Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

INTERNET OF THINGS (IoT)

Project Documentation on

“Temperature and Humidity Monitor System”

By: SHARANYA ADIGA(585849)

Date: 25.01.2023

To: Prof. Dr. Alexander Huhn
(Alexander.Huhn@HTW-Berlin.de)

Table of contents

1. Introduction 3
    1.1 Purpose 3
    1.2 Project Scope 3
2. Component Overview 4
    2.1 Project Components 4
3. Hardware - Sensors, Microcontrollers etc. 5
    3.1 Raspberry Pi 5
    3.2 ESP8266 and DHT22 5
    3.3 Other Hardware & Equipment 7
4. Software & Programming 7
    4.1 Libraries & Output 7
    4.2 Node RED with InfluxDb Library 9
    4.3 Influx-Out Node Configuration 9
5. Data Storage & Visualization 10
    5.1 InfluxDb - Time Series Database 10
    5.2 Using Node RED dashboard 11
    5.3 Metrics Monitored 10
    5.4 Telegram Bot 12
    5.5 Visualization of other cities's temperature and humidity 12
6. Conclusion & Future Possibilities 14

References 15

1. Introduction

The following section provides an overview of the application of a Temperature and Humidity Monitor for the subject Internet of Things (IOT) and its Application. To begin with, the purpose of the document is presented and its intended implementation is outlined in detail. Subsequently, the scope of the project specified by the document is given with a particular focus on what the project will do and the relevant benefits associated with it. The nomenclature used in this document is provided for convenience.

1.1 Purpose

The purpose of this document is to outline both the business and functional requirements of the Temperature and Humidity Monitor System. In addition to said requirements, the document also provides a detailed profile of hardware, software, and technology used along with circuit design laid upon the implementation. It is the intention that the presented set of requirements possesses the following qualities; correctness, unambiguousness, completeness, consistency, verifiability, modifiability and traceability. Consequently, the document should act as a foundation for efficient and well-managed project completion and further serve as an accurate reference in the future.

1.2 Project Scope

The intended purpose of the Temperature and Humidity Monitor will provide a golden platform for domestic (household) users who can use such a system to constantly get statistical reports of the Temperature and Humidity around their living spaces, and can control the comfort level of their indoor environments. Temperature controls are used to regulate the warmth or coolness of a room, while humidity controls are used to adjust the amount of moisture in the air.

Additionally, temperature and humidity are also important factors in industries such as agriculture, food storage, and pharmaceuticals, where the correct conditions must be maintained to ensure the quality and safety of products.

2. Component Overview

2.1 Project Components

As we can see in the diagram attached below, the system consists of a temperature humidity sensor connected together with a ESP8266 microcontroller in harmony with the Raspberry Pi in order to be able to collect, process, store and visualize the required data.

The Raspberry Pi here has Micro-USB power input (5V @ 2A) connected to a power supply in order to draw its power for fulfilling its functional motives. Sensor is further connected with the ESP8266 microcontroller which helps collect all the data for our Temperature and Humidity monitor system which is then connected to Raspberry Pi. These are connected via jumper wires to the relevant pins of the ESP8266.

The Raspberry Pi is connected to the required external keyboard, mouse and ethernet.

