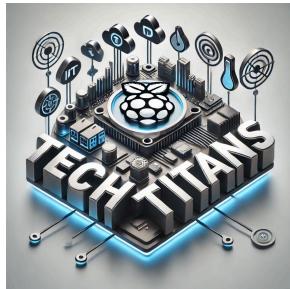


Bharat Dome Innovation Pvt. Ltd.

Internship 2024-25



Raspberry Pi based Weather Station

By TECH TITANS

Suhani Shetty, Keerthana AV, Pragna KP (ECE)

Bindu Madhavi V, Rakshitha R, G Dharmika (CSE)

Under the guidance of

Sathish B S (CEO)

Dr. P G Diwakar (ISRO Chair Professor)

Acknowledgement

We express our sincere gratitude to Sathish BS sir for providing such a wonderful opportunity.

We thank Dr. PG Diwakar sir for extending his constant support throughout the internship and giving us a chance to present our idea at NIAS.

Lastly, we would like to extend our heartfelt thanks to our HODs, Dr.Bino Prince Raja D, Dr. Bharath Vinjamuri, Dr. Kumaraswamy S, and Dr. HS Manjunatha Reddy at Global Academy of Technology for their unwavering support during our internship.

Table of Contents

Abstract	4
Introduction	5
Nature of work carried out	6
Results	11
Conclusion	15
References	16

Abstract

This project presents the design and implementation of a weather station using a Raspberry Pi. The weather station collects real-time data on various weather parameters, including temperature, humidity, atmospheric pressure, wind speed, wind direction, and rainfall, pm2.5 and pm10. The collected data is then processed and displayed for remote monitoring and analysis. The system utilizes a combination of sensors, including a BME280, an anemometer, wind vane, rain gauge and a laser PM2.5 SDS018 for sensing particulate matter. The Raspberry Pi controls the data acquisition process, processes and transmits the data using Wi-Fi connectivity. The web page provides a user-friendly interface for visualizing the collected data in real-time and historical trends. The project aims to provide a cost-effective and versatile solution for weather monitoring, enabling users to make informed decisions based on weather conditions.

Introduction

Weather stations have been instrumental in meteorological research and forecasting for centuries. Since our college is on an elevated platform and lacks a weather station that can monitor weather conditions we took up this problem statement in order to develop a feasible solution.

This project presents the design and implementation of a weather station built around a Raspberry Pi, a versatile single-board computer. The Raspberry Pi, coupled with a variety of sensors, allows for the collection of real-time weather data. The collected data is then processed and displayed on a webpage for analysis and visualization. This project will discuss the components used, the data acquisition and processing techniques, and the web-based interface for monitoring.

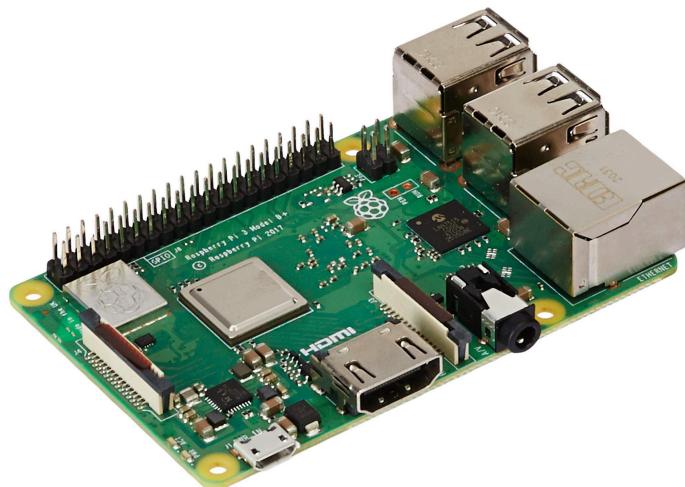


Fig. 1 Raspberry Pi 3 Model b+

Nature of Work Carried out

We have divided the entire project into 7 major tasks and below is the breakdown of each task.

