WEATHER REPORT SYSTEM

Project report

2023-24
Abishek Arun
Sabareesh CK
Sivakumar
Sheshadri R

Table of Contents

1. Abstract

Temperature and humidity monitoring are essential in various domains such as agriculture, industrial processes, and environmental control systems. This project aims to develop a robust monitoring system utilizing the DHT11 sensor for accurate data collection. The system architecture comprises a DHT11 sensor interfaced with an ESP32 microcontroller, facilitating seamless data acquisition. The collected data is wirelessly transmitted to both a Raspberry Pi and a laptop, enabling real-time monitoring and analysis.

The project focuses on addressing the need for continuous and reliable temperature and humidity monitoring without explicitly emphasizing it as an IoT system. By leveraging wireless communication and cloud-based storage capabilities, the system offers a versatile solution applicable across different environments and industries. The Raspberry Pi serves as an intermediary layer, converting temperature readings from Celsius to Fahrenheit before storing them in the Firebase Realtime Database.

Key components of the system include hardware and software architectures, interaction diagrams, and algorithmic implementations. The hardware setup involves connecting the DHT11 sensor to the ESP32 microcontroller and establishing wireless communication with the Raspberry Pi and laptop. The software implementation encompasses data transmission, temperature conversion, database storage, and data analysis.

The results demonstrate the successful implementation of the monitoring system, showcasing its efficacy in real-time temperature and humidity monitoring. The system's versatility and scalability make it suitable for various applications, ranging from agriculture to industrial automation. Future enhancements may focus on expanding the system's capabilities by incorporating additional sensors and refining data analysis algorithms for enhanced insights.

Overall, the project presents a comprehensive solution for temperature and humidity monitoring, catering to diverse needs across different domains.

2. Problems in Current Systems

In current temperature and humidity monitoring systems, several challenges hinder their effectiveness and efficiency. These challenges include:

- **Limited Scalability:** Traditional systems often lack scalability, making it difficult to expand or modify the monitoring infrastructure as needed. This limitation can be particularly problematic in environments where monitoring requirements change over time or where additional sensors need to be integrated.

- **Lack of Real-time Data Collection:** Many existing systems rely on manual data collection methods, which can result in delays in obtaining crucial information. Manual data collection is not only time-consuming but also prone to errors, as it relies on human intervention for data retrieval.

- **Cumbersome Data Analysis Processes:** Once data is collected, the analysis process can be cumbersome and time-consuming. In some cases, data analysis may require specialized software or expertise, further complicating the process and delaying decision-making.

2.1 Implementation

To address the challenges outlined above, the proposed system utilizes advanced technologies and techniques to improve temperature and humidity monitoring. Specifically, the implementation of the system focuses on:

- **Wireless Communication:** By leveraging wireless communication technology, the proposed system eliminates the need for wired connections between sensors and data processing units. This not only simplifies installation and maintenance but also enables real-time data transmission, ensuring timely access to critical information.

- **Cloud-based Storage Solutions:** The system utilizes cloud-based storage solutions, such as the Firebase Realtime Database, to store collected data securely and access it from anywhere. Cloud-based storage offers scalability, reliability, and accessibility, allowing for seamless data management and analysis.

- **Automation of Data Collection and Analysis:** The proposed system automates the data collection and analysis processes, reducing the reliance on manual intervention and minimizing the risk of errors. By implementing algorithms for real-time data processing and analysis, the system enables proactive decision-making based on accurate and up-to-date information.

2.2 Introduction to Fog and Edge Computing

In the realm of IoT-based systems, Fog and Edge Computing are two pivotal paradigms that address the challenges of data processing and analysis at the network's edge. They offer decentralized computing capabilities, bringing computational resources closer to the data source, thereby reducing latency, conserving bandwidth, and enhancing privacy and security.

- **Fog Computing:**

  Fog Computing extends the cloud to the edge of the network, typically within the same LAN or WLAN, providing a hierarchical architecture where computing resources are distributed across multiple levels. It enables real-time data processing and analysis by offloading computational tasks from end devices to nearby fog nodes. Fog nodes, often deployed in close proximity to the data source, offer low-latency processing and enable timely decision-making without relying solely on distant cloud servers.