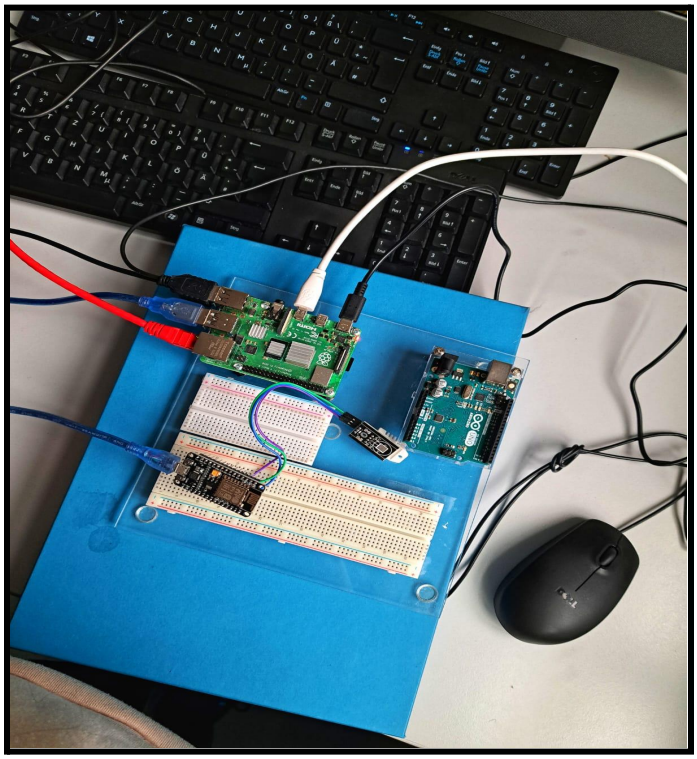


Figure 1.0 Setup

3. Hardware – Sensors, Microcontrollers etc

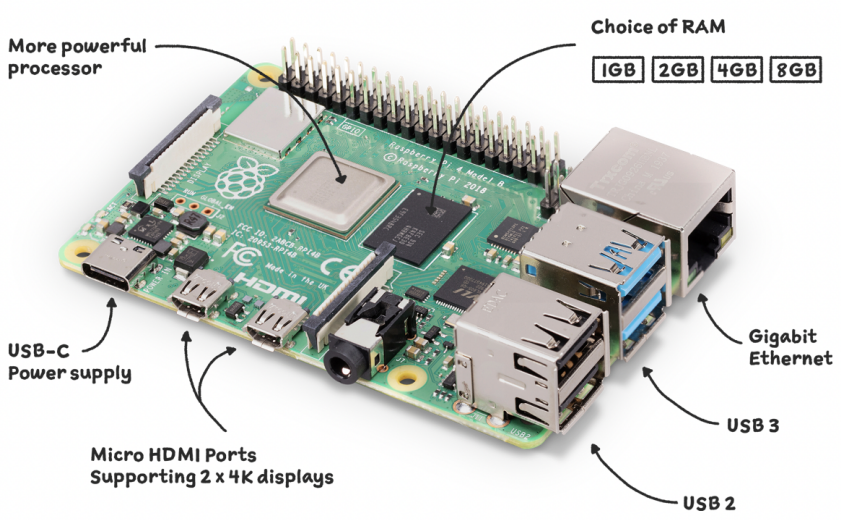


Figure 2.0 Raspberry Pi

3.1 Raspberry Pi

The Raspberry Pi is a small, low-cost computer developed by the Raspberry Pi Foundation in the United Kingdom.It can be used for a variety of projects, such as media centers, game consoles, and home automation systems. It runs on a variety of operating systems, including Linux and Windows 10 IoT Core, and can be programmed using a variety of languages, including Python, C, and Scratch.

Here we have used Raspberry pi 4 Model B. The Raspberry Pi 4 Model B was released in June 2019 with a 1.5 GHz 64-bit quad core ARM Cortex-A72 processor, on-board 802.11ac Wi-Fi, Bluetooth 5, full gigabit Ethernet (throughput not limited), two USB 2.0 ports, two USB 3.0 ports, 1–8 GB of RAM, and dual-monitor support via a pair of micro HDMI (HDMI Type D) ports for up to 4K resolution.

3.2 ESP8266 AND DHT22

ESP8266:

The ESP8266 is a low-cost microcontroller with built-in WiFi capabilities, developed by Espressif Systems. It can be used to add WiFi functionality to devices that would otherwise not have it, such as lamps, thermostats, or other small appliances. It can be programmed using the Arduino IDE, Lua, or MicroPython. The ESP8266 is a very popular choice for IoT projects due to its small size, low cost, and built-in WiFi capabilities. It also has a large community and many open-source libraries available.

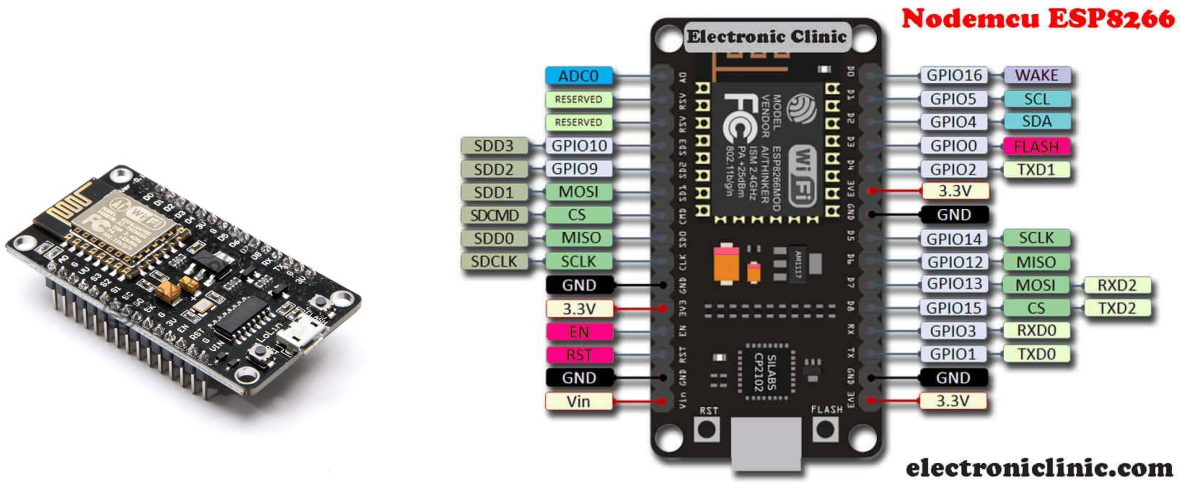


Figure 3.0: ESP8266 and its PIN Configuration

DHT22:

The DHT22 is a digital humidity and temperature sensor. It is a more accurate and expensive version of the popular DHT11 sensor. It has a measurement range of 0-100% for humidity and -40 to 80°C for temperature. The DHT22 sensor uses a single wire digital interface for communication with microcontrollers. It can be easily interfaced with microcontrollers like arduino, ESP8266 or Raspberry Pi with the help of libraries. The DHT22 sensor is commonly used in projects such as weather stations, temperature and humidity monitoring, and other applications that require accurate measurements of these parameters.