Task 1 - Sensor selection and hardware setup

- Choose appropriate sensors for the weather station and assemble them on a breadboard.
- Research and source the required sensors (BME280 for temperature, humidity, pressure, anemometer and wind vane for wind speed/direction; rain gauge for rainfall).
- Ensure accurate data collection.
- Calibrate BME280 for temperature, humidity, and pressure.
- Calibrate wind sensors and rain gauge (debounce circuitry).



Fig. 2 BME280

Task 2 - Sensor integration and enclosures

- Build robust, outdoor-ready enclosures.
- Design weatherproof enclosures with ventilation for BME280.
- Secure sensor cables with waterproof glands.
- Test sensors in different environmental conditions.

Working mechanism of Rain Gauge

- The rain gauge uses a tipping bucket mechanism with a Hall sensor.
- When 2.5 ml of rainwater fills the bucket, it tips, triggering the sensor to record the event as a fixed rainfall amount.
- Paired with an anemometer, this setup provides real-time data on rainfall and wind speed for accurate weather tracking.

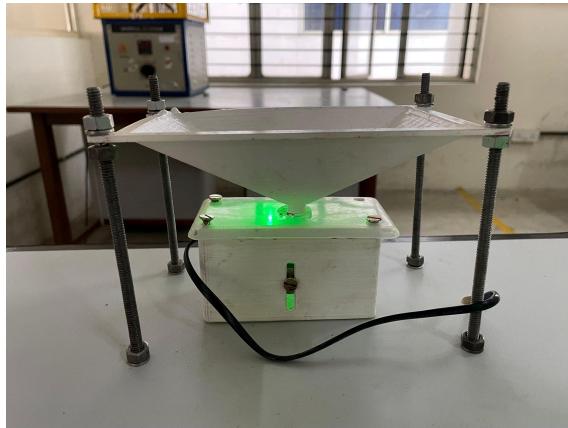


Fig. 3 Rain gauge

Working mechanism of Wind Vane system for direction detection

- Wind Vane Body: 3D-printed design with a tail fin to ensure directional stability.
- Hall Effect Sensors (4): Used for magnetic field detection to determine the orientation of the wind vane. Each sensor detects the magnetic field from a magnet aligned with the vane.
- Detection Capability: The system can determine up to 8 distinct directions (N, NE, E, SE, S, SW, W, NW).
- The wind vane aligns itself with the wind direction.
- Hall Effect Sensors: Positioned around the vane, each sensor reacts when aligned with a magnetic field, producing unique signals for each direction.
- Direction Encoding: Based on sensor signals, the Raspberry Pi interprets and maps the vane position to one of the 8 cardinal and intermediate directions.

- Raspberry Pi processes these signals in real-time.



Fig. 4 Wind vane

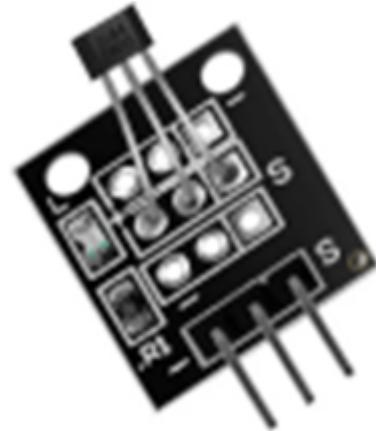


Fig. 5 Hall effect sensor

Working mechanism of Anemometer

- Magnetic Detection: Anemometer's rotating disc has a magnet on one arm. A Hall effect sensor detects each rotation as the magnet passes by.
- Pulse Generation: Each pass generates a pulse, counted by the Raspberry Pi's GPIO.
- Wind Speed Calculation: Python script on the Raspberry Pi calculates wind speed based on pulse frequency and a calibration factor.



Fig. 6 Anemometer

Working mechanism of Particulate Matter Sensor

- The SDS018 - Laser PM2.5 Sensor SDS018 V2 Air Quality Sensor uses principle of laser scattering to get the particle concentration between 0.3 to 10um in the air.
- The interface is digital UART output and built-in fan is stable and reliable.
- For our purposes, we are focusing on PM2.5 and PM10 measurements.



