**S**mart **A**ir-Quality **M**onitoring **S**ystem

**SAMS**



Table of Contents

[2 The BIG Idea 4](#_Toc3331811)

[2.1 Scary facts about Indoor Air Quality 4](#_Toc3331812)

[2.2 The Idea 4](#_Toc3331813)

[2.3 Existing Solutions 5](#_Toc3331814)

[3 Research Questions 5](#_Toc3331815)

[3.1 What are the different parameters to be measured for Indoor Air Quality? 5](#_Toc3331816)

[3.2 What are the different types of sensors easily available to measure different air quality parameters and how are they programmed? 6](#_Toc3331817)

[3.2.1 DHT22 sensor - measures humidity and temperature 6](#_Toc3331818)

[3.2.2 SGP30 sensor - measures CO2 and TVOC 6](#_Toc3331819)

[3.3 How can the data be sent and saved to a Web Server? 7](#_Toc3331820)

[3.4 What would be required to create graphs on webpage? 7](#_Toc3331821)

[3.5 How can appliances be switched ON or OFF programmatically? 8](#_Toc3331822)

[3.6 Are there existing products that perform these functionalities? 8](#_Toc3331823)

[3.7 What functionalities are not performed by these products? 8](#_Toc3331824)

[4 Engineering Goals 9](#_Toc3331825)

[5 Design 10](#_Toc3331826)

[5.1 Detector: 10](#_Toc3331827)

[5.2 Web Server: 10](#_Toc3331828)

[5.2.1 Install & Configure 10](#_Toc3331829)

[5.2.2 Web-Server Methods 11](#_Toc3331830)

[5.3 Webpage 11](#_Toc3331831)

[5.4 Automating a power switch 11](#_Toc3331832)

[6 Procedure 12](#_Toc3331833)

[6.1 Wiring the Sensors 12](#_Toc3331834)

[6.2 Checking the Wireless connection between the Detector and Webserver 12](#_Toc3331835)

[6.3 Sending, receiving, logging and reading data 12](#_Toc3331836)

[7 Risk and Safety: 13](#_Toc3331837)

[8 Data Analysis 13](#_Toc3331838)

[9 Challenges in implementation 13](#_Toc3331839)

[9.1 Raspberry Pi not getting connected to the laptop 13](#_Toc3331840)

[9.2 Sensors going offline 15](#_Toc3331841)

[10 Future Research and Improvements 16](#_Toc3331842)

[11 Bibliography 16](#_Toc3331843)

Smart Air-Quality Monitoring System

**SAMS**

# The BIG Idea

Indoor air quality is an invisible threat. People are unaware of the real condition of the indoor air, be it humidity, temperature or pollution. Poor indoor air quality can cause serious respiratory infections or allergies.

## Scary facts about Indoor Air Quality

1. The air quality indoors can be up to a 100 times worse than the air quality outside.
2. Pediatric Asthma rates up jumped by 72%.
3. The EPA\* ranks indoor air pollution within the top 5 environmental dangers.
4. An average human spends 90% of their time indoors.

EPA\* = Environmental Protection Agency

## The Idea

The idea is to inform the user of the indoor air quality, in terms of temperature, humidity and any harmful gases present indoors. Furthermore, if the user already has a humidifier or an air purifier, he/she can switch these appliances ON or OFF automatically, based on certain rules/triggers set by the user on a website.

## Existing Solutions

There are many air quality monitors so what is so special about SAMS?

SAMS is much cheaper - under $50. The indoor air quality could vary from room to room. For example, the air quality of your living room would be very different from the attic, or the air quality of your bathrooms would be very different from your kitchen. The only way to accurately measure the indoor air-quality would be to have multiple monitoring units – one per room. Most of the commercial products similar to SAMS are very expensive (“Foobot” or “Google Nest” can cost $200+ per unit) And with multiple rooms; that will surely make a dent in your wallet. SAMS will be relatively inexpensive solution ( 75% less costly)

Secondly, SAMS does something that none of the commercial products do effectively. These commercial monitors measure the indoor air-quality, but don’t act on it. For example, if they detect that the air-quality is bad, they would not switch on an existing air-purifier. At best they would just warn the user. SAMS will power-on and power-off existing appliances ( and these appliances need not be ‘smart’ appliance)

# Research Questions

## What are the different parameters to be measured for Indoor Air Quality?

There are various different parameters that help in measuring the air-quality.