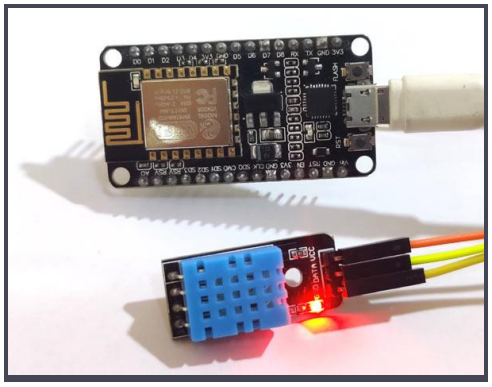


Figure 4.0: DHT22 & DHT22 with ESP8266

The DHT22 serves as a temperature and humidity sensor which gives the desired temperature and humidity output. This is connected to ESP8266 to get required readings for the above-mentioned particles.

- Output: Temperature and Humidity(in Celsius and g/m3 respectively).
- Works on: 3 ~ 5.5V.

Pin(ESP8266)	DHT22
GND	-
3.3V	+
D2	OUT

Table 1.0 ESP8266 Pin Mapping with DHT22.

3.3 Other Hardwares and Equipments

1. Jumper Wires
2. Micro SD Card (32GB)
3. Power Supply (power source to Raspberry Pi)
4. Micro USB cable (to connect ESP8266 to RaspberryPi)
5. Raspberry Pi 4
6. External Keyboard and mouse
7. Ethernet Cable (to connect Raspberry Pi to local network for IP Assignment & remote connection).

4. Software and Programming

The Micropython programming we did on ESP8266 helps us collect all the required sensor data and transmit it to Raspberry Pi which receives and processes later on. The screenshot of the code is provided below:

4.1 Libraries & Output

- **uos:** It provides access to some basic operating system services like changing the current working directory, renaming and removing files, and creating and removing directories.
- **machine**:It provides access to low-level hardware control on microcontroller boards like the one running MicroPython.It also provides access to the reset and sleep functions of the microcontroller, which can be used to reset the board or put it into a low power sleep mode.
- **gc**: Garbage collector module in MicroPython provides access to the garbage collection feature of the interpreter. It allows you to manage the memory usage of your program and make sure that the memory is being used efficiently.
- **ubinascii**: The ubinascii module in MicroPython provides functions for working with binary data and ASCII strings. It allows you to convert between binary data and ASCII strings, and provides functions for encoding and decoding data in various formats such as hexadecimal, base64, and more.
- **dht**: The DHT module in MicroPython is a library specifically designed to interface with the DHT22 digital humidity and temperature sensor. It provides an easy-to-use API for reading the

sensor's values, and abstracts away the low-level details of communicating with the sensor over a single-wire digital interface.

-

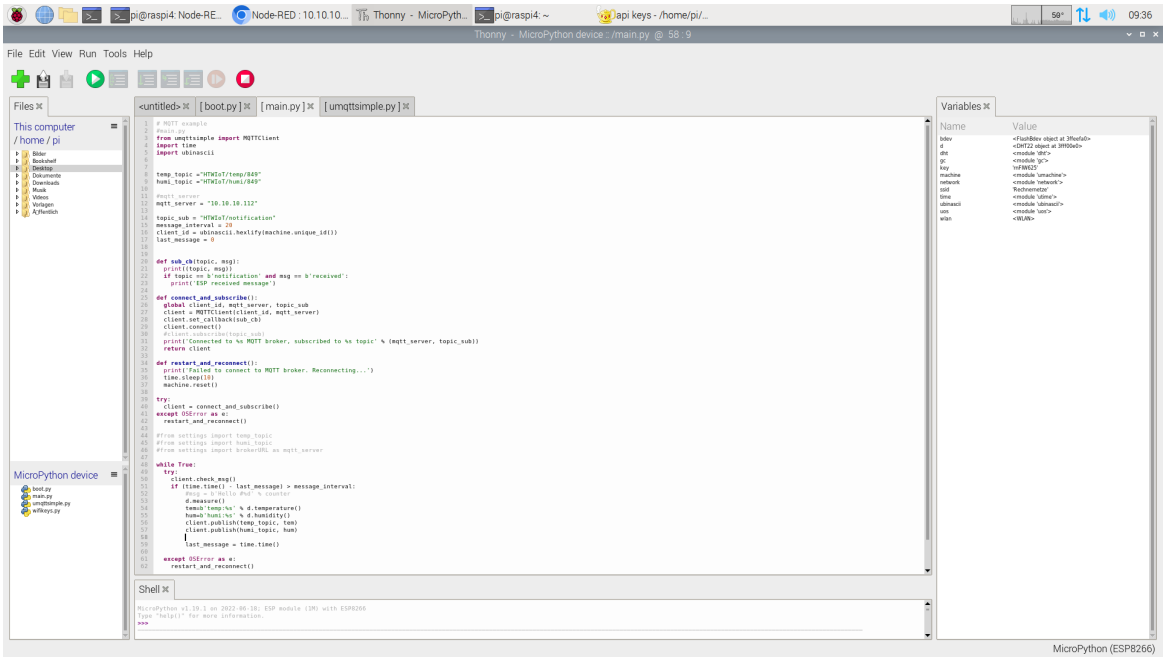


Figure 5.0 MicroPython programming

- **network**:The network module in MicroPython provides an easy-to-use API for working with network interfaces, such as Ethernet and WiFi. It allows you to scan for available networks, connect to a network, and configure network interfaces.
- **Output** : The program outputs temperature and humidity data. This will be then stored in a time series database through a proper format using Node RED.