Fig 7. Laser PM2.5 Sensor SDS018

Task 3 - Communication Interface Design

- Ensure reliable sensor communication.
- Implement I2C (BME280)
- Test circuits for rain gauge/anemometer (pulse signals).
- Add error-checking and fail-safes for sensor data.

Task 4 - Data transmission and power management

- Optimize power and data transmission.
- Apply power-saving strategies (disable unused components).
- Set up WiFi for data transmission.
- Test power stability (battery/solar backup options).

Task 5 - Data collection and sensor-software integration

- Python scripts to capture data from the sensors
- Implement data filtering algorithms
- Format the data as required.

Task 6 - Data logging

- Modify the sensor data into a CSV format.

Task 7 - Data visualisation

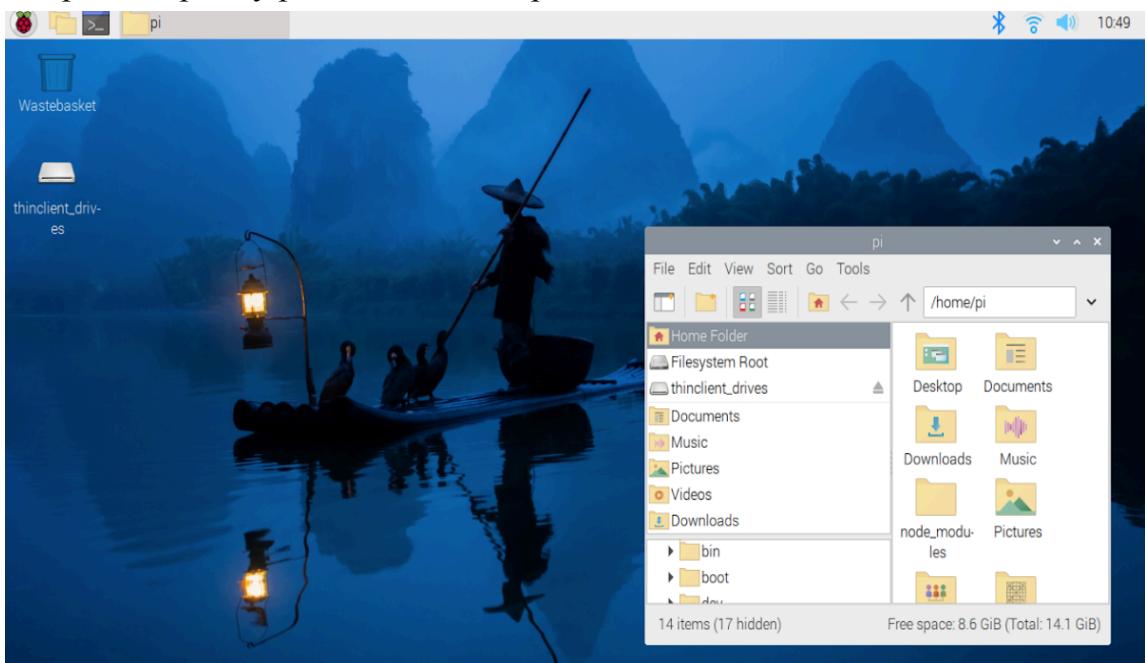
- Develop a UI using web technologies like HTML, CSS and Flask
- Use matplotlib for data visualization