|  |  |  |
| --- | --- | --- |
|  | Measured In... | What is it? |
| CO2 | Particles Per Million  [350-1000 ppm - Normal](https://www.kane.co.uk/knowledge-centre/what-are-safe-levels-of-co-and-co2-in-rooms) | Carbon dioxide content in air. Higher concentrations on CO2 could be very harmful for human physical and cognitive health ([NASA](https://ston.jsc.nasa.gov/collections/TRS/_techrep/TM-2016-219277.pdf)) |
| TVOC | ug/m3  (micro grams per cubic meter)  [Below 300 ug/m3 - Normal](https://easlab.com/iaqref.htm) | Total Volatile Organic Content - indoor Volatile Organic compounds could be propane (from cooking gas), ethanol (from various cleaning products), acetone ( from cosmetic products ) etc. There is a complete list of these products available at the [EPA website](https://ofmpub.epa.gov/sor_internet/registry/substreg/substance/details.do?displayPopup=&id=83723) |
| Temperature | °Celsius / °Fahrenheit | Indoor Temperature |
| Relative Humidity | Percentage (%) | It is the ratio of the current absolute humidity to the highest possible absolute humidity |

## What are the different types of sensors easily available to measure different air quality parameters and how are they programmed?

### https://lh5.googleusercontent.com/IXZ3d4lv00l88PTtQa-86qlmtnYNCgT5owdZF3_g8bFM9Fp8Tr0US_7INJhB98PIUVjgtkLd3sPzyc0mvuih4w3zqTuwxNyy6e6xqTvcf7uxj0hORHfpAMstQ6DDS464XhnsboonDHT22 sensor - measures humidity and temperature

DHT22 is a Temperature and Humidity sensor. The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller.

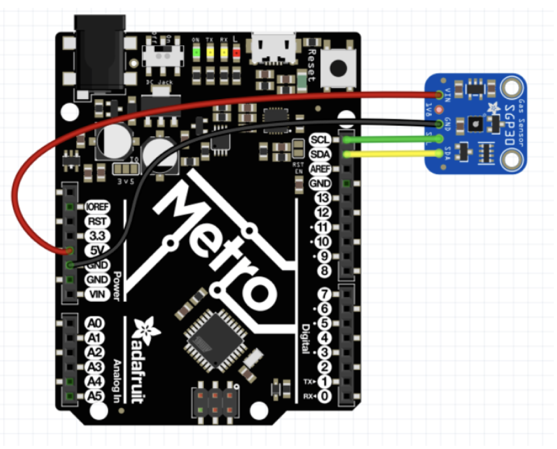
The leftmost pin of this sensor is the power pin and is connected to the 3.3 volt pin of the Raspberry Pi. The second leftmost pin is the data line and is connected to pin 7 of the Raspberry Pi or GPIO4, this sends information of the humidity and temperature to the Pi

Finally, the rightmost pin connects to pin 6 GND, and the second rightmost pin remains free and is there for extra stability.

More details can be found at <https://learn.adafruit.com/dht/overview>

### SGP30 sensor - measures CO2 and TVOC

The SGP30 has a 'standard' hot-plate Metal Oxide (MOX) sensor, It has a small microcontroller that controls power to the plate, reads the analog voltage, tracks the baseline calibration, calculates TVOC and eCO2 values, and provides an I2C interface to read from. The eCO2 is calculated based on H2 concentration, it is not a 'true' CO2 sensor.In indoor environments, the H2 concentration is expected to correlate well with the CO2 concentration as human breath contains significant concentrations of both, CO2 (4%) and H2 (10 ppm). Furthermore, humans are the only major source of CO2 and H2 in typical indoor environments. (<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5948493/> )

**Power Pins:**

Vin - this is the power pin.

GND - common ground for power and logic

**Data Pins**

SCL - I2C clock pin, connect to your microcontrollers I2C clock line. It has a 10K pull-up  resistor to Vin

SDA - I2C data pin, connect to your microcontrollers I2C data line. It has a 10K pull-up  resistor to

Vin

## How can the data be sent and saved to a Web Server?

1. Nodejs is a very simple web-server that is available on Ubuntu and Windows. The interesting aspect of this web-server is that it is very simple to program. It uses Javascript as the programming language ( which is the same programming language used in the web-page).
2. A python program running the Raspberry Pi can measure data and send it to the Node.js server on the laptop.
3. There are methods/functions on the Node.js that could be called from the Raspberry Pi program. These methods are called “REST” APIs. These methods could be called by any program that connects to the Node.js web-server.
4. As these are methods- they accept parameters too.
5. The data would be stored in a MySQL database in form of tables (columns / fields) and records (rows) ( very much like Excel)
6. The Raspberry Pi sends the air quality data to a node.js web server running on the laptop. Then the node.js server stores the data in a MySql database and displays it on a webpage.

