DESIGN CREDIT

April 2024

Sensor Interfacing for IoT Applications

Deepak Bhatter B22EE022
Rahul Sharma B22EE051

Supervisor- Dr. Saakshi Dhanekar

TABLE OF CONTENTS

Tip: Use links to go to a different page inside your presentation.

THE OVERVIEW

This project explored the exciting world of sensor interfacing for Internet of Things (IoT) applications. The focus was on connecting various sensors – both analog and digital – with Raspberry Pi and Arduino Uno boards. Through different connection methods, the project aimed to acquire sensor data, process it effectively, and ultimately deliver valuable insights.

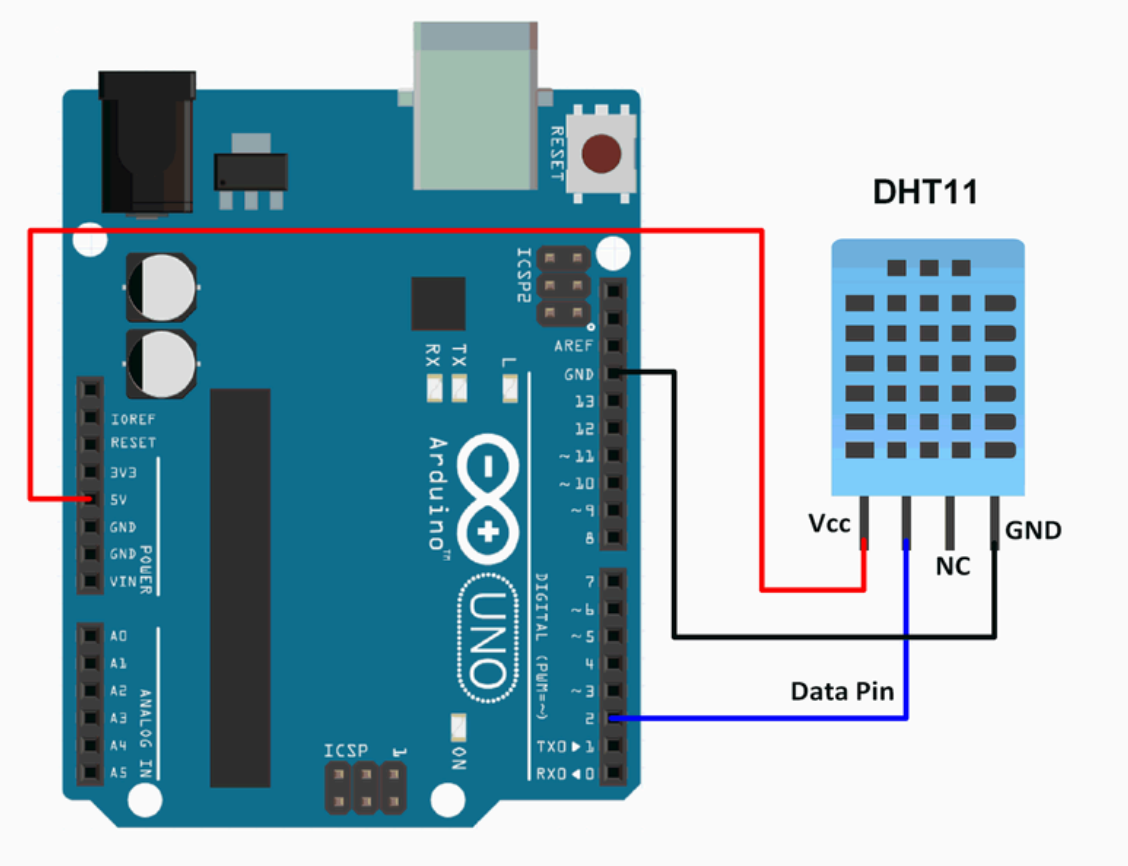
- **Sensor Integration**: A range of sensors were interfaced with the development boards. This included digital sensors like DHT11 (temperature and humidity) and analog gas sensors like MQ2, MQ3, and MQ135. Different connection strategies were employed, including direct connection to Raspberry Pi for digital sensors and the use of an MCP3008 analog-to-digital converter for analog sensors interfaced with Raspberry Pi. Additionally, an Arduino Uno was utilized as an intermediary for some analog sensors, bridging the gap between the sensor and the Raspberry Pi.
- **Data Acquisition and Processing**: Python scripts played a crucial role in this project. They facilitated communication with the sensors (either directly or via the Arduino), retrieved raw sensor data, and performed essential processing tasks. This could involve unit conversions, scaling of values, or filtering of data to ensure accuracy and clarity.
- **Data Visualization and Storage**: The processed sensor data was not just stored, but also transformed into informative plots. These plots, generated using Python libraries, provided a clear visual representation of sensor readings over time. This visualization allows for easy identification of trends, fluctuations, or potential anomalies in the sensor data. Additionally, the project implemented functionalities to store the collected data on the Raspberry Pi. This stored data becomes a valuable resource for future analysis, trend identification, or even potential use in machine learning applications.
- **Cloud Integration**: While not explicitly mentioned in every activity, the project aimed to explore the possibilities of sending sensor data to the cloud. This could involve utilizing cloud platforms like ThingSpeak to store and visualize data remotely, enabling access from anywhere with an internet connection.