Results

1. Assembly of components



2. Setup the raspberry pi OS with the help of RealVNC Viewer.



3. A glimpse of the mainscript to capture sensor readings.

```

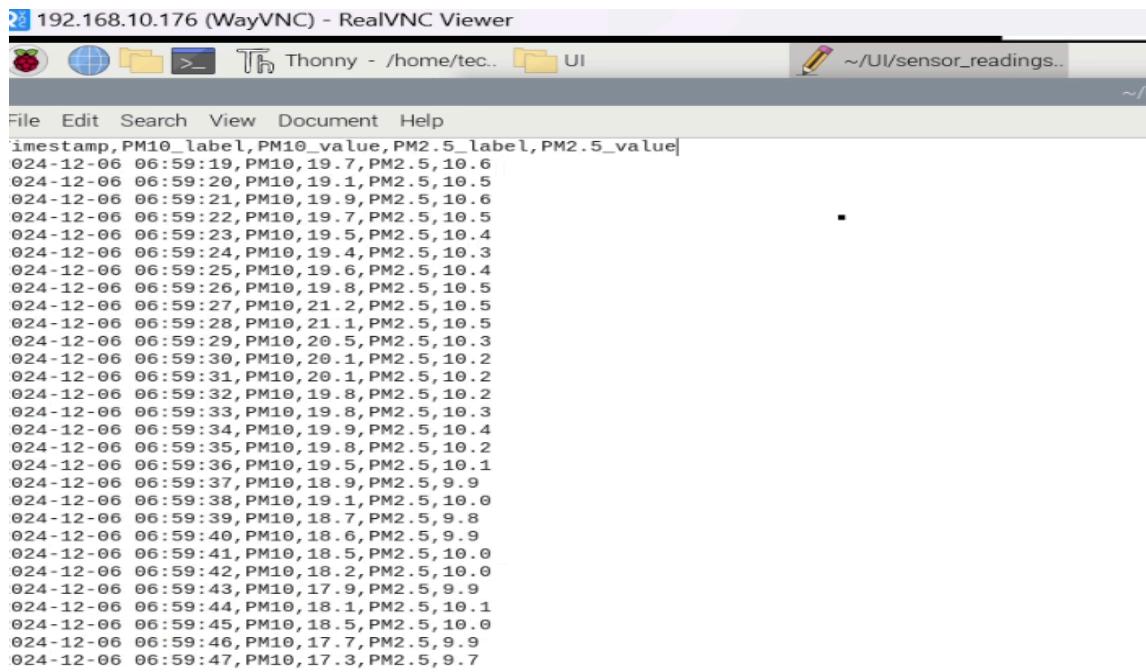
1 import threading
2 import time
3 import csv
4 import os
5 import RPi.GPIO as GPIO
6 from datetime import datetime, timedelta
7 import smbus2
8 import bme288
9 from serial import Serial, SerialException
10 import logging
11
12 # Air Quality (Nova SDS018) Setup
13 DEFAULT_SERIAL_PORT = "/dev/ttyUSB0" # Serial port to use if no other specified
14 DEFAULT_BAUD_RATE = 9600 # Serial baud rate to use if no other specified
15 DEFAULT_SERIAL_TIMEOUT = 1 # Serial timeout to use if not specified
16 DEFAULT_READ_TIMEOUT = 1 # How long to sit looking for the correct character sequence.
17 DEFAULT_LOGGING_LEVEL = logging.DEBUG # Set to DEBUG for detailed logs
18
19 MSG_CHAR_1 = b'\x0A' # First character to be received in a valid packet
20 MSG_CHAR_2 = b'\x09'
21
22 # CSV File Setup
23 csv_file = "weather_data.csv"
24 if not os.path.exists(csv_file):
25     with open(csv_file, "w", newline="") as file:
26         writer=csv.writer(file)
27         writer.writerow([ "Timestamp", "Windspeed", "Rainfall", "Winddirection", "Temperature", "Pressure", "Humidity"])
28
29 # Cleanup any previous configurations
30 GPIO.cleanup()
31
Shell
2024-12-06 12:29:30,438 - SDS018 Interface - 188 - DEBUG - 225
2024-12-06 12:29:31,432 - SDS018 Interface - 188 - DEBUG - 225
Pulse count: 0
Pulse count: 2
Pulse count: 3
2024-12-06 12:29:32,434 - SDS018 Interface - 188 - DEBUG - 222
Pulse count: 4
Pulse count: 5
2024-12-06 12:29:33,437 - SDS018 Interface - 188 - DEBUG - 223
Pulse count: 6
2024-12-06 12:29:34,446 - SDS018 Interface - 188 - DEBUG - 225