## What would be required to create graphs on webpage?

There is a JavaScript library called “Plotly.js”. This allows us to create a realtime graph of the air quality values.

## How can appliances be switched ON or OFF programmatically?

The Raspberry Pi can be connected to a relay switch and the relay switch can be connected to a humidifier or other air purifying device. This way the Pi can automatically turn on and off other devices to make the air quality better.

## Are there existing products that perform these functionalities?

1. Foobot is the only commercially product that comes close to these functionalities. It does measure the temperature, humidity, CO2 and TVOC. It has a website where a user could login and monitor the indoor air quality.
2. Foobot does claim to control a smart device like NEST. However, there was no documentation on how it could be done. It looked like a feature that would come in future release of the product. The product specification from their website does not show this as a feature <https://foobot.io/foobotspecs.pdf>
3. I do not own a Foobot, hence the research is based on the material available on the foobot.io and various other independent review available on youtube.com

## What functionalities are not performed by these products?

1. Foobot cannot take actions on non-smart devices  ( like a regular dehumidifier or air purifiers). This means for someone to regulate the indoor air quality, they will have to also invest in smart (and expensive) devices and cannot reuse their existing appliances
2. A single foobot , costing $199 can cover a flat area of 1500 square feet. Typically a home has many rooms, attic, garage. The type of indoor air quality in different places in a home might be very different, for example - bathroom might have more TVOC value as perhaps cosmetic sprays might be used, for air freshening spray might be used.

# Engineering Goals

1. The device should be able to measure the indoor temperature, humidity and the presence of harmful gases.
2. The system should be able to detect and record the measurements from devices installed at different rooms (or indoor locations - like attic or garage or bathroom).
3. The user should be able to view the aggregated or specific recorded data in a visual manner on a website, in near real time.
4. The system would warn the users if the air-quality parameters are outside the defined range.
5. The user should be able to set threshold/ranges for different air-quality parameters.
6. Based on these rules, the system should automatically switch ON or OFF the appliance (like a humidifier, or air purifier or a heater). These appliances need not necessarily be expensive Smart IOT devices.

# Design

There are 4 major components of the system

## Detector:

1. This is a Raspberry Pi 3 based system. There are 2 sensors (DHT22 & SGP30) that are wired to the Raspberry Pi.
2. There is a software code that is running on Raspberry Pi. It periodically reads the data from these sensor and sends the data to the server.
   1. For reading the data from DHT22, there is a library provided by a vendor Adafruit.
   2. For reading the data from the SGP30 sensor, i2C protocol is used. The details of the commands are provided by the sensor datasheet
   3. The program is developed in Python 3 programming language
3. The data read from the sensor is send to the web server using the http protocol
   1. http “post” method is used to send the data
   2. The data is packed in a JSON format – which has the timestamp, CO2, TVOC, Temperature & Humidity values.
   3. Care has been taken to handle error conditions -like if the server is not running or not reachable.
   4. If the webserver is not reachable, the program will continue measuring the values

## Web Server:

### Install & Configure

1. The web-server is developed using Node.js free javascript based webserver.
2. The Node.js server was installed and configure on the Ubutun 18.04 laptop
3. Some additional packages like ‘mysql’ for database, and ‘express” were installed using the ‘npm’ ( ‘node package manager’)
4. The websever runs on port ‘3000’. The Raspberry Pi detector sends data to this IP address and 3000 port number
5. MySQL database server was also installed on the Ubuntu laptop.
   1. There is a ‘sams.db’ database created with specific username and password
   2. There is a table – AirQuality that has various fields like
   3. Room# (INT) , Room-name (VARCHAR(25)) Timestamp (DATETIME) , CO2 (FLOAT), TVOC(FLOAT), Temperature (FLOAT) & Humidity(FLOAT)
   4. The table is pre-created using a mysql script called ‘sams.sql’

### Web-Server Methods

The webserver has 3 basic methods

1. The first operation after starting it does is to connect to a MySQL database. It has the details of the database and the credentials
2. There is “post\sgp30” method. This method accepts the date from the Raspberry Pi. It then extracts the parameter from the JSON payload
3. Runs an “insert” query to log the data into the database. This will insert a single row for each measurement into the airquality table
4. There is ‘get\sgp30’ method. This method returns all the recorded data. This method is called from the webpage. When this method is invoked, it will run a “SELECT” query to readout all the records from the airquality table

## Webpage

1. The webpage is an HTML page with stylesheets, a table and some Javascript
2. The stylesheet is created for the table – color, shadow, border-type
3. The Javascript is used to call the webserver, collect the data and add each record as a row into the html table.
4. In future, this page will have login control, and real time graphs (using plotly.js)

## Automating a power switch

1. This would be a Raspberry Pi based system that can switch an appliance ON or OFF based on a command send by the Server.
2. It will have a programmable relay switch.

