen by calculating the route with the least amount of red tags and a portion of the yellow tags.
10. After choosing the best route, it is possible that some parts of the route may still be in the red region. To avoid these red areas, the algorithm tries to find slight deviations to the route.

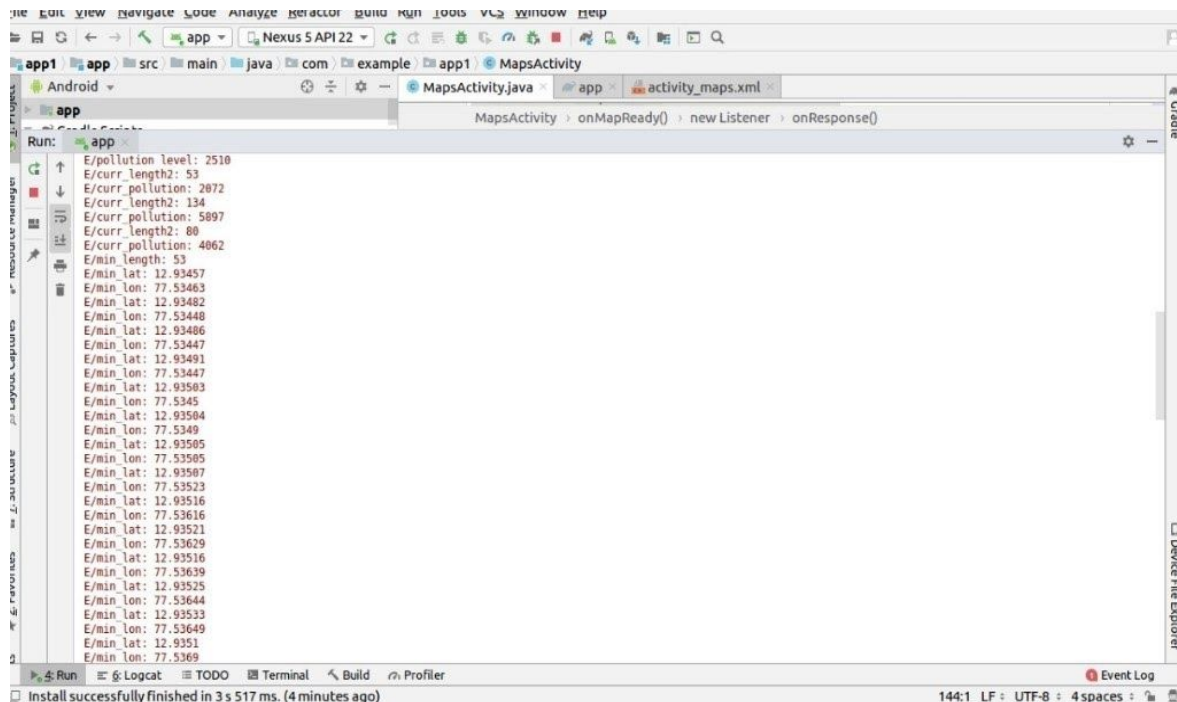


Fig 4.27: Pollution levels in different routes



11. For these slight deviations, the Dijkstra's shortest path algorithm has been used with pollution levels as weights
12. The final route after all the steps is still in the format of an array of latitude and longitude points
13. Using these sets of latitude and longitude points the polylines are plotted onto the map frame in the app.
14. The colour of these polylines is based on the tags that have been assigned previously, along with green coloured lines for the deviations in the route suggested.



4.7 Data Visualization

Data visualization is the realistic portrayal of information. It includes creating pictures that convey connections between the data collected, to the users. This correspondence is accomplished using a deliberate mapping between realistic imprints and information esteems in the making of the representation.

Based on the data stored in the server from their previous navigation we have created visualization on the average AQI level on Android Studio that is customized based on the travel history in the app used by every person. We show the user two kinds of graphs, first is based on the pollution that a person is exposed to everyday and the other is based on the month. The figures below give the view of the app.

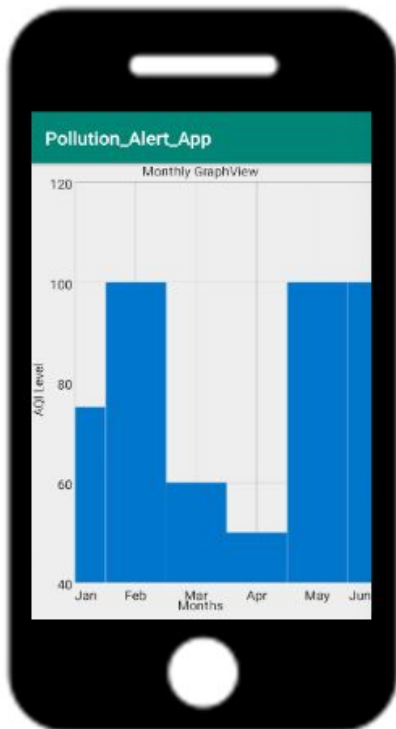


Fig 4.28: Data visualization based on monthly data.