```

Local Python 3 • /usr/bin/python3

4. CSV files that recorded all the sensor readings

Timestamp	Windspeed	Rainfall	Winddirection	Temperature	Pressure	Humidity
2024-12-06 12:29:30.172995	9.71999	0	South East	0, 0, 0		
2024-12-06 12:29:30.173656	9.71999	0	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.174431	9.71999	0	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.175028	9.71999	0	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.175789	9.71999	2.5	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.176518	9.71999	2.5	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.177123	9.71999	5.0	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.177713	9.71999	5.0	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.178334	9.71999	7.5	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.178906	9.71999	7.5	South East	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.179828	9.71999	10.0	South West	25.156334507861175	919.5083883138301	59.961394984259144
2024-12-06 12:29:30.180542	16.2	10.0	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.181103	16.2	12.5	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.181771	16.2	12.5	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.182454	16.2	15.0	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.183073	16.2	15.0	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.183983	16.2	17.5	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.184887	16.2	17.5	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.185561	16.2	20.0	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.186195	16.2	22.5	South West	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.187312	16.2	22.5	South	25.171719652047614	919.5589439170632	59.79221891584619
2024-12-06 12:29:30.187955	16.2	25.0	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.188541	16.2	25.0	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.189138	16.2	27.5	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.189785	16.2	30.0	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.190421	16.2	32.5	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.191785	16.2	35.0	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.192553	16.2	35.0	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.193117	16.2	37.5	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.193762	16.2	37.5	South	25.207618332002312	919.5021030360756	59.11422029870344
2024-12-06 12:29:30.194541	16.2	42.5	South	25.207618332002312	919.5021030360756	59.11422029870344



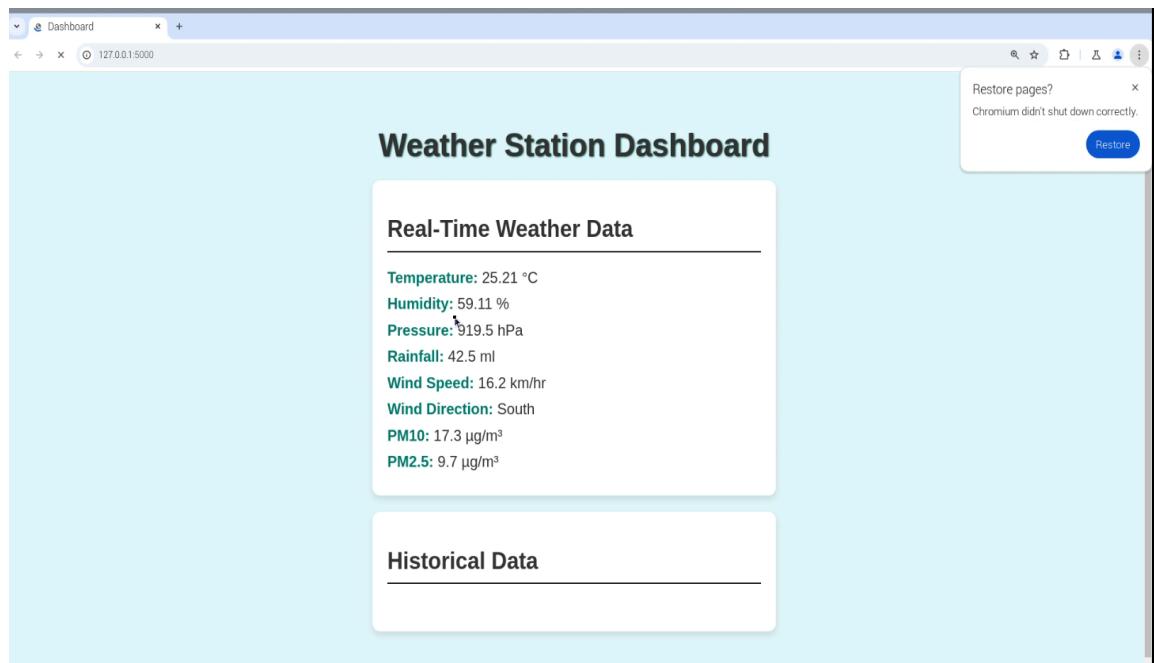
192.168.10.176 (WayVNC) - RealVNC Viewer