ARDUINO UNO WITH DHT-11

Click here to view all the required files

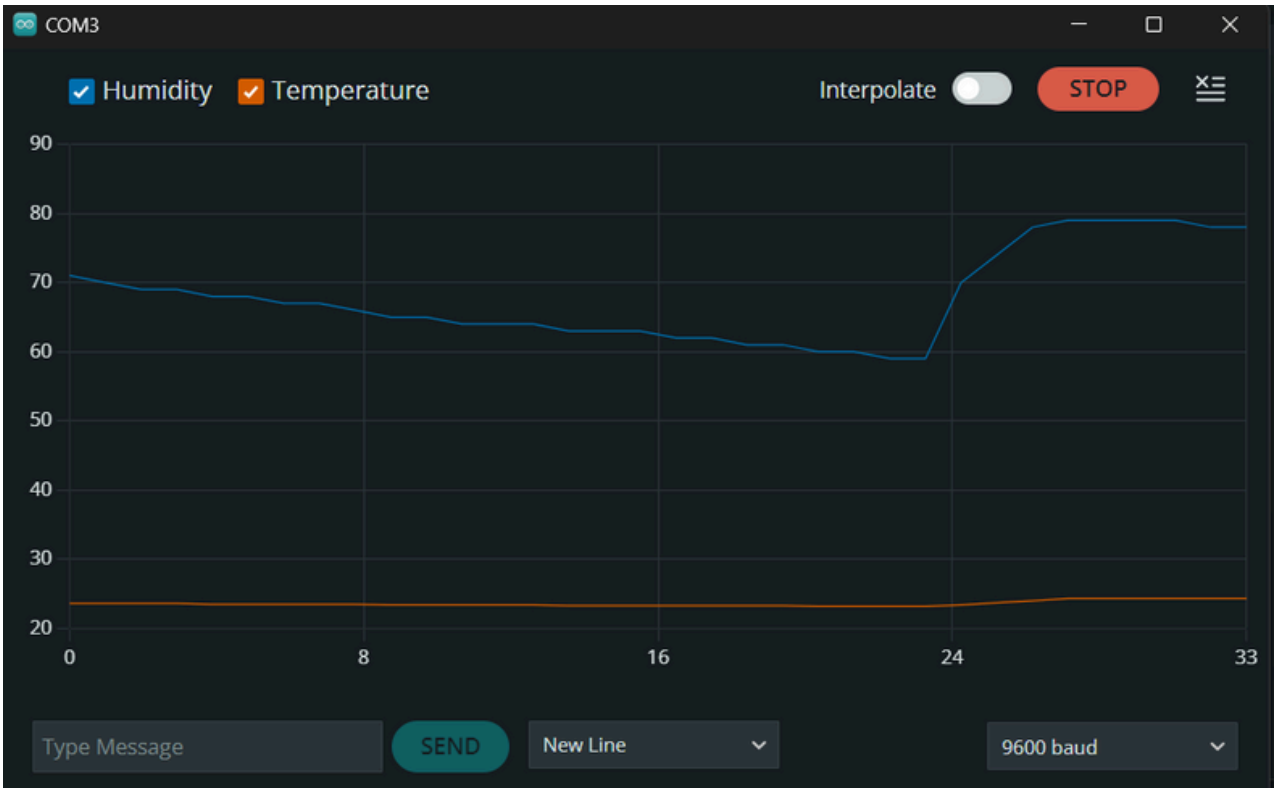
In the initial phase of the project, an Arduino Uno microcontroller board was employed to explore fundamental sensor interfacing concepts. A DHT11 sensor, capable of measuring both temperature and humidity, was directly connected to the Arduino Uno. We developed code specifically designed for the Arduino platform to interact with the DHT11 sensor. This code facilitated the retrieval of temperature and humidity readings from the sensor, allowing us to verify sensor functionality and gain experience with data acquisition from digital sensors.

CIRCUIT DIAGRAM



SERIAL PLOT AND OUTPUT

The Plot of temperature and humidity measured by the DHT-11 sensor is plotted by serial plotter of Arduino IDE which can be seen below:

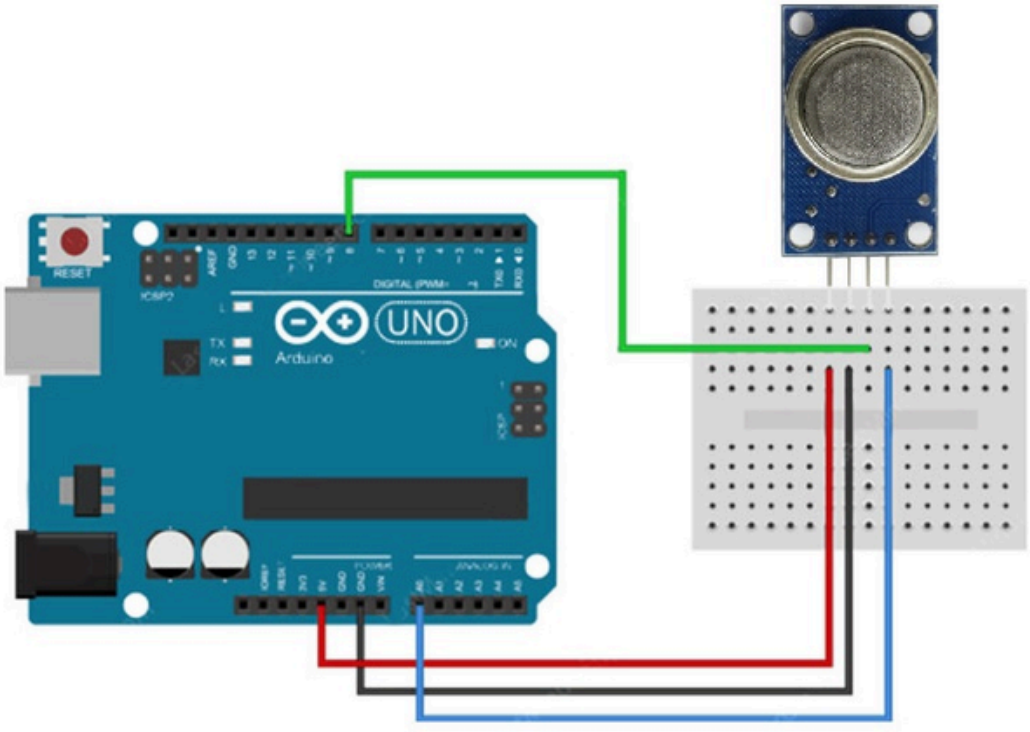


ARDUINO UNO WITH MQ-2

Click here to view all the required files

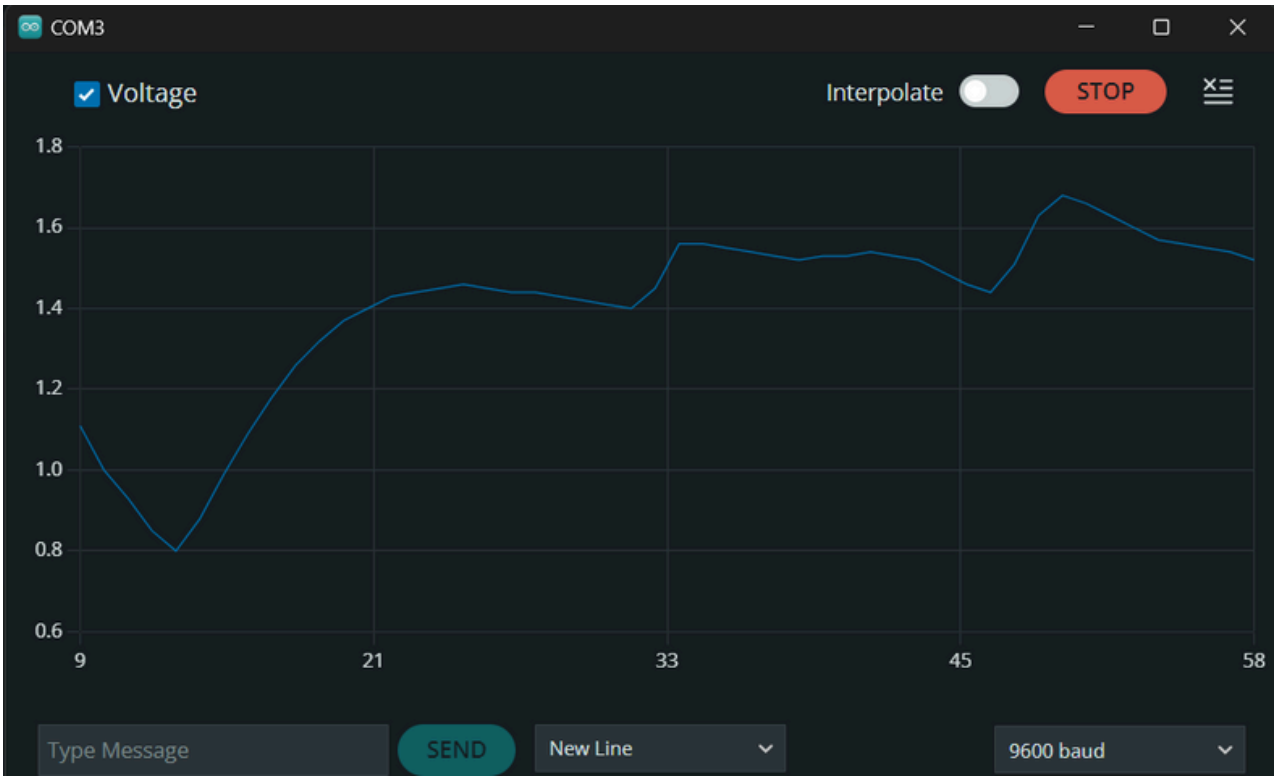
Building upon the initial exploration with digital sensors, the project investigated interfacing with analog sensors. An MQ2 sensor, known for its ability to detect the presence of combustible gases, was introduced. Code specific to the Arduino platform was developed to communicate with the MQ2 sensor. The MQ-2 sensor reads the data in 10 bits which is further converted to corresponding voltage by simple multiplication taking some assumptions. This setup allowed us to assess the functionality of the MQ2 sensor and gain experience with interfacing analog sensors

CIRCUIT DIAGRAM



SERIAL PLOT AND OUTPUT

The Plot of voltage measured by the MQ-2 sensor is plotted by serial plotter of Arduino IDE which can be seen below:

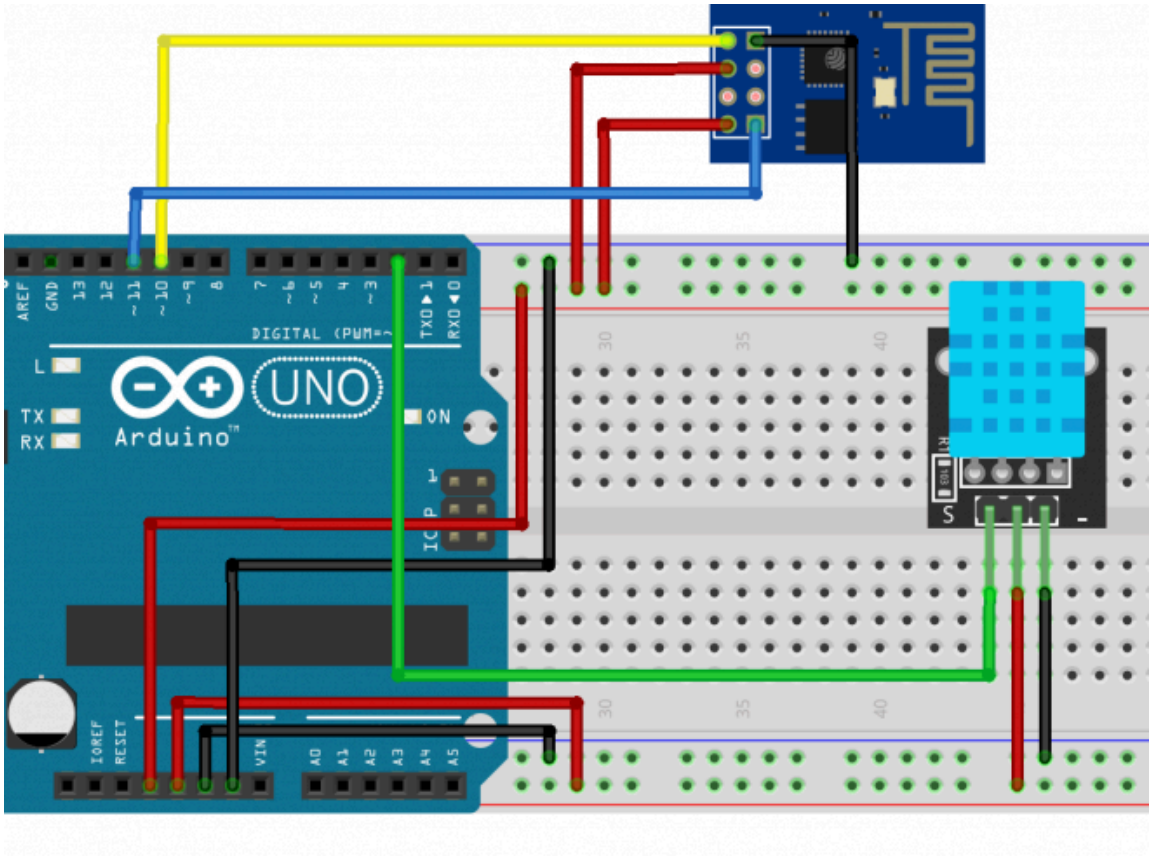


ARDUINO UNO WITH DHT-11 WITH WIFI MODULE

Click here to view all the required files

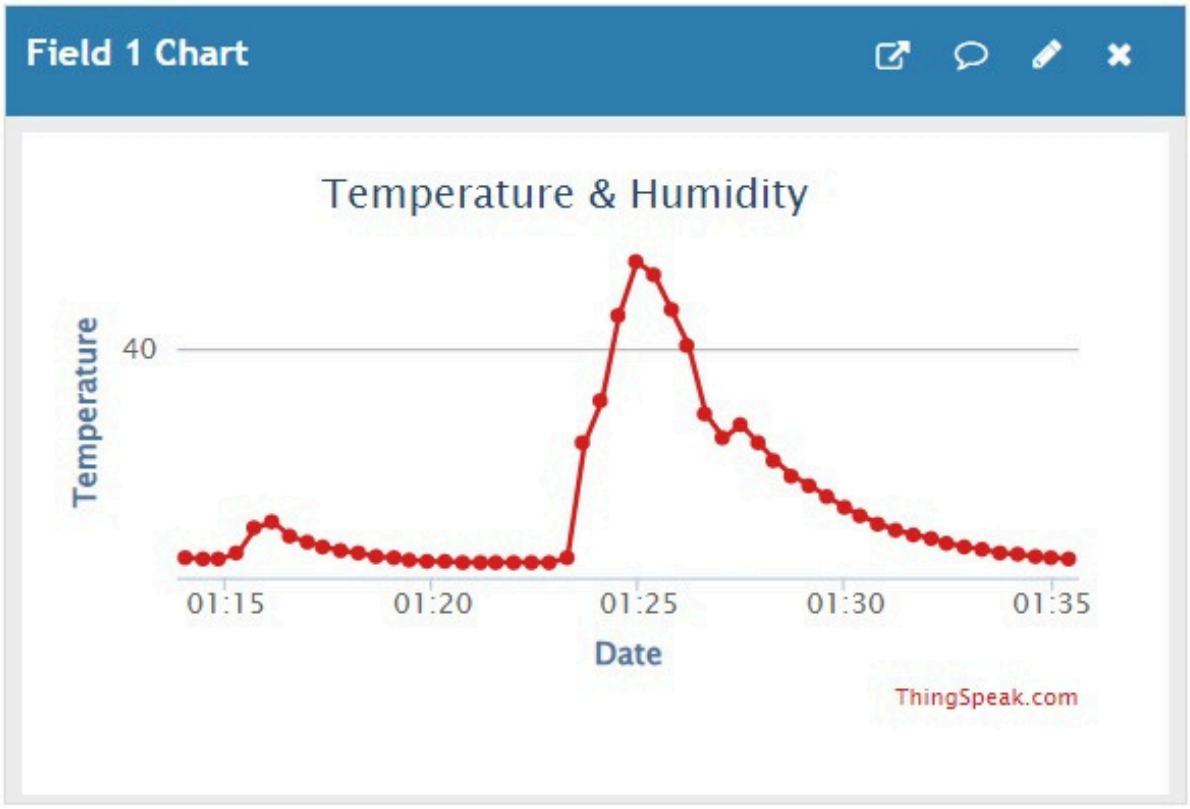
The project progressed towards wireless data transmission by incorporating an ESP8266 Wi-Fi module alongside the Arduino Uno and DHT11 sensor. The ESP8266, equipped with Wi-Fi capabilities, enabled the system to transmit the acquired temperature and humidity data from the DHT11 sensor wirelessly. This involved developing code for the Arduino Uno to not only interact with the DHT11 sensor but also to communicate and collaborate with the ESP8266 module. The ESP8266, programmed with its own firmware, facilitated the connection to a Wi-Fi network and the subsequent transmission of sensor data. This setup demonstrated the capability of sending sensor readings remotely, paving the way for integration with cloud platforms for data storage and visualization.

CIRCUIT DIAGRAM



THINGSPEAK TEMPERATURE PLOT

The Plot of temperature measured by the DHT-11 sensor is sent to cloud ThingSpeak and the following graph is plotted with time:



RASPBERRY PI GPIO

VOLTAGES

Two 5V pins and two 3.3V pins are present on the board, as well as a number of ground pins (GND), which can not be reconfigured. The remaining pins are all general-purpose 3.3V pins, meaning outputs are set to 3.3V and inputs are 3.3V-tolerant.

OUTPUTS

A GPIO pin designated as an output pin can be set to high (3.3V) or low (0V).

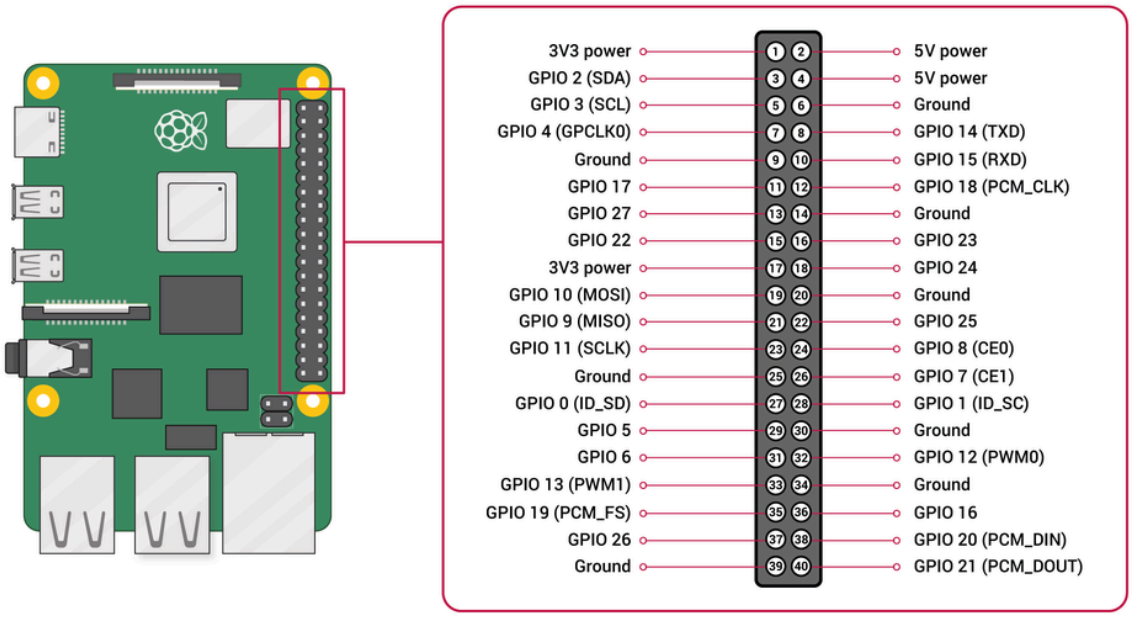
INPUTS

A GPIO pin designated as an input pin can be read as high (3.3V) or low (0V). This is made easier with the use of internal pull-up or pull-down resistors. Pins GPIO2 and GPIO3 have fixed pull-up resistors, but for other pins this can be configured in software.