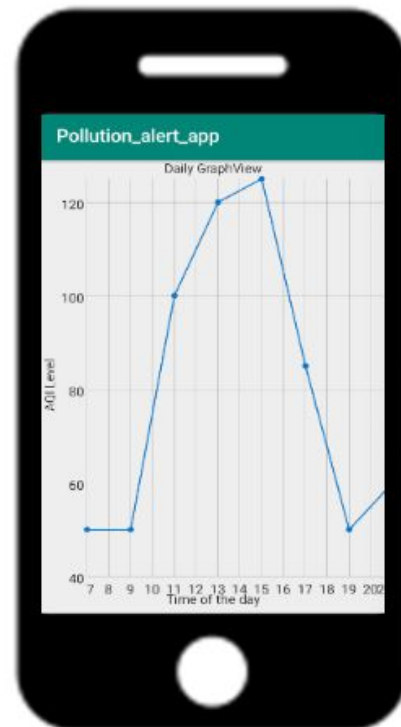


Fig 4.29: Data visualization on hourly basis



5. RESULTS

Using the data available from the server, we have created the Least Polluted Route Algorithm which gives the best route keeping in mind the distance and AQI levels.



Fig 5.1: Different routes possible from source to destination

The diagram above shows the different routes possible to take from a source location to the destination location based on distance and pollution levels.

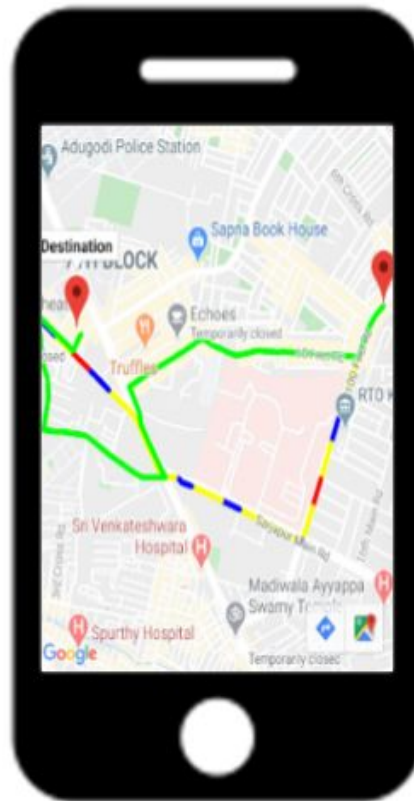


Fig 5.2: Best route taken by Least polluted route algorithm

In the above figure, the green line is the best suited route suggested by our algorithm which is giving weightage to both distance and pollution levels.

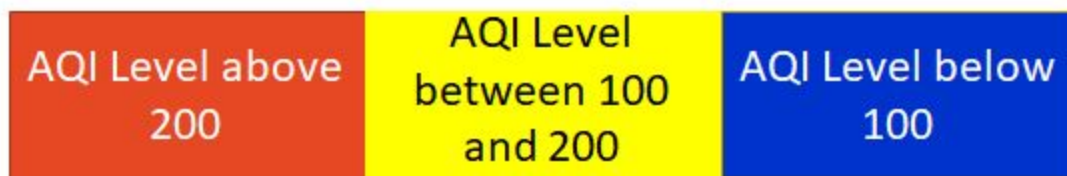


Fig 5.3: Colour Representations of AQI levels

In the map the colours on the route would suggest the pollution levels in that area taken from the data in the server. These are three colours we have taken to represent on the route. The Blue zone is not hazardous. The yellow zone suggests that users sensitive to pollution should avoid it. The



red zone is the hazardous zone. A notification is also given to the user that the pollution levels are high and they should take measures or stay indoor.

The routing algorithm along with the other features of the app help the user stay aware of the intake of pollution and the current status of pollution in different areas so that measures can be taken accordingly.



6. CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The Android app integrated with the Air Quality data on real-time basis provides deep insights of pollution levels in a given region. Citizens commuting in the city can be aware of the levels of various pollutants in and around their routes. Users armed with this information can make safer choices when they venture out on the road, for various purposes, like work, entertainment, walking/jogging etc. The simple and secure mobile app is able to detect and monitor pollution in a given vicinity, along with several simple visual indicators to avoid places with high pollution density. Besides, it provides and recommends alternate courses of routes from the source to the destination based on the pollution levels in those regions. With the monitoring of pollution with the help of a wearable device on the body, more accurate information is collected. Based on the exposure history of the pollutants to the individual, the app cautions individuals by customizing recommendations of routes and notifications for alerting the user in advance. With the pollution scenario in Indian cities being far from satisfactory, the app can be of great help to vulnerable individuals who need to restrict exposure to pollutants. The law makers can also use this app for information for taking remedial action to reduce pollution by making policy changes in the public transportation, operating hours of industries and so on.