Thonny - /home/tec.. UI

File Edit Search View Document Help

```
timestamp,PM10_label,PM10_value,PM2.5_label,PM2.5_value
024-12-06 06:59:19,PM10,19.7,PM2.5,10.6
024-12-06 06:59:20,PM10,19.1,PM2.5,10.5
024-12-06 06:59:21,PM10,19.9,PM2.5,10.6
024-12-06 06:59:22,PM10,19.7,PM2.5,10.5
024-12-06 06:59:23,PM10,19.5,PM2.5,10.4
024-12-06 06:59:24,PM10,19.4,PM2.5,10.3
024-12-06 06:59:25,PM10,19.6,PM2.5,10.4
024-12-06 06:59:26,PM10,19.8,PM2.5,10.5
024-12-06 06:59:27,PM10,21.2,PM2.5,10.5
024-12-06 06:59:28,PM10,21.1,PM2.5,10.5
024-12-06 06:59:29,PM10,20.5,PM2.5,10.3
024-12-06 06:59:30,PM10,20.1,PM2.5,10.2
024-12-06 06:59:31,PM10,20.1,PM2.5,10.2
024-12-06 06:59:32,PM10,19.8,PM2.5,10.2
024-12-06 06:59:33,PM10,19.8,PM2.5,10.3
024-12-06 06:59:34,PM10,19.9,PM2.5,10.4
024-12-06 06:59:35,PM10,19.8,PM2.5,10.2
024-12-06 06:59:36,PM10,19.5,PM2.5,10.1
024-12-06 06:59:37,PM10,18.9,PM2.5,9.9
024-12-06 06:59:38,PM10,19.1,PM2.5,10.0
024-12-06 06:59:39,PM10,18.7,PM2.5,9.8
024-12-06 06:59:40,PM10,18.6,PM2.5,9.9
024-12-06 06:59:41,PM10,18.5,PM2.5,10.0
024-12-06 06:59:42,PM10,18.2,PM2.5,10.0
024-12-06 06:59:43,PM10,17.9,PM2.5,9.9
024-12-06 06:59:44,PM10,18.1,PM2.5,10.1
024-12-06 06:59:45,PM10,18.5,PM2.5,10.0
024-12-06 06:59:46,PM10,17.7,PM2.5,9.9
024-12-06 06:59:47,PM10,17.3,PM2.5,9.7
```

5. Dashboard that displays the latest data collected



6. Visualization using matplotlib



Conclusion

The developed Raspberry Pi-based weather station successfully collects, processes, and transmits real-time weather data. The web-based interface enables remote monitoring and analysis, making it a valuable tool for various applications. Future improvements could include integrating additional sensors such as solar radiation sensors. Additionally, exploring machine learning techniques for predictive weather forecasting would enhance the system's capabilities. By continuing to develop and refine this technology, we can gain deeper insights into weather patterns and make more informed decisions in various fields.

References

[1] Build your own Weather Station

<https://projects.raspberrypi.org/en/projects/build-your-own-weather-station/0>

[2] Pi Documentation

[Getting started - Raspberry Pi Documentation](#)

[3] RJ11 Breakout 6-Pin Connector SparkFun

<https://evelta.com/bob-14021-rj11-breakout-6-pin-connector-sparkfun/>

[4] Laser PM2.5 Sensor SDS018 V2 Air Quality Sensor Module UART Output

<https://evelta.com/laser-pm2-5-sensor-sds018-v2/>