OTHER FUNCTIONS

As well as simple input and output devices, the GPIO pins can be used with a variety of alternative functions, some are available on all pins, others on specific pins.

PWM (pulse-width modulation)	SPI	I2C	Serial
• Hardware PWM available on GPIO12, GPIO13, GPIO18, GPIO19	• SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7) • SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)	• Data: (GPIO2); Clock (GPIO3) • EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)	• TX (GPIO14); RX (GPIO15)

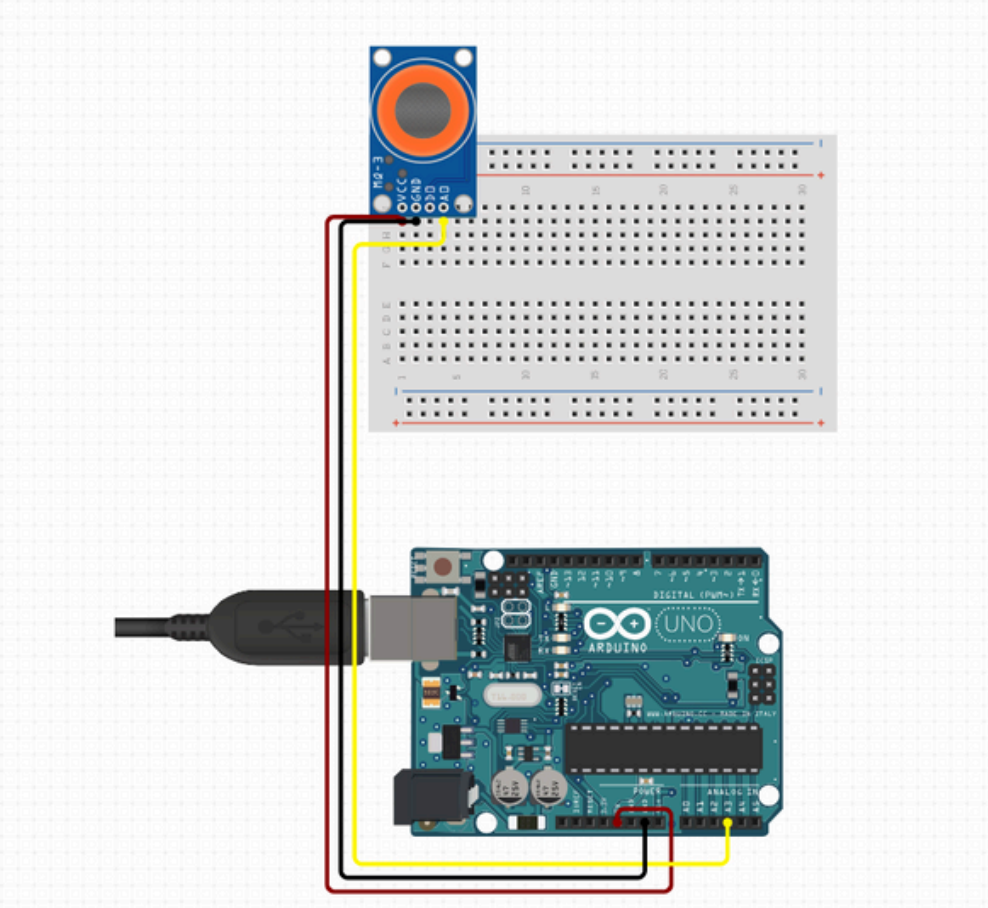


RASPBERRY PI WITH MQ-3 WITH ARDUINO UNO

Click here to view all the required files

This section investigated interfacing an MQ3 sensor, used for detecting combustible gases, directly with the Raspberry Pi. The MQ3 is an analog sensor, and while the Raspberry Pi itself doesn't have a built-in ADC, we employed an external Arduino Uno. This bridges the gap by converting the sensor's analog output into a digital value the Raspberry Pi can understand. A Python script running on the Raspberry Pi reads the serial input given by Arduino at the port it is connected, allowing it to acquire the digital sensor readings. The script then processed the data and generated informative plots to visualize the sensor readings over time using libraries like Matplotlib. Additionally, the script incorporated functionalities to store the collected data in a text file on the Raspberry Pi.