- **Edge Computing:**

  Edge Computing, on the other hand, pushes the computational infrastructure even closer to the data source, typically within the device itself or at the network's edge. It focuses on processing data locally, at or near the point of data generation, reducing the need for data transmission to centralized servers for analysis. Edge devices, equipped with computing capabilities, handle data processing tasks autonomously, minimizing latency and network congestion.

- **Integration with IoT-Based Systems:**

  In the context of IoT-based temperature and humidity monitoring systems, Fog and Edge Computing play crucial roles in enhancing system efficiency and performance. The utilization of these computing paradigms allows for real-time processing and analysis of sensor data, enabling timely response to environmental changes and optimizing resource utilization.

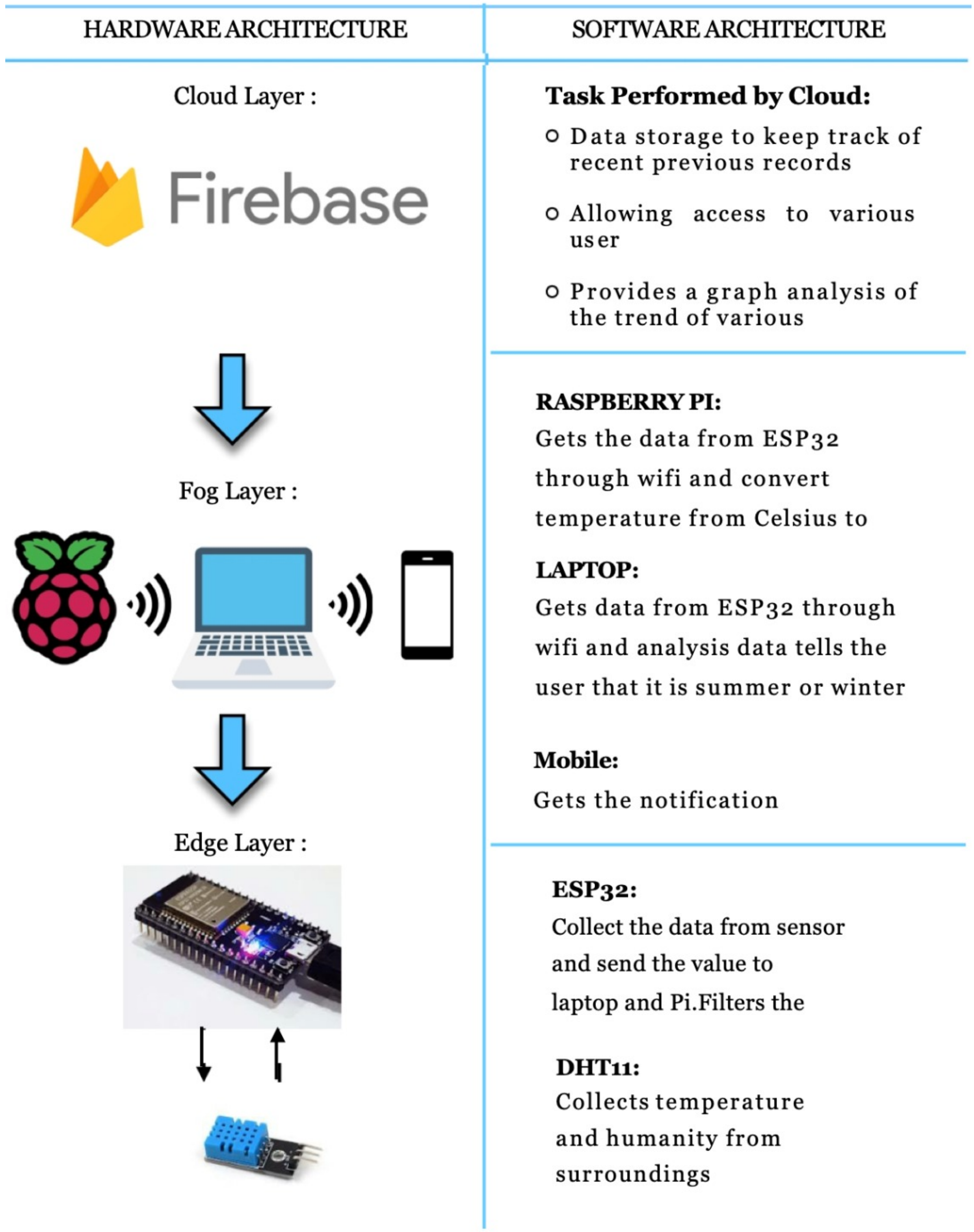
3. Problem Statement

In various sectors like agriculture, industrial automation, and environmental control, the lack of real-time temperature and humidity monitoring systems hampers efficient decision-making. Existing methods often suffer from manual data