6.2 Limitations

In Real-time air quality data monitored by the central and various state Pollution Control Boards, not all pollutants are monitored at all stations leading to gaps in the data. However the deployment of a wearable pollution monitoring device can close the gaps, since the pollution measured can be tracked at all locations accurately. This will guarantee that the pollution information is precise based on the areas, as opposed to monitoring devices being placed in remote areas.

The app and alerts have been designed only to include Bangalore city, currently. The data collected for each location is not very precise due to hardware limitations. The routing algorithm can be made more accurate, though it would be computationally expensive to train. Also, with more amounts of the pollution data collected, the algorithm can be made better by including the trends of the pollution levels for computations.



6.3 Applications

Taking this app forward into people's lives we want to create an awareness about Air quality helping people make safe choices when they leave their homes. For an effective air quality management system it is essential to have an ambient air quality monitoring system to collect data which helps in

1. Assessing the extent of pollution and develop a warning system or alarm when heavy smog is forecasted helping the common man to stay indoors.
2. Evaluate the effectiveness of emission control strategies and help industries and factory areas to curb their pollution and stay within the threshold limit .
3. Provide information on air quality trends and support implementation of air quality standards
4. Provide data for analysis of air quality models and support long term studies of the effects of air pollution on health.

6.4 Future Scope

The scope of the project can be further widened by including the routing maps to more cities like Bangalore. We also encourage and motivate more people to use the device so that we can crowd-source the data and use it for computation of pollution levels with more accuracy, both temporally and spatially. Based on this data the app can be customised for each user specifically, thereby making it more user-friendly and personalized. Users with respiratory disorders or such disorders that are sensitive to pollution levels can breathe a huge sigh of relief, by receiving notifications, alerts of potential and alarming levels in the regions of their travel or movement for avoiding or taking remedial measures. The pollution monitoring devices can be made more precise, reliable, durable and power-conserving in wearable forms.



7. REFERENCES

1. Md Nazmul Hoq, Rakibul Alam, Ashraful Amin “Prediction of possible asthma attack from air pollutants: towards a high-density air pollution map for smart cities to improve living” Presented at 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), 7-9 February 2019
2. Swati Dhingra, Rajasekhara Babu Madda, Amir H. Gandomi, Senior Member, IEEE, Rizwan Patan, Mahmoud Daneshmand, Life Member, IEEE “Internet of Things Mobile - Air Pollution Monitoring System (IoT-Mobair)” presented at IEEE internet things of journal, 2019
3. Sagar V Belavadi, Sreenidhi Rajagopal, Ranjani R, and Rajasekar Mohan , “Air Quality Forecasting using LSTM RNN and Wireless Sensor” presented at the 11th International Conference on Ambient Systems, Networks and Technologies, 2020.
4. David Hasenfratz, Olga Saukh, Silvan Sturzenegger, and Lothar Thiele Computer Engineering and Networks Laboratory ETH Zurich, Switzerland “Participatory Air Pollution Monitoring Using Smartphones” presented at the 2nd International Workshop on Mobile Sensing, 2012.
5. Liu, X. M., F. F. Wang, and Z. B. Zeng. "Design and Implementation of Indoor Environmental Quality Monitoring System Based on ZigBee." In International Conference on Computer Information Systems and Industrial Applications (CISIA 2015)
6. Shaban, Khaled Bashir, Abdullah Kadri, and Eman Rezk. "Urban air pollution monitoring system with forecasting models." IEEE Sensors Journal 16
7. L. Morawska, P. Thai, X. Liu, A. Asumadu-Sakyia, G. Ayoko, A. Bartonova, A. Bedini, F. Chai, B. Christensen, and M. Dunbabin, “Applications of low-cost sensing technologies for air quality monitoring and exposure assessment: how far have they gone?,” Environ. Int., 2018.
8. Kadri, A.; Yaacoub, E.; Mushtaha, M.; Abu-Dayya, A. ”Wireless sensor network for real-time air pollution monitoring.”. In Proceedings of the 2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA), Sharjah, UAE, 12–14 February 2013
9. G. Lo Re, D. Peri, and S. D. Vassallo, “Urban air quality monitoring using vehicular sensor networks,” in Advances onto the Internet of Things, Springer, 2014
10. Directions API: <https://developer.tomtom.com/>
11. Google API: <https://developers.google.com/maps/documentation>
12. Real time Air Quality Index: data.gov.in/catalog/real-time-air-quality-index



APPENDIX

The source code of the application has been uploaded to a github repository. To run this application, the repository can be cloned or downloaded. Android studio software is required to run this code.

https://github.com/ramyaakella/Pollution_alert_app

Word Count: 5746

Plagiarism Percentage 14%



Matches

1

World Wide Web Match

[View Link](#)

2

World Wide Web Match

[View Link](#)

3

World Wide Web Match

[View Link](#)

4

World Wide Web Match

[View Link](#)

5

World Wide Web Match

[View Link](#)

6

World Wide Web Match

[View Link](#)

7

World Wide Web Match

[View Link](#)

8

World Wide Web Match

[View Link](#)

9

World Wide Web Match

[View Link](#)

10

World Wide Web Match

[View Link](#)

11

World Wide Web Match