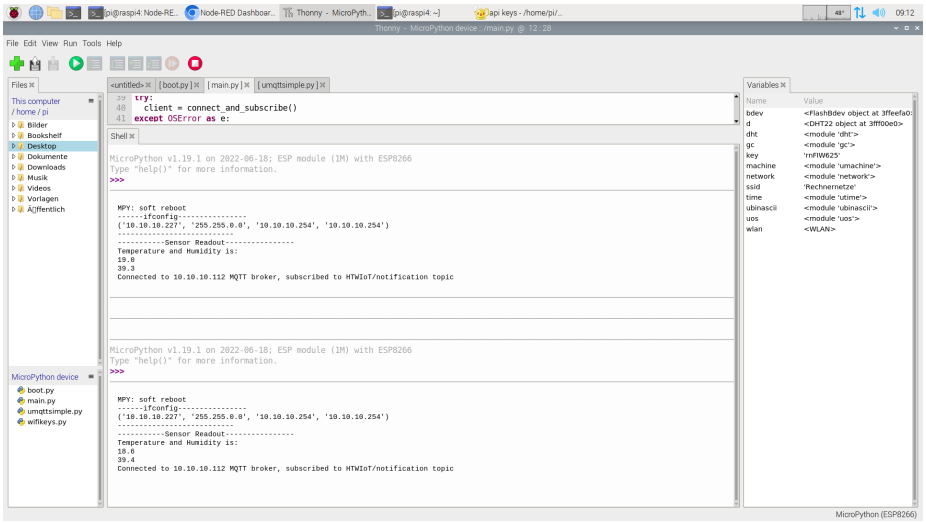


Figure 6.0 Output.

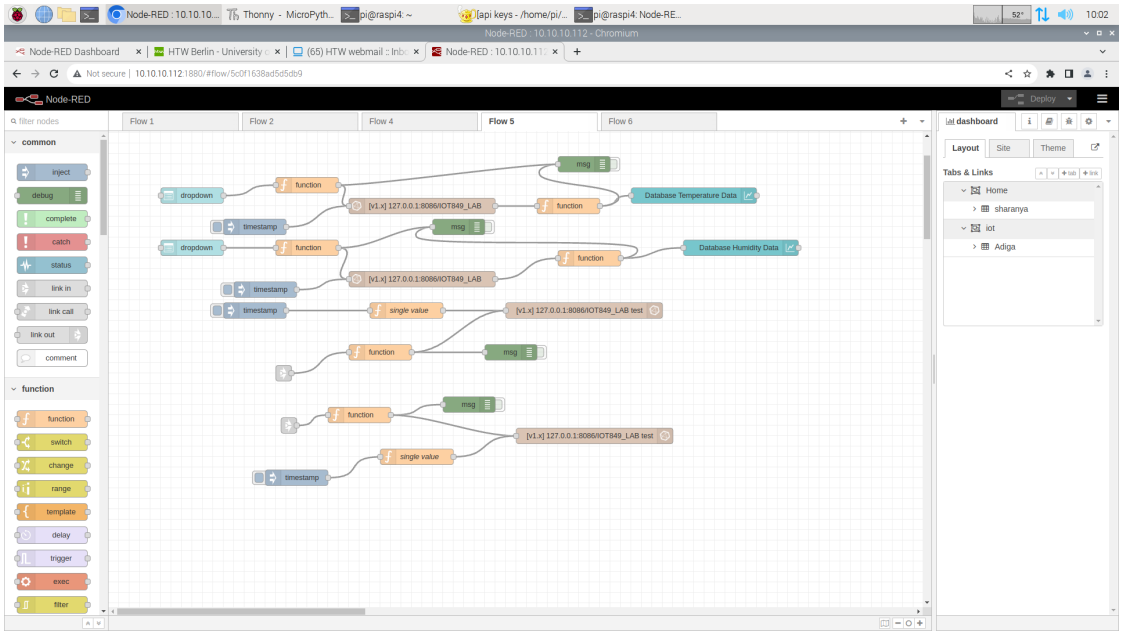


Figure 7.0 Node RED flow.