collection, leading to delays and inaccuracies in data analysis. This hinders the timely identification of environmental changes and the implementation of necessary interventions. As a result, there is a critical need for an automated monitoring system capable of continuously collecting, analyzing, and presenting temperature and humidity data in real-time to facilitate proactive decision-making in various applications.

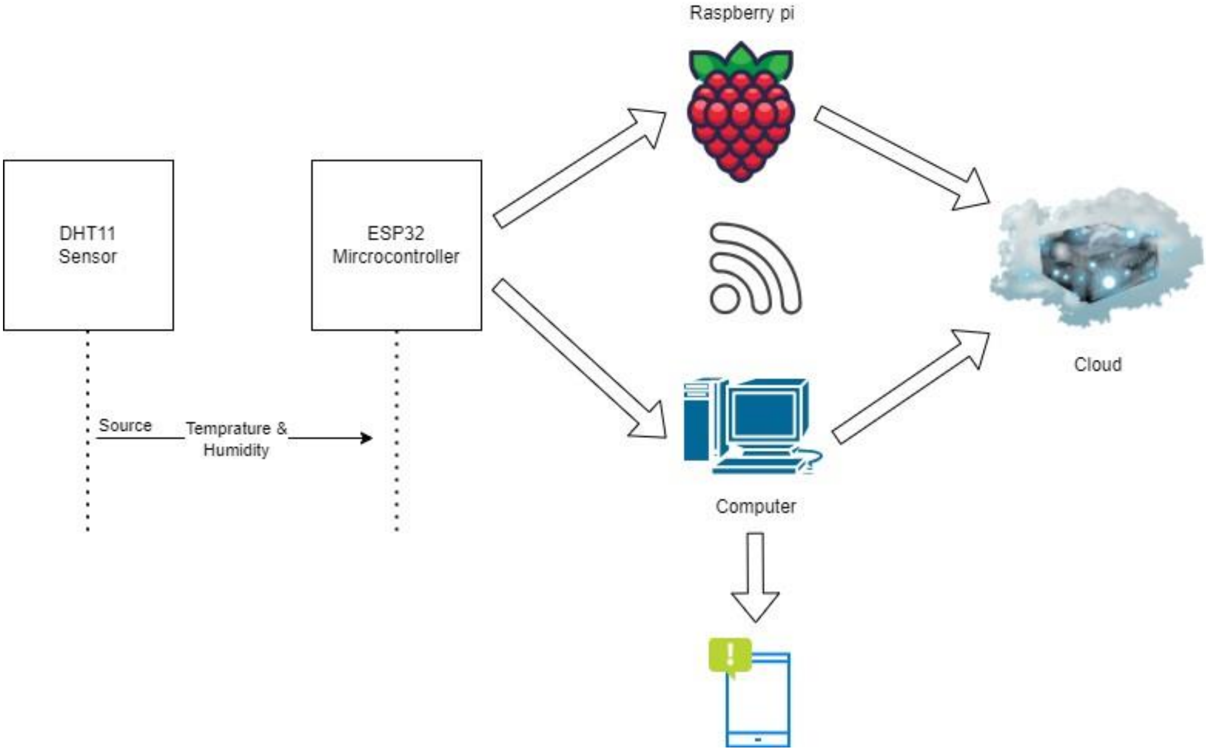
4. Proposed System

4.1 Hardware and Software Architecture

The proposed system architecture consists of a DHT11 sensor connected to an ESP32 microcontroller, which facilitates wireless data transmission to both a Raspberry Pi and a laptop. The Raspberry Pi serves as an intermediary layer, converting temperature readings from Celsius to Fahrenheit before storing them in the Firebase Realtime Database.



4.2 Interaction Diagram



4.3 Algorithm/Technique of System

The system utilizes the DHT11 sensor for data collection and implements Forward Error Correction (FEC) algorithms to ensure reliable data transmission. The Raspberry Pi performs temperature unit conversion and stores the data in the Firebase Realtime Database.

