IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 1, January 2022

DOI: 10.17148/IJARCCE.2022.11123

Project YU: An Economic & Scalable approach for monitoring Air Pollution

Karan Gopal Gaur[1], Mayank Tiwari[1]

Computer Science Engineer, Department of Computer Science & Engineering

Swami Keshvanand Institute of Technology, Management & Gramothan, Ram Nagariya Rd,

Shivam Nagar, Jagatpura, Jaipur-302017, (R.J), India[1]

Abstract: Air pollution is one of the biggest dangers to human health but even then the data related to it is usually limited and mostly not accurate or updated. Commonly, there are only 3-4 pollution monitoring stations in each city which means that the data that we get from these stations are accurate only to close proximity of the monitoring station and may vary considerably just a few kilometres away from the station. The purpose of the paper is to present the choice of architecture and the implementation of a mobile platform which by using connected sensors collects real-time air quality data. The data can then be provided to the general public where primarily people suffering from diseases related to respiration like asthma can benefit from real-time information about air quality.

Keywords: Air quality, Sensors, Location Based Services, Android application, Raspberry Pi, Arduino, Air Pollution.

I. INTRODUCTION

Project YU is a combination of both hardware and software technologies working in a tightly integrated environment that aims to provide an economically viable and easily scalable solution for monitoring air pollution, especially in the individual sections of a city. This approach helps to build a detailed map of the air quality in the city which can prove to be a key element in the development of countermeasure strategies. Also, it's designed to have a low initial setup and overall operational cost so it can be utilised even in countries with a limited budget and the hardware used in the project has a minimum energy requirement which helps to limit the carbon footprint as well. Basic operation involves the collection of pollution data using portable devices consisting of a microcontroller and various gas sensors, this data is then sent to the servers from where the end-user can access it using the mobile app.

II. SOFTWARE/HARDWARE USED

A. VSCode Editorial - VS Code is a well-known code editor created by Microsoft that comes with a variety of useful features and tools to assist in the development. VS Code has vast support and extensions to work with git, javascript, express, databases, etc. - which assists in the development and version control.

B. Express - Express is a lightweight and flexible Node.js web application framework that provides a powerful set of features for web and mobile applications development. Its ability facilitates the rapid development of Node based Web applications is why we have used it for our server implementation.

C. Ionic - In our project, we have used Ionic for building an android app for easy access to the pollution data. Ionic is an open-source mobile application development platform for building high quality, cross-platform native and web apps. Ionic Framework focuses on the frontend part (UI and UX interaction of an app)— UI controls, animations, interactions, gestures.

D. MongoDB - For database, we have used MongoDB in our project which is a popular unstructured NoSQL database. Its fast query response time and high flexibility make it an ideal choice for our project.

E. Arduino Nano - Arduino is an open-source electronic prototyping platform that comes in a variety of shapes and sizes. In our project, we have used Arduino Nano for collecting data from various gas sensors which is the smallest microcontroller in the Arduino family. Its small size helps to keep the monitoring device compact and mobile.

F. Raspberry PI - Raspberry PI is a small credit-card size single-board computer. Its low price makes it a popular choice for prototyping. Since the microcontroller that we have used in our projects is not capable of relaying the data

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 1, January 2022

DOI: 10.17148/IJARCCE.2022.11123

collected to the server via the internet so we used raspberry pi to fetch that data from the microcontroller and then send it to the server. The model that we have used is Raspberry Pi 3B+ which comes with a 1.4 GHz 64-bit quad-core processor and 1 GB of RAM

G. Gas Sensors - We have used a variety of gas sensors to detect the concentration of various gases in the atmosphere. Currently, we are using MQ7, MQ135 and MQ136 to detect gases including CO2, CO and H2S.

III. DEVICE CONSTRUCTION

Device hardware consists of two main parts - the microcontroller (Arduino Nano) and the SBC (Raspberry PI 3B+). Arduino reads the analogue data received from the sensors attached to the analogue pins and then sends it to raspberry pi for transmission to the server using the raspberry pi's onboard ethernet connection. Currently, the device uses three different metal-oxide based gas sensors which are sensitive to CO2, CO and H2S and can give out a general value for the air quality through the change in the resistance of the material used inside the sensor, which can be detected as a change in output voltage by Arduino. Based on this voltage value we can easily find the type and concentration of the gas can also be estimated. For power, the device can either be powered by a 5 volt DC power brick or through a 5 volt AA battery pack so it can remain mobile.