CIRCUIT DIAGRAM

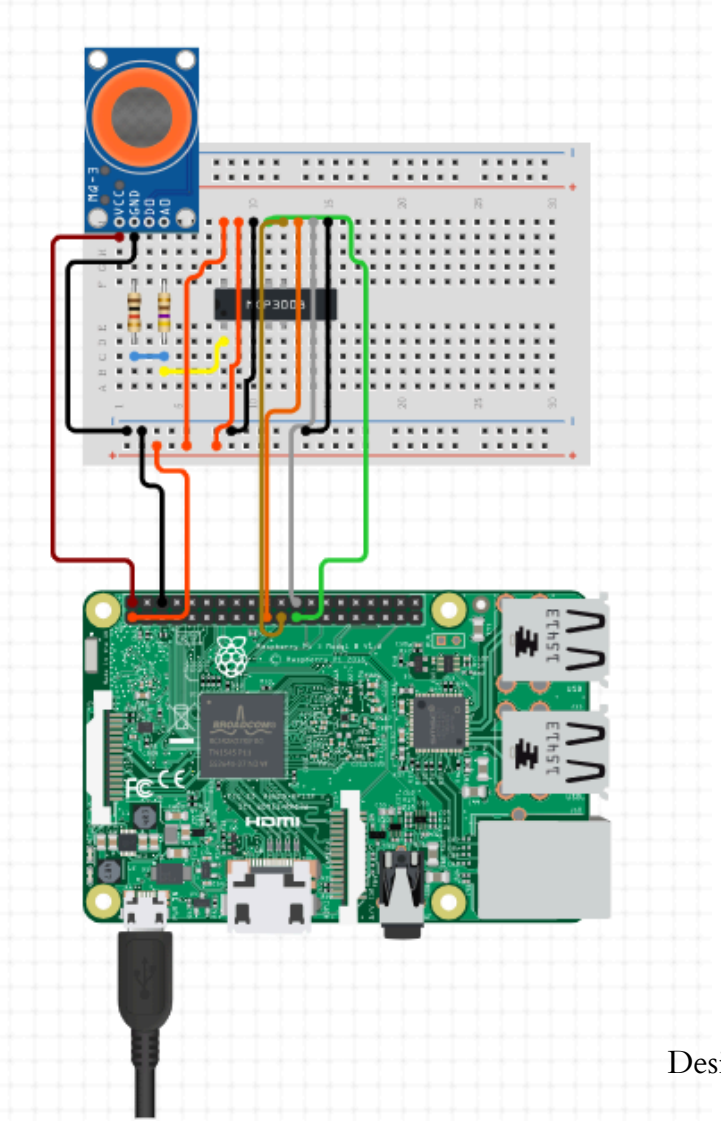


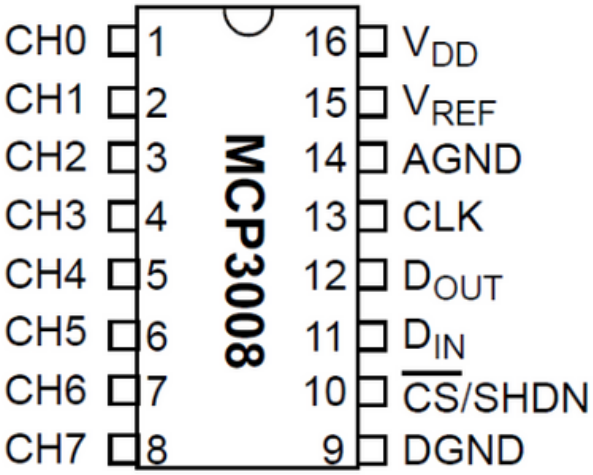
RASPBERRY PI WITH MQ-3 WITH MCP3008

Click here to view all the required files

This section explored interfacing an MQ3 sensor, used for detecting combustible gases, directly with the Raspberry Pi. Since the MQ3 is an analog sensor, the Raspberry Pi itself was utilized for analog-to-digital conversion (ADC) capabilities. We employed an MCP3008 ADC chip to convert the sensor's analog output into a digital value the Raspberry Pi could understand. A Python script running on the Raspberry Pi facilitated communication with the MCP3008, acquiring the digital sensor readings. The script then processed the data (potentially including scaling or filtering), and generated informative plots to visualize the sensor readings over time using libraries like Matplotlib.

CIRCUIT DIAGRAM

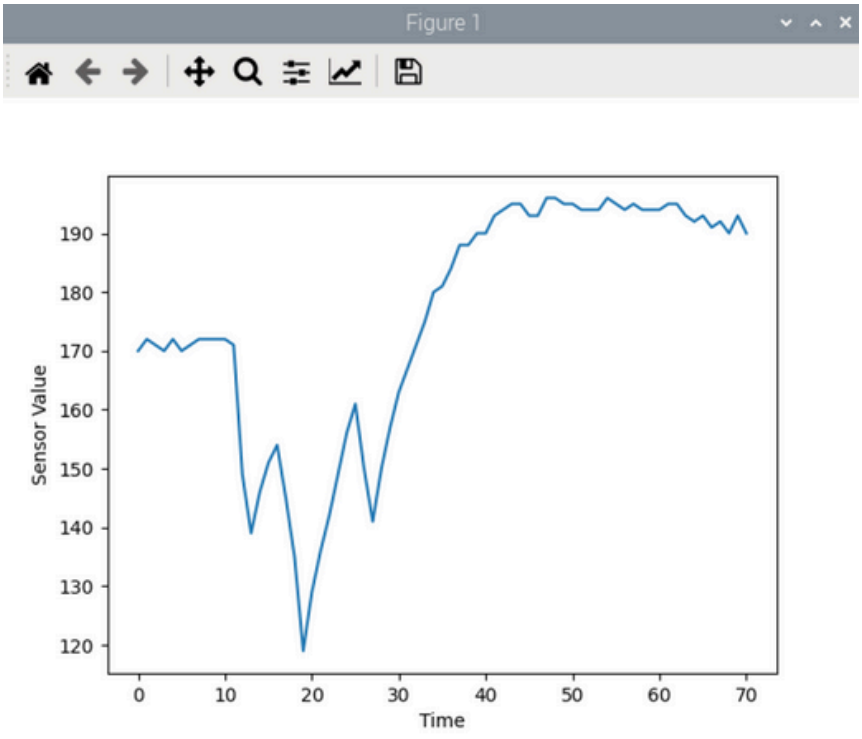




Wiring the MCP3008 to the Raspberry Pi as follows:

- MCP3008 VDD to Raspberry Pi 3.3V
- MCP3008 VREF to Raspberry Pi 3.3V
- MCP3008 AGND to Raspberry Pi GND
- MCP3008 DGND to Raspberry Pi GND
- MCP3008 CLK to Raspberry Pi SCLK
- MCP3008 DOUT to Raspberry Pi MISO
- MCP3008 DIN to Raspberry Pi MOSI
- MCP3008 CS/SHDN to Raspberry Pi CE0