Raspberry Pi Programming:

The Raspberry Pi is programmed using Python scripts to receive data from the Arduino ESP32 over Wi-Fi and perform various tasks such as temperature unit conversion.

Code:

https://github.com/asequas/weather-report-system/blob/main/rasp_firebase.py

Explanation:

- The script establishes a connection to the Firebase database using the Pyrebase library.
- It continuously reads temperature, humidity, and rainfall data from the specified paths in the database.
- The temperature data is converted from Celsius to Fahrenheit for easier interpretation.
- Additional functionalities, such as sending data to Adafruit IO or performing data analysis using machine learning models, can be included based on project requirements.
- Finally, the script delays execution for a few seconds before reading data again to avoid overloading the system.

Arduino Programming:

The Arduino ESP32 is programmed to collect data from sensors and transmit it to the Raspberry Pi via Wi-Fi.

Code:

https://github.com/asequas/weather-report-system/blob/main/arduino.py

Explanation:

- The Arduino code begins by including necessary libraries for WiFi communication (WiFi.h), sensor communication (Wire.h), and the DHT sensor (DHT.h).
- It defines the WiFi credentials (SSID and password) and the IP address of the Raspberry Pi server.
- In the setup function, it initializes serial communication, the DHT sensor, and establishes a WiFi connection.
- The loop function reads temperature and humidity data from the DHT sensor and checks for validity.

- If the data is valid, it establishes a connection to the Raspberry Pi server and sends an HTTP GET request with the temperature and humidity data as parameters.
- The server responds to the request, and the Arduino prints the server's response to the serial monitor.
- This process repeats in a loop, with a delay of 2 seconds between each iteration.

4.4 Pushover Notification Integration

In addition to the core functionalities of the weather monitoring system, a new feature has been added to enhance user experience and energy efficiency. When the temperature readings exceed a certain threshold, indicating a need for cooling, the system sends a notification to the user's smartphone via the Pushover notification API. This notification prompts the user to turn on the air conditioning system, ensuring a comfortable indoor environment. Conversely, when the temperature decreases below another predefined threshold, signaling cooler conditions, another notification is sent to the user's smartphone to turn off the air conditioning system, thereby conserving electricity. The integration of the Pushover notification API enables real-time communication and control, enhancing the overall effectiveness and user-friendliness of the weather monitoring system.

5. Results

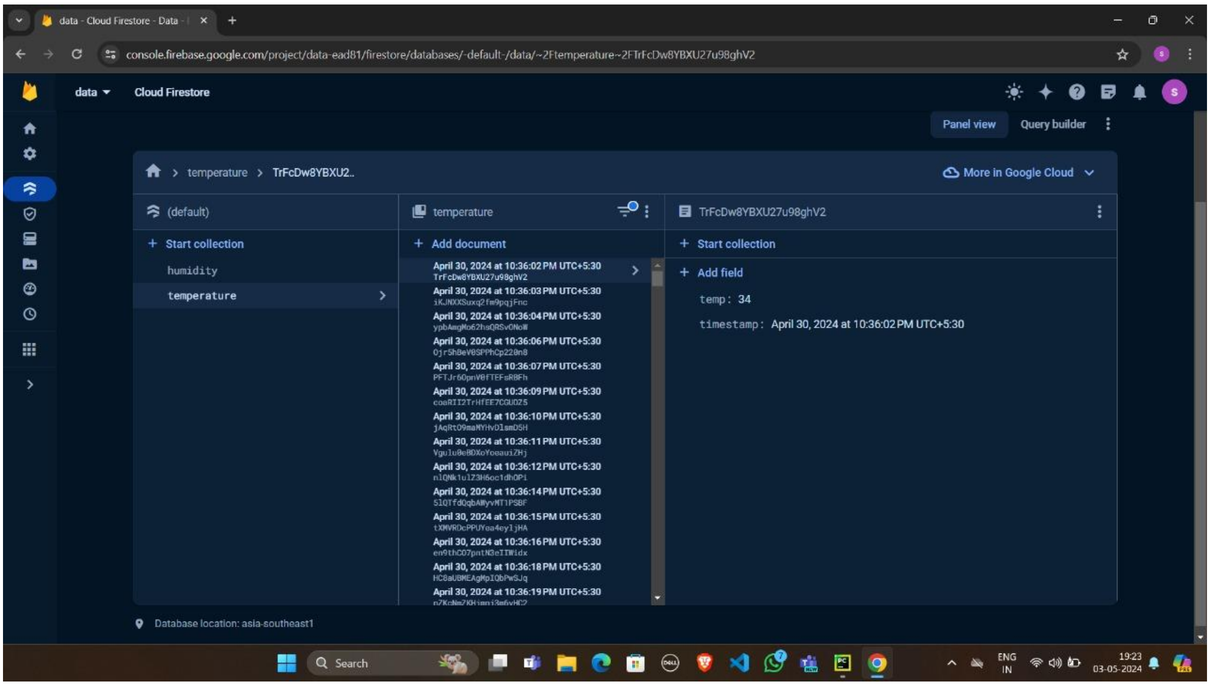
The implementation of the proposed system demonstrates successful real-time temperature and humidity monitoring, showcasing its efficacy in various applications such as agriculture, industrial automation, and environmental monitoring.