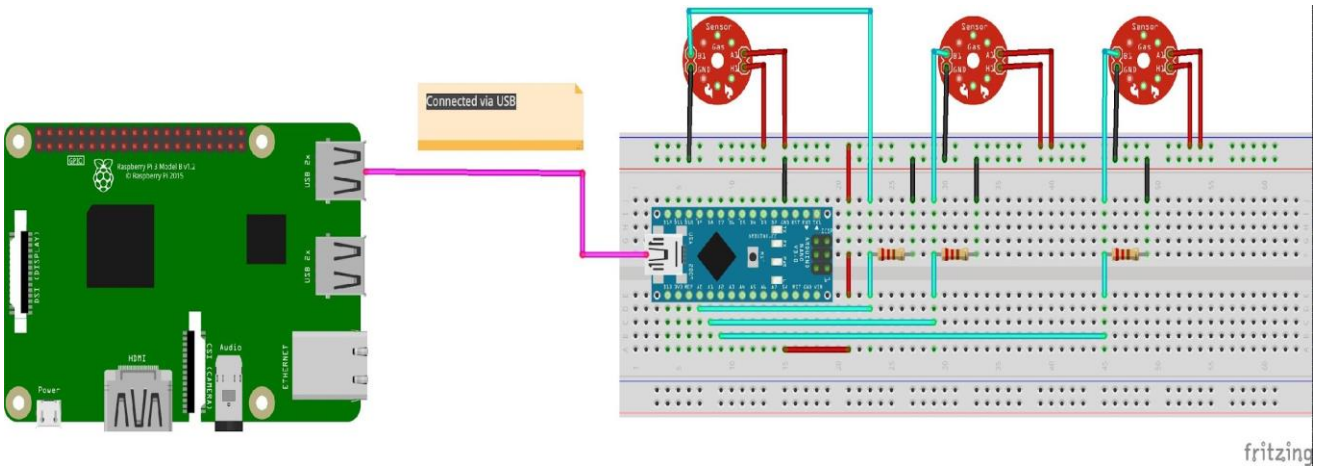

Fig 1 - Monitoring Device Internal Wiring

For estimating the PPM of each gas detected by the sensors we used the "sensitivity characteristics graph" for each sensor provided in the datasheet of every sensor. Below shown is the graph for MQ-135. Here PPM can be estimated using the following formula.

$$\mathbf{PPM = a*(R_s/R_o) \wedge b}$$

With the use of power regression we can find out the scaling factor (a) and the exponent (b) for the gas we would like to measure. Then we can calculate R_0 using the below formula.

$$\mathbf{R_o = R_s * sqrt(a / ppm, b) = R_s * exp(ln(a / ppm) / b)}$$

So, for calibrating the sensor the only requirement is the known amount of certain gas, then from the sensor, we can read the output resistance value (R_S) and can easily compute the R_0 Value.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 1, January 2022