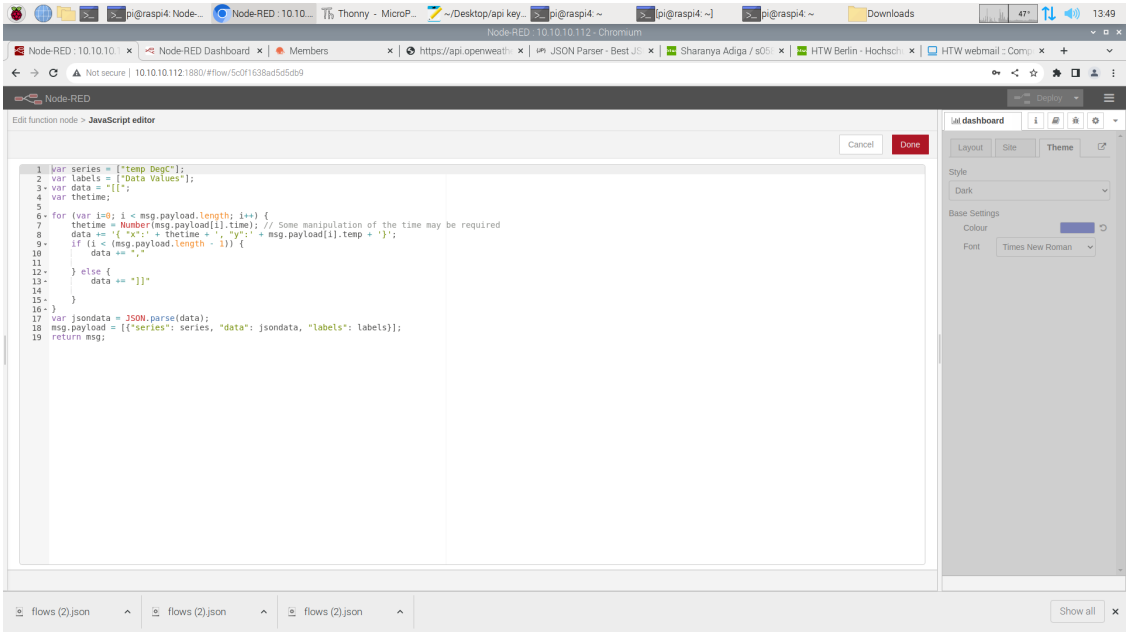


Figure 8.0 To display temp values from the database.

4.2 Node RED with InfluxDb Library

The data from the sensor and then is passed onto a function node written in JavaScript that allows our payload to be morphed into a format which we can directly store into the InfluxDb Database. We then send this formatted payload object to the Influx Out node which is configured to the database for data storage. And also this can be visualized by using the above code.

4.3 Influx-Out Node Configuration

The node “InfluxDb Out” helps output data to our database as a particular measurement as follows:

1) Measurement Name: temp and humi
2) Database: IOTLAB_849.

5. Data Storage and Visualization

5.1 InfluxDb - Time Series Database

We need to consider our data context which is a large volume of real-time data coming from our sensors. And therefore, a relational database was not a good choice here. Hence, we use a time series database “InfluxDb” for this task. This allows us to store sensor values as a measurement in a database and assign timestamps to each record of reading.

It can also be easily visualized due to its easy integration with Node RED (which we discuss ahead) . We use the measurement “temp and humi” in our “IOTLAB_848 database” monitor to print the results that are constantly being stored in it.

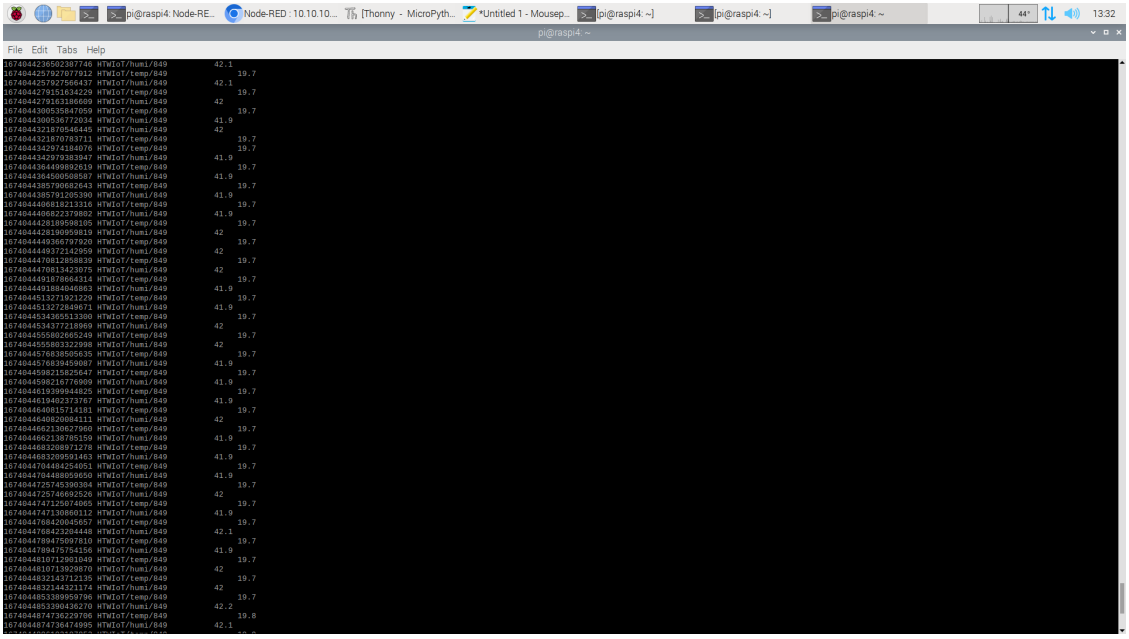


Figure 9.0 InfluxDb Database

5.2 Using Node RED Dashboard

For Visualizing the data that we gathered, we can use Node RED Dashboard as a choice because of its easy integration with InfluxDb. We simply view a node RED dashboard in the new tab. I choose Node RED dashboard as this is a very transparent and easy-to-handle data visualization and monitoring tool that

provides a detailed User Interface for manipulating the graphs in any way we want. (Querying, and other comprehensive functions for graph design).