DATA PLOT

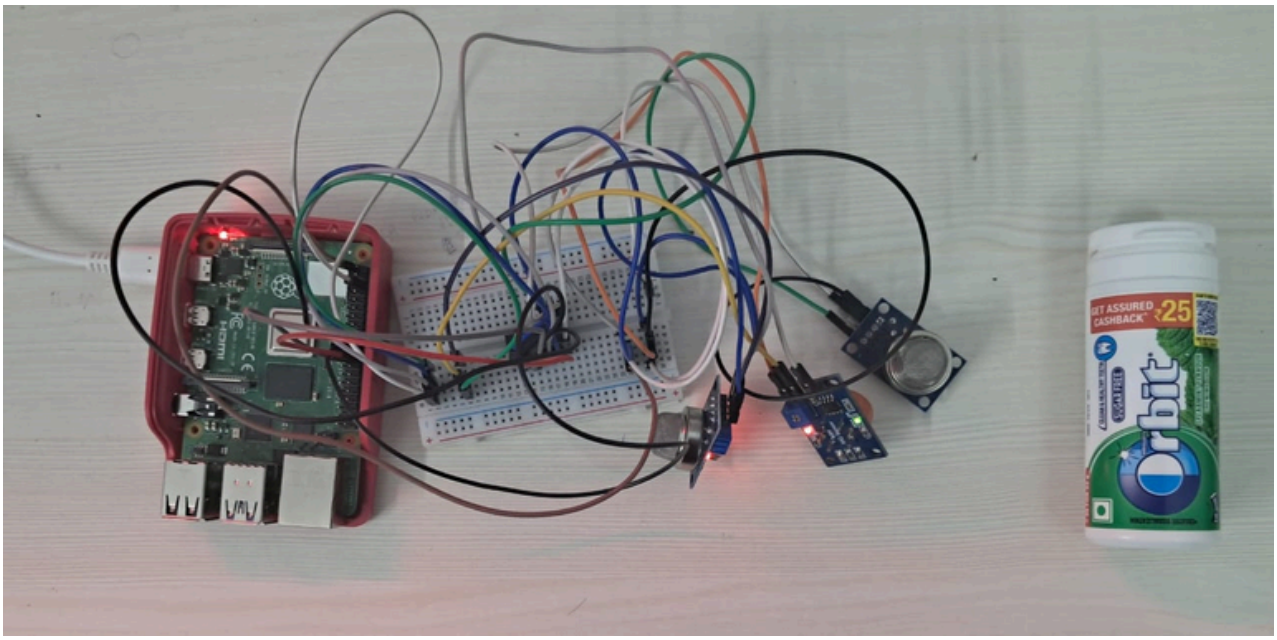


RASPBERRY PI WITH MULTIPLE SENSORS

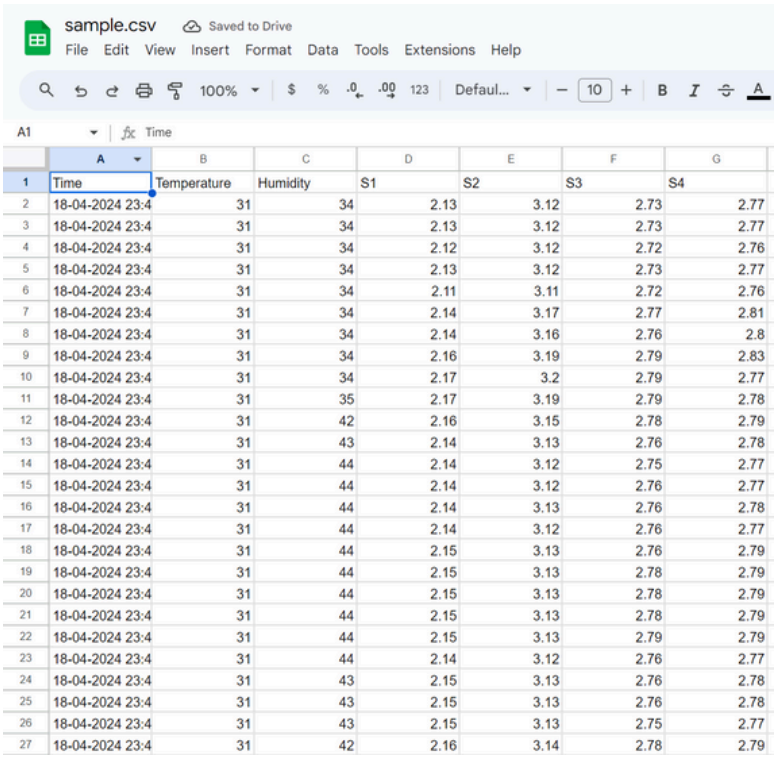
Click here to view all the required files

The final task, we were assigned were was the collection of data for array of sensors, which can collect data in real-time and store them in a csv file format. Later on, this data can be shared to a cloud platform for further analysis. Further, we connect the AO (analog pin) of all three sensors to the three channels of the MCP3008, namely pin 1, 2, 3. We connected the VCC of collective sensors to 3V3 on raspberry pi and ground to GND. The data collected on the drive under the name of sample.csv can be viewed as well.

CIRCUIT DIAGRAM

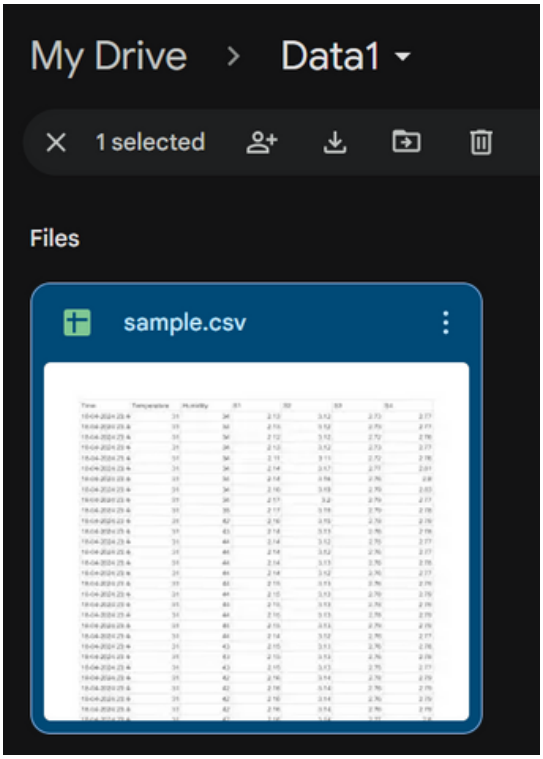


Note: To run this code, download the code.py file. Create a service account, and a folder on Google Drive, and share this folder with the email id from Service Account. Create a key as a json file, and rename it as service_account.json. In the code file, under the parent_folder variable, rename it as the key of the drive folder. Run the code and to stop the code, hit Ctrl + C, and BAM!!, you are done. Check your Google Drive folder, it should have the csv file.