DOI: 10.17148/IJARCCE.2022.11123

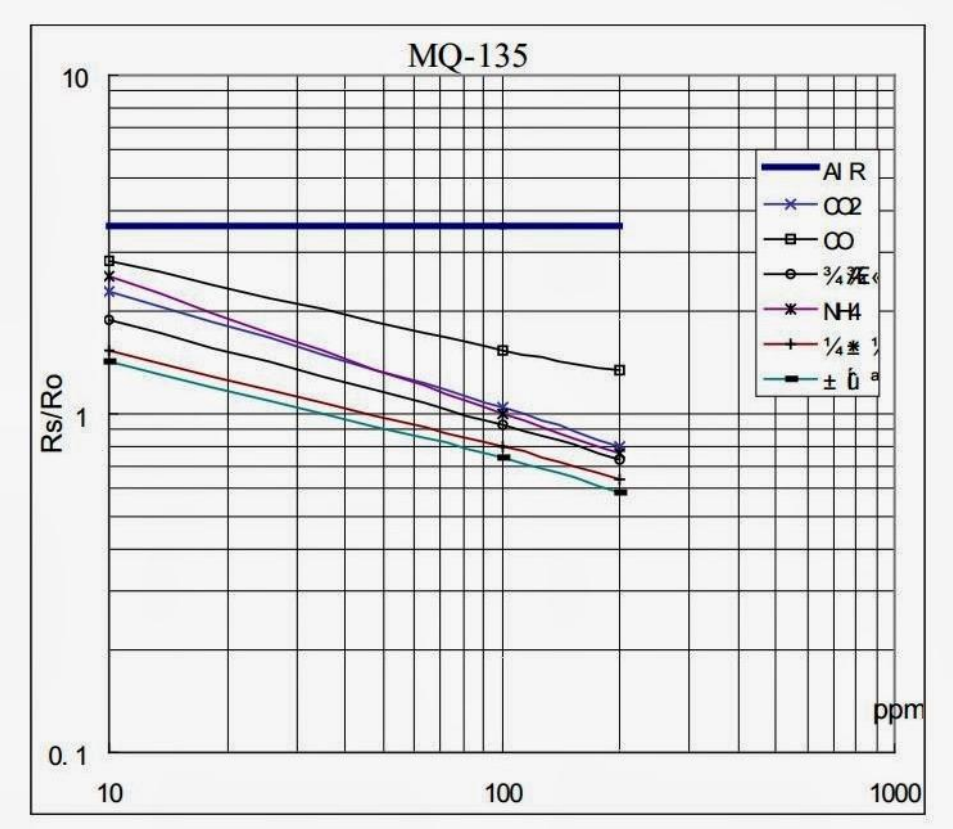

Fig 2 - Sensitivity Characteristic Graph

IV. SERVER DESIGN

The most important part of the whole system is the server as it is responsible for not only the collection of data from different monitoring devices but also for storage and serving of data to the end-user. For the implementation, we have used javascript as our language of choice and the server is built on top of the NodeJS runtime environment using the ExpressJS framework.

Here, the advantage of using NodeJS is that we can perform non-blocking I/O operations using its event loop mechanism by assigning operations to the operating system whenever and wherever possible and since most of the operating systems are multi-threaded they can easily handle multiple operations executing the background. When one of these operations is finished, Kernel relays this information to the NodeJS and the respective callback assigned to that operation is added to the event queue which will eventually be executed. This implementation helps to ensure that the server remains equally efficient even in a high load.

Another important part that can prove to be a bottleneck for the server performance is the database and therefore choosing a perfectly suitable database is equally essential. Here, we have used MongoDB for our database which is a schema-less database. It provides faster access to the data due to its nature of using internal memory for the storage and unlike the relational database, it's quite easy to tune its performance. In MongoDB, all the data is stored in a document-based data model in the form of binary JSON (BSON), ruby hashes etc so it also eliminates the need for implementing complex joins.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 1, January 2022