5.3 Metrics Monitored

We therefore take the values of the sensors, and plot different graphs for each reading. The graphs are drawn according to the reading's units with respect to the timeframes during which they persist. The screenshot below can be seen. Temperature and humidity for a certain time is visualized in the below screenshots.

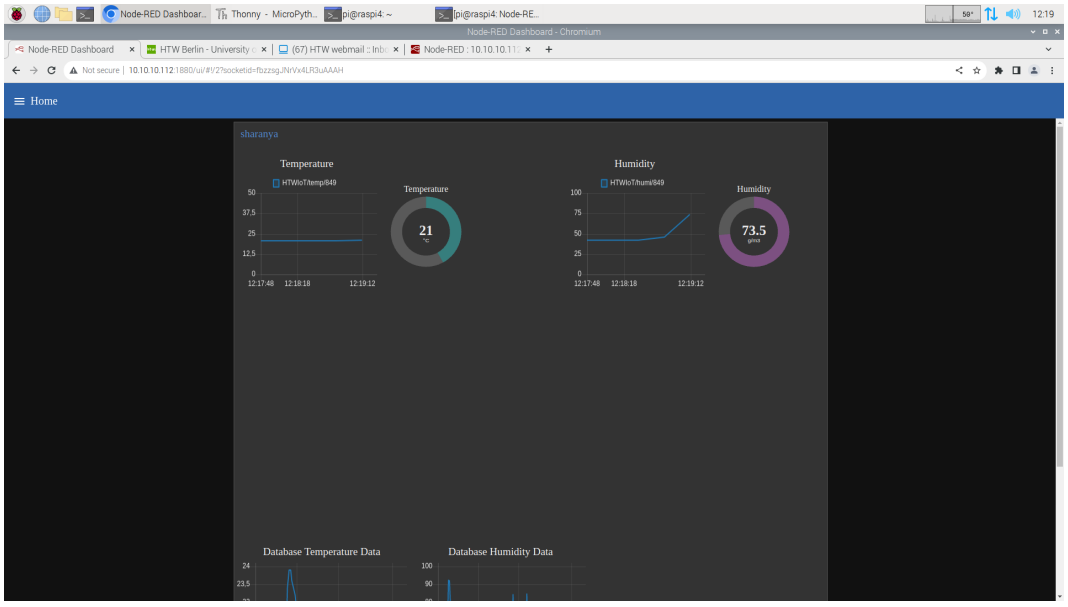


Figure 10.0 Temperature and Humidity data

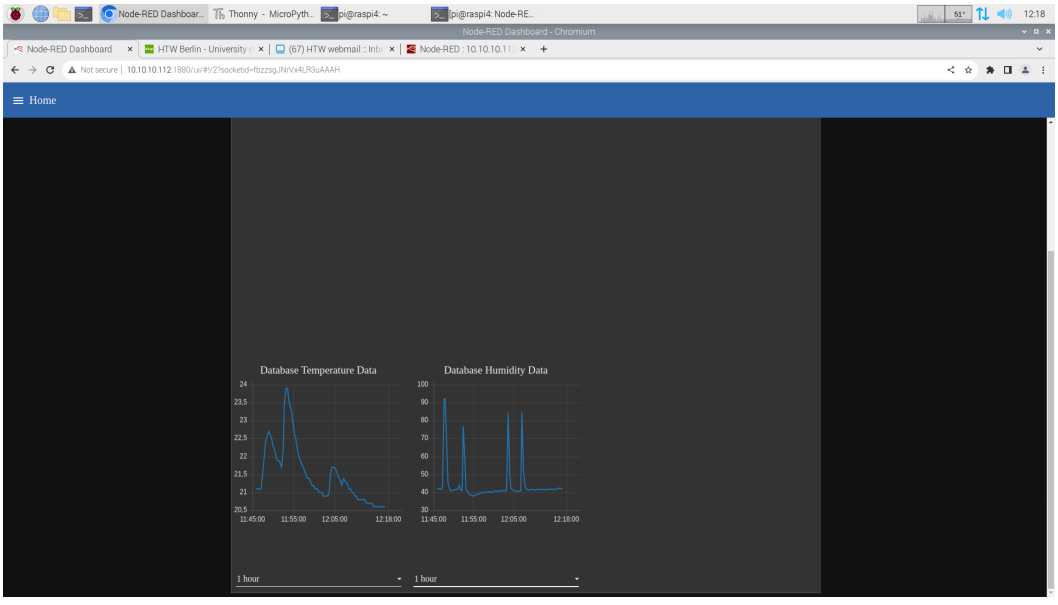
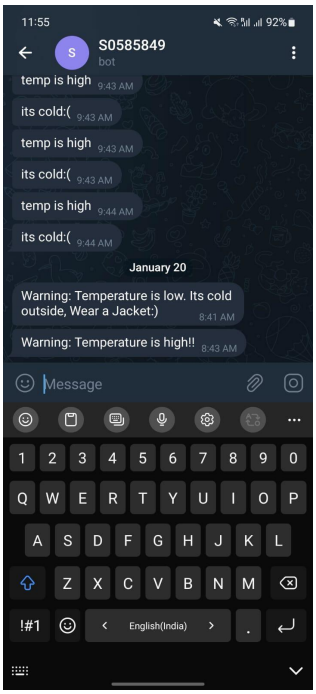


Figure 11.0 Temperature and Humidity data from the database for 1 hour.