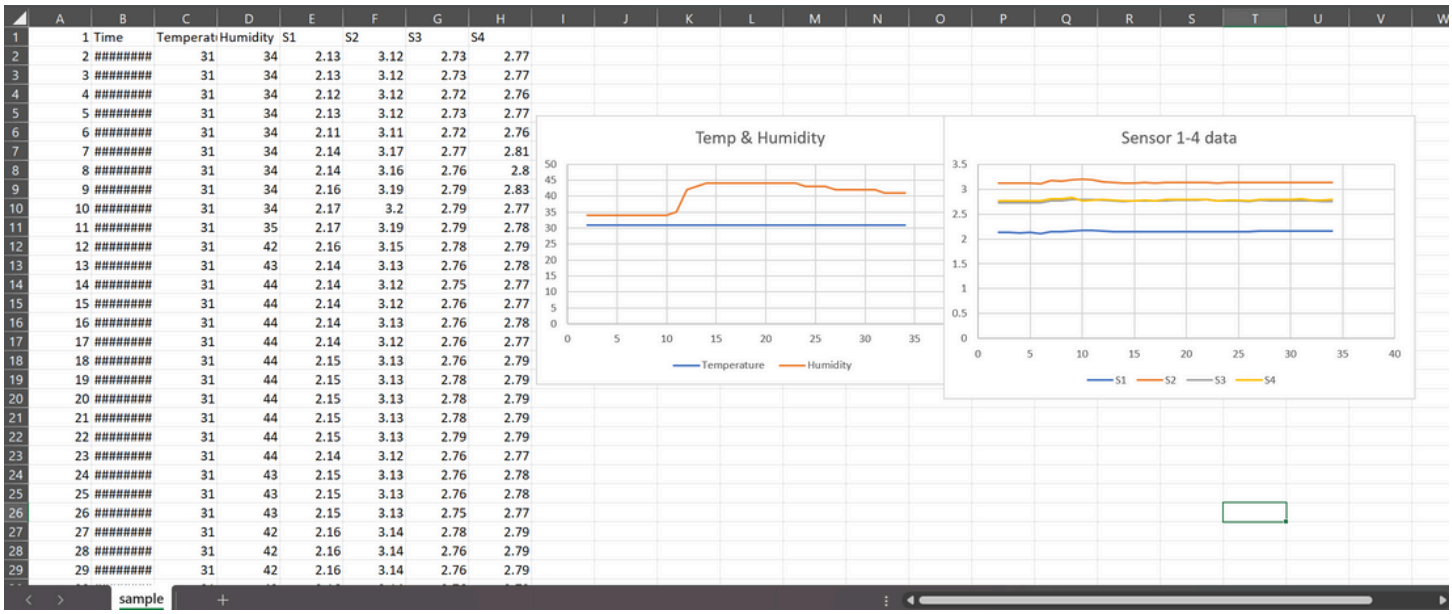
	A	B	C	D	E	F	G
1	Time	Temperature	Humidity	S1	S2	S3	S4
2	18-04-2024 23:4	31	34	2.13	3.12	2.73	2.77
3	18-04-2024 23:4	31	34	2.13	3.12	2.73	2.77
4	18-04-2024 23:4	31	34	2.12	3.12	2.72	2.76
5	18-04-2024 23:4	31	34	2.13	3.12	2.73	2.77
6	18-04-2024 23:4	31	34	2.11	3.11	2.72	2.76
7	18-04-2024 23:4	31	34	2.14	3.17	2.77	2.81
8	18-04-2024 23:4	31	34	2.14	3.16	2.76	2.8
9	18-04-2024 23:4	31	34	2.16	3.19	2.79	2.83
10	18-04-2024 23:4	31	34	2.17	3.2	2.79	2.77
11	18-04-2024 23:4	31	35	2.17	3.19	2.79	2.78
12	18-04-2024 23:4	31	42	2.16	3.15	2.78	2.79
13	18-04-2024 23:4	31	43	2.14	3.13	2.76	2.78
14	18-04-2024 23:4	31	44	2.14	3.12	2.75	2.77
15	18-04-2024 23:4	31	44	2.14	3.12	2.76	2.77
16	18-04-2024 23:4	31	44	2.14	3.13	2.76	2.78
17	18-04-2024 23:4	31	44	2.14	3.12	2.76	2.77
18	18-04-2024 23:4	31	44	2.15	3.13	2.76	2.79
19	18-04-2024 23:4	31	44	2.15	3.13	2.78	2.79
20	18-04-2024 23:4	31	44	2.15	3.13	2.78	2.79
21	18-04-2024 23:4	31	44	2.15	3.13	2.78	2.79
22	18-04-2024 23:4	31	44	2.15	3.13	2.79	2.79
23	18-04-2024 23:4	31	44	2.14	3.12	2.76	2.77
24	18-04-2024 23:4	31	43	2.15	3.13	2.76	2.78
25	18-04-2024 23:4	31	43	2.15	3.13	2.76	2.78
26	18-04-2024 23:4	31	43	2.15	3.13	2.75	2.77
27	18-04-2024 23:4	31	42	2.16	3.14	2.78	2.79

CSV File that can be viewed later on



CSV File received on the mail

Furthermore, the csv file can be downloaded and the data can be plotted again in excel file to get the same graph.



	A	B	C	D	E	F	G	H
1		1 Time	Temperati	Humidity	S1	S2	S3	S4
2		2 ########	31	34	2.13	3.12	2.73	2.77
3		3 ########	31	34	2.13	3.12	2.73	2.77
4		4 ########	31	34	2.12	3.12	2.72	2.76
5		5 ########	31	34	2.13	3.12	2.73	2.77
6		6 ########	31	34	2.11	3.11	2.72	2.76
7		7 ########	31	34	2.14	3.17	2.77	2.81
8		8 ########	31	34	2.14	3.16	2.76	2.8
9		9 ########	31	34	2.16	3.19	2.79	2.83
10		10 ########	31	34	2.17	3.2	2.79	2.77
11		11 ########	31	35	2.17	3.19	2.79	2.78
12		12 ########	31	42	2.16	3.15	2.78	2.79
13		13 ########	31	43	2.14	3.13	2.76	2.78
14		14 ########	31	44	2.14	3.12	2.75	2.77
15		15 ########	31	44	2.14	3.12	2.76	2.77
16		16 ########	31	44	2.14	3.13	2.76	2.78
17		17 ########	31	44	2.14	3.12	2.76	2.77
18		18 ########	31	44	2.15	3.13	2.76	2.79
19		19 ########	31	44	2.15	3.13	2.78	2.79
20		20 ########	31	44	2.15	3.13	2.78	2.79
21		21 ########	31	44	2.15	3.13	2.78	2.79
22		22 ########	31	44	2.15	3.13	2.79	2.79
23		23 ########	31	44	2.14	3.12	2.76	2.77
24		24 ########	31	43	2.15	3.13	2.76	2.78
25		25 ########	31	43	2.15	3.13	2.76	2.78
26		26 ########	31	43	2.15	3.13	2.75	2.77
27		27 ########	31	42	2.16	3.14	2.78	2.79
28		28 ########	31	42	2.16	3.14	2.76	2.79
29		29 ########	31	42	2.16	3.14	2.76	2.79