DOI: 10.17148/IJARCCE.2022.11123

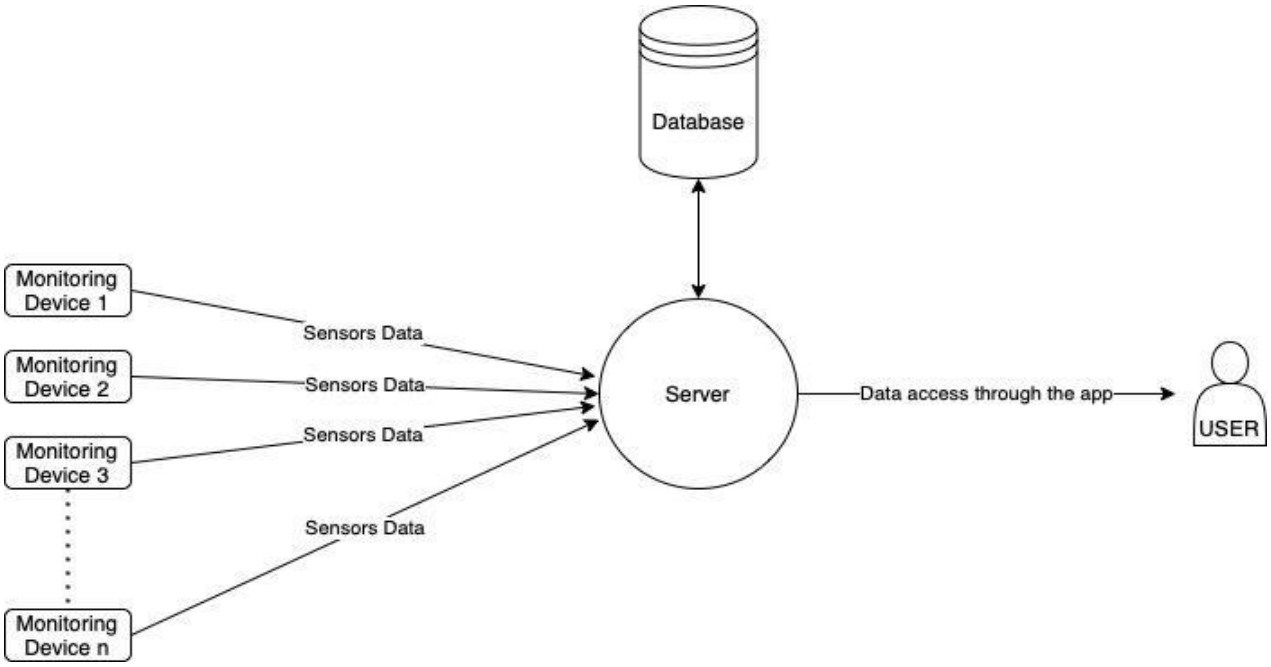

Fig 3 - Overall workflow of the system

V. MOBILE APPLICATION

After the implementation of the server, the final logical thing that remains to do is to provide an easy and convenient way for the end-user to access this data and for this purpose, we have designed an android application that provides a location-wise list of the air quality data from the user's current city.

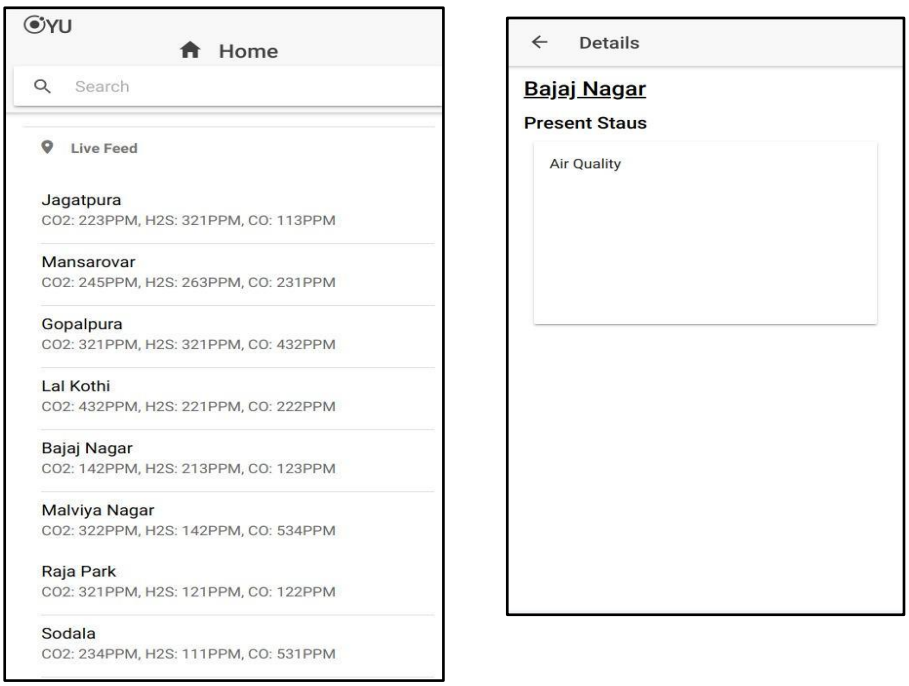

Fig 4 - YU Android Application

Impact Factor 7.39 ⁜ Vol. 11, Issue 1, January 2022

DOI: 10.17148/IJARCCE.2022.11123

The data provided by the app is always real-time data and the mobile app can also push alert notifications to the user if the user is in a high pollution area to ensure the safety of the end-user. The advantage of building a mobile app over a standard website is that the user can access the real-time data much more quickly by simply launching the app rather than going to a browser and punching in the website link first. It also makes it easier to provide personalised notifications to the user and provides an overall rich and convenient experience to the user.