The visualization helps provide such an insight and also gets an alarm on telegram which is explained further, once the user notices certain levels rising, they can then take necessary decisions.

5.4 Telegram Bot



A Telegram bot is a software program that interacts with the Telegram messaging platform, using the Telegram Bot API. Telegram bots can perform a variety of tasks, such as sending messages, handling inline queries, and processing payments. They are controlled by sending them commands, which are messages with specific text or other data. Telegram bots can be created using various programming languages, including Python, JavaScript, and PHP. The screenshot of the telegram bot when temperature reaches a certain level it alerts the user. A telegram sender/receiver node is connected to get the data from the sensor.

Figure 12.0 Telegram bot

5.5 Visualization of other cities's temperature and humidity

We can get the weather of other cities using OpenWeatherMap. Node RED node that gets the weather report and forecast from OpenWeatherMap. Installation has to be made and usage includes API key which we can get from https://openweathermap.org/appid. The node fetches the current weather or 5 day forecast at a location specified by city and country or latitude and longitude every 10 minutes - and outputs a **msg** if something has changed. Weather data is provided by OpenWeatherMap.org/.

Below are the screenshots of the dashboard and node RED flow. From figure 13.0 we can see the weather report namely Temperature, Humidity, wind speed and direction. Simultaneously we can visualize two different continent's weather report.

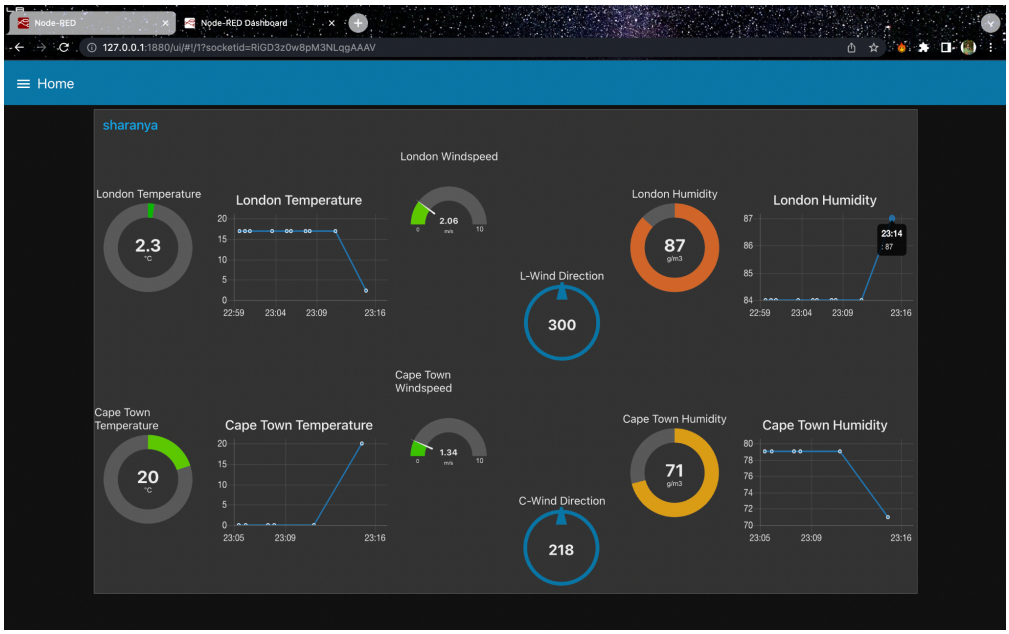


Figure 13.0 Weather report of London and Cape Town.

Figure 14.0 represents the Node RED flow using OpenWeatherMap nodes for 2 different cities. Only API key, city name and country name are required to setup OpenWeatherMap node. The weather data can be seen easily on the debug window(right side) as shown.