# Procedure

## Wiring the Sensors

1. Each device has sensors connected to the GPIO pins of the Raspberry Pi.
2. The datasheet of the sensors gives the specific pins and circuit details.
3. Each sensor is individually tested first to see if the pins and circuits are connected correctly. This is done by creating the circuit on a breadboard first.
4. SGP30 sensor used i2c interface on Raspberry Pi. Hence I had to run a tool called ‘i2cdetect’ to figure out the port number for SGP30.
5. Next, both the sensors are connected to the Raspberry Pi via the breadboard.
6. Once, the sensors reliably measure data. (sensors not going offline), next thing is to solder the sensors to the Raspberry Pi.
7. Python program is run for a long duration to see the results on the Python 3 console. SGP30 takes approximately 10-20 seconds to start showing proper results. This is due to the fact that it takes some time to heat up.

## Checking the Wireless connection between the Detector and Webserver

1. A separate Python test program was written to make sure that a message can be passed from the detector to the webserver that running on a laptop.
2. Error handling was put in place to make sure even if the server was not reachable the detector program continues to run.

## Sending, receiving, logging and reading data

1. The Python program that measures data, and the python program that sends data are merged into a single python program. This program continuously measures sensor data. After each measurement cycle is sleeps for 2 seconds
2. Data that is measured is shown on the Raspberry Pi Python 3 console
3. Data that is received on the Node.js webserver is shown the on an Ubuntu terminal.
4. Once the above steps were successfully tested with various error conditions ( like server down, or sensor disconnected or Raspberry Pi restarting), data is getting inserting to the database.
5. Next developed a simple webpage to connect the webserver and read data from the databas

# Risk and Safety:

This project is a prototype of an idea. The accuracy of the results depends a lot on the type of data reported by the different sensors. It will require lot of testing , with different sensor and conditions to measure data accurately. Due to this reason, the system might give some erroneous results.

# Data Analysis

1. The average data would be displayed with a line-graph on given interval. (collection might happen more frequently, but the data displayed would be averaged of that interval)
2. The data could be aggregated to show a daily report, weekly report or a monthly report.
3. There would be a line graph for each parameter (temperature, humidity, harmful gases)
4. The maximum and minimum measured value would also be shown.
5. In future, it can be modified to identify patterns

# Challenges in implementation

## Raspberry Pi not getting connected to the laptop

Each step of the process was a challenge as I did not find the information in a single place. Also, lot of the information on the Raspberry Pi site was outdated.

I had the following goals in mind regarding the connectivity of my laptop with Raspberry Pi -

* Connect Raspberry Pi over Wi-Fi
* The connection console should be a Graphical User Interface (GUI) and not a Command Line Interface (CLI ,like Secure Shell-SSH)
* I should be able to connect the Raspberry Pi via my Ubuntu 18.04 laptop.

The following steps, though a bit roundabout,  helped me achieve my above goals.

1. Unpacked the Raspberry Pi 3 kit. The  kit already came with a SD card which had the latest version of the Raspbian  and NOOBs OS.
2. Now, the  Pi had to be connected to a monitor and keyboard. I decided to use my Windows laptop first,  as the display and the keyboard for the Raspberry Pi.
3. I thought off starting of with a wired connection first. I used an Ethernet cable (RJ45) and directly connected the laptop’s network port to the Pi’s network port.
4. Next  on the Windows 10 laptop, I performed the following steps.
5. Go to “Control Panel > Network and Internet > Network and Sharing Center
6. Click on “Properties” of the wired network connection , then go to Sharing and click on "Allow other network users to connect". Make sure that the networking connection is changed to "Local Area Connection":
7. Based on all my research, the above steps should have helped me in connecting to the Raspberry Pi, but the connection still did not work. After some more researching I found out that SSH was disabled by default on newly installed Raspberry Pi. Hence, I  needed a monitor for the initial setup.
8. So, I had to connect a USB Keyboard and a TV monitor (using the HDMI port) on the Raspberry Pi and reboot  it. Once the Pi booted, I had to set some “first time use” parameters. Then ( as per the suggestions from the official documentation), I did the   the following.
9. *Launch Raspberry Pi Configuration from the Preferences menu*
10. *Navigate to the Interfaces tab*
11. *Select Enabled next to SSH*
12. *Click OK*
13. After this, I needed a method to know the IP address of the Raspberry Pi. I used a free tool called “Advanced IP Scanner” . Following steps were required to find the IP address using this tool.
14. First on the Windows laptop, from the “Command Prompt” run the following command

*>ipconfig*

1. Look at the local (laptop’s ) IP Address. Next in the “Advanced IP Scanner” run scan which has the local IP Address.
2. One of the IP address will be Raspberry Pi’s IP address.