For building the mobile application we have used the Ionic framework. The reason behind using Ionic is that it is a cross-platform framework and works with Android, iOS and also supports UWP (Universal Windows Platform) and uses HTML, CSS and Javascript for building the application which is all very easy to work with. Another advantage of Ionic is the performance benefits, Using Ionic with native mobile app code in PhoneGap (Apache Cordova) allows for higher performance in comparison to hybrid applications. AngularJS allows Ionic to rely on native hardware acceleration. Ionic uses CSS transitions as a way to leverage the GPU and maximise available processor time and since Ionic uses Cordova plugins we can use it to access the native

VI. FUTURE IMPROVEMENTS

There are a few things which can be improved in the current system to make it more efficient and effective some of which are outlined below

A. Using the data acquired over a period of time to predict climate changes and future atmospheric conditions.
B. Detection of other gases responsible for air pollution which are not being tracked by the device currently (possibly with the use of additional sensor modules).
C. Detection and measurement of pm 2.5 and pm 10 in the air.
D. To make the device more compact and economical different microcontrollers can be used for example a node MCU can be used to replace both Arduino and raspberry pi.
E. Sensor readings can be made more precise by using graphical data to convert analogue readings into ppm. Currently, the system is using pre-built libraries for the respective sensor which do not give exact readings but a close approximate.
F. MQTT protocol can be used to replace the current protocol used in clustering of the device as it is more suitable for low bandwidth applications and is specially designed for microcontroller systems.
G. An IOS application can also be designed similar to the android application to increase accessibility.

VII. REFERENCES

A. World Health Organisation, “Ambient air pollution: A global assessment of exposure and burden of disease.” 2016.
B. World Health Organisation, “Ambient (outdoor) air quality and health,” WHO Media Centre, Sep-2016.
C. The World Air Quality Index project, “Air Pollution in Sweden: Real-time Air Quality Index Visual Map.”.
D. The World Air Quality Index, “Real-time World Air Quality Index,”. [Online]. Available: https://waqi.info/.
E. The European Commission, “GOEASY, GalileO-based trusted applications for health and sustainability,”. [Online]. Available: http://cordis.europa.eu/project/rcn/212432_en.html.
F. Future Electronics, “What is a Single Board Computer?”. [Online]. Available: http://www.futureelectronics.com/en/display-solutions/single-board-computer.aspx.
G. Raspberry Pi Foundation, “Raspberry Pi - Teach, Learn, and Make with Raspberry Pi,” Raspberry Pi. [Online]. Available:https://www.raspberrypi.org/.
H. Arduino, “Arduino - Home,”. [Online]. Available: https://www.arduino.cc/.
I. Raspberry Pi Foundation, “Raspberry Pi Desktop,” Raspberry Pi.
J. Andrew Chalkley, “The Absolute Beginner’s Guide to Arduino,”. [Online]. Available: http://forefront.io/a/beginners-guide-to-arduino/.
K. National Oceanic and Atmospheric Administration, “GPS & Selective Availability Q&A.”
L. European Space Agency, “What is Galileo?” European Space Agency.
M. United States Environmental Protection Agency, “Health and Environmental Effects of Particulate Matter (PM),”
N. Government of New South Wales, “Particulate Matter (PM10 and PM2.5),” 29-Apr-2013.[Online]. Available: http://www.health.nsw.gov.au/environment/air/Pages/particulate-matter.aspx.
O. The United States Environmental Protection Agency, “Air Quality Index(AQI),” presented at the Air Quality Communication Workshop, San Salvador, El Salvador, 16-Apr-2012.