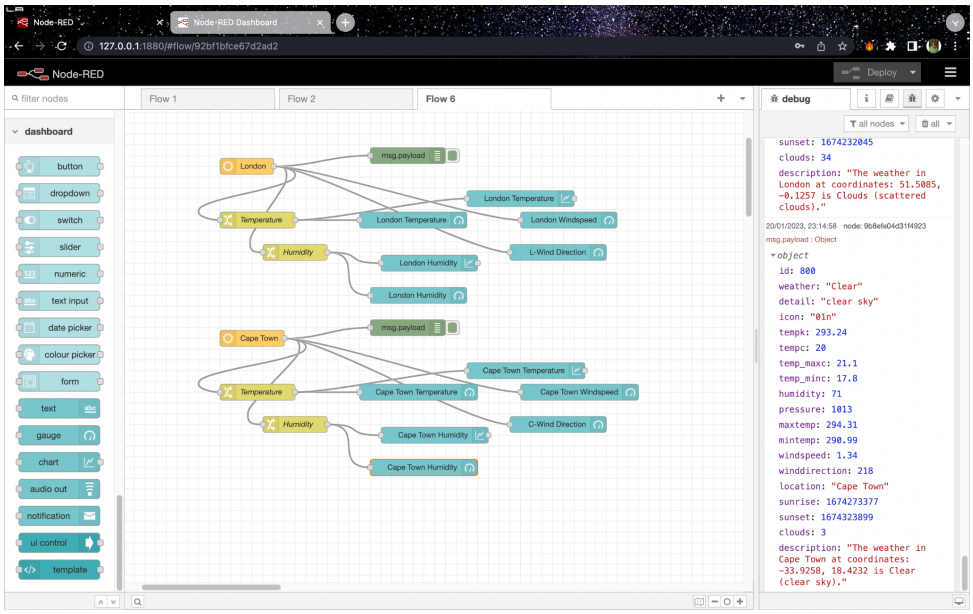


Figure 14.0 Node RED flow using OpenWeatherMap node

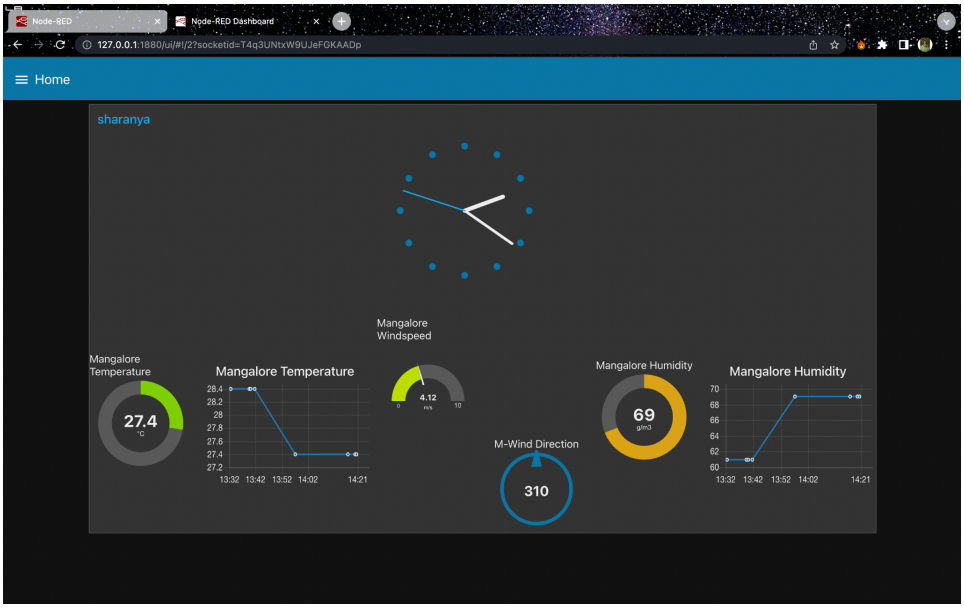


Figure 15.0 Weather report of Mangalore city with clock(Local time).

Figure 15.0 represents the weather report for Mangalore city along with an analog clock display. Here the ui_clock node has to be installed and it has to be integrated with the function node to get a local time.

6. Conclusion and Future Possibilities

We could take one step further by integrating another alarm system with our monitoring system and telegram bot to ring a buzzer if any threshold crosses. Thus, a user can acquire immediate notification in the form of mail. It could also be channeled with an email notification subscription network where a user could receive regular emails about Temperature of the surroundings (whether it's indoors or outdoors). Also along with these can integrate some more sensors and can display other values(to predict rain etc). Easily can be implemented in Smarthomes, Industries, Agriculture and healthcare. Overall, the future of this monitor system looks very promising as the demand is increasing due to the growing number of applications and industries that are adopting these sensors to optimize performance and improve efficiency.

References:

[1] VI cheat sheet (https://moodle.htw-berlin.de/pluginfile.php/1655361/mod_resource/content/1/VI-Help-Sheet-01-large2.jpg).
[2] MQTT Broker install (https://moodle.htw-berlin.de/pluginfile.php/1658745/mod_resource/content/1/installMQTT_Broker_en.md).
[3] Telegram Bot API(https://core.telegram.org/bots/api).
[4] Node RED (http://noderedguide.com/).
[5] Micropython Networking (https://docs.micropython.org/en/latest/esp8266/quickref.html#networking).
[6] Micropython DHT (https://docs.micropython.org/en/latest/esp8266/quickref.html#dht-driver).
[7] Git cheat sheet (chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://moodle.htw-berlin.de/pluginfile.php/1637188/mod_resource/content/1/git-cheat-sheet.pdf).
[8] InfluxDB Documentation (https://docs.influxdata.com/influxdb/v1.6).

GitLab Repository URL:

https://gitlab.rz.htw-berlin.de/s0585849/s0585849_iot