I was able to connect the Raspberry Pi over SSH via a wired connection from my Windows laptop, using this IP Address.

1. I was able to check if the Raspberry Pi was able to get to the Internet. I connected to the Raspberry Pi using SSH ( Putty) . On the Raspberry Pi terminal, I executed the following command

*>wget google.com*

1. This command executed successfully.
2. Next, I had to connect to the Raspberry Pi via a GUI terminal. For this I decided to use “Remote Desktop Connection” from my Windows10 laptop. But, before I could use this tool from my laptop, I needed to enable RDP server on Raspberry Pi. Following steps were used to achieve this
   1. Used Putty (SSH) to log into the Raspberry Pi.
   2. Opened a “terminal” and ran the following command

*>sudo apt-get update*

*>sudo apt-get install xrdp*

* 1. Restart Pi

1. After the restart, I was able to use “Remote Desktop Connection” to have a GUI Terminal connected to Raspberry Pi
2. Next, I enabled the Wi-Fi connection on the Raspberry Pi. I connected to the same Wi-Fi network that my laptop was connected too. Now I no more required  the wired cable, I could connect from a “Remote Desktop Connection” over the Wi-Fi to the Raspberry Pi.
3. Last step was to use Ubuntu laptop to wirelessly connect to the Raspberry Pi. This was fairly simple. Instead of the “Advanced IP Scanner” i used “Angry IP Scanner” ,and for GUI I used “Remmima”.

## Sensors going offline

During my program run, very often the sensor would return no data.

1. After some troubleshooting, I figured out that connecting wire ends were loose connections to the breadboard were the principal cause for this problem.
2. However, after fixing that ( y firmly inserting the loose ends), some time the sensors would still go offline. Next option was to get the sensors firmly soldered.
3. The probability of this problem had drastically reduced. But it would still occur randomly – senor would just go offline. I figured out that SGP30 sensor was trying to read data using i2c interface. If I used the ‘i2cdetect’ command from a separate terminal while the python program was also running, both the programs would show that the sensor was offline. This meant that the i2C interface was not “threadsafe”.

# Future Research and Improvements

## Unfinished portions of the project

1. Supporting multiple Sensors in a single home
2. Real time graphs with user defined thresholds
3. Creating and implementing the “relay” control

## Future:

### Domestic uses:

1. Collecting indoor and near-outdoor (patio/garden/balcony) Air-Quality data
2. Providing the user analytics – “based on the air-quality pattern in the kitchen and time, I can predict that the time when you cook and I will automatically roll-up the windows and start the exhaust fans” OR “Your garage is the single biggest source of CO2, every weekday at 6 pm I see a peak in the CO2, probably the time you return from work and park your car”
3. Provide the ‘micro-level’ air quality report to the city authorities who can corelate the data with health data.
4. Compare how good or bad is your air-quality relative to others (confidential) in the neighborhood.

### Commercial uses:

1. Fitness center could monitor and regulate the indoor-air quality remotely across all their locations
2. School Districts could monitor and regulate the indoor-air quality of each area of schools – from class rooms, locker rooms, bathrooms, office , teachers toom, fitness rooms etc.
3. Wineries that require very accurate air-quality, could monitor and regulate it remotely

# Bibliography

**Air Quality related:**

United States Environmental Protection Agency : <https://www.epa.gov/indoor-air-quality-iaq/introduction-indoor-air-quality>

**Raspberry Pi & Sensors Related**

<https://www.raspberrypi.org/>

<https://www.raspberrypi.org/forums/>

<https://adafruit.com/>

<https://www.programcreek.com/python/example/92777/Adafruit_DHT.read_retry>

<https://thepihut.com/collections/sensors>

**Web Server and Database Related:**

Installing Node JS on Ubuntu <https://www.youtube.com/results?search_query=installing+node+js+in+ubuntu>

Simple  Node JS Application <https://www.codeproject.com/Articles/1239390/%2FArticles%2F1239390%2FCreating-Simple-API-in-Node-js>

Installing MySQL on Ubuntu 16.04 <https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-16-04>

Build A RESTful Api With Node.js And Express.Js

<https://medium.com/@purposenigeria/build-a-restful-api-with-node-js-and-express-js-d7e59c7a3dfb>