5.1 Implementation and Simulation Results

The system's versatility and scalability make it suitable for diverse environments and industries. Future enhancements may focus on expanding the system's capabilities by incorporating additional sensors and refining data analysis algorithms for enhanced insights.

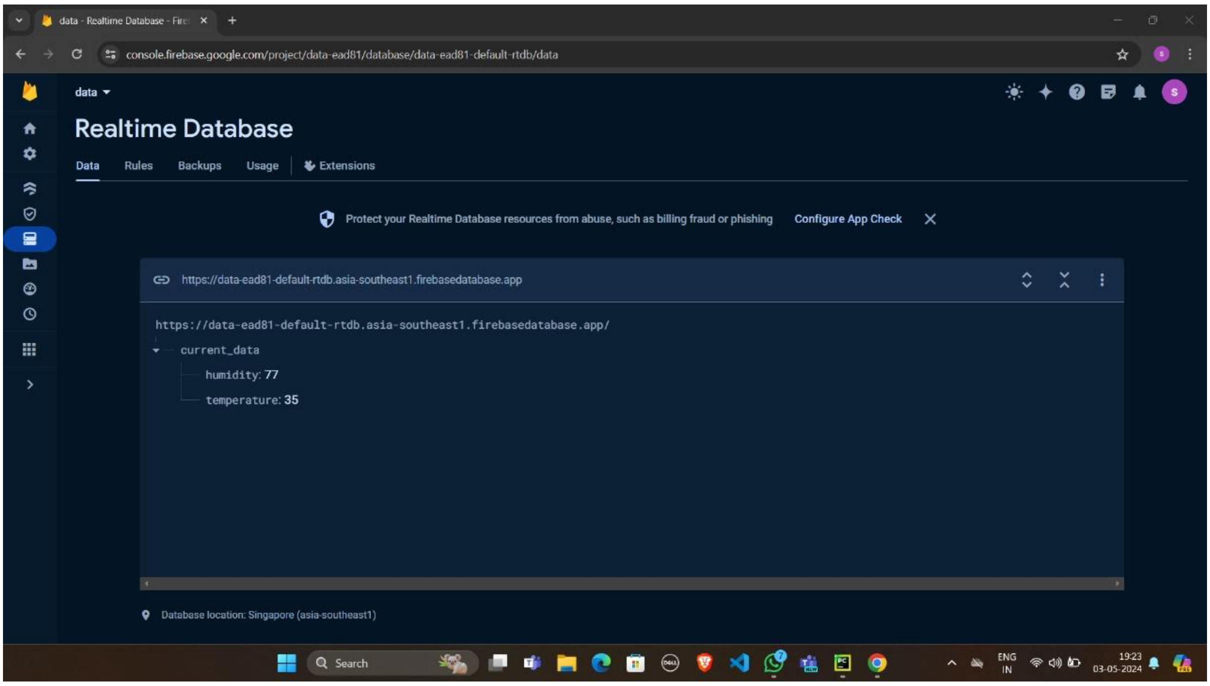
following screenshots demonstrate the functionality and effectiveness of the proposed IoT weather monitoring system:

- **Data Collection and Storage in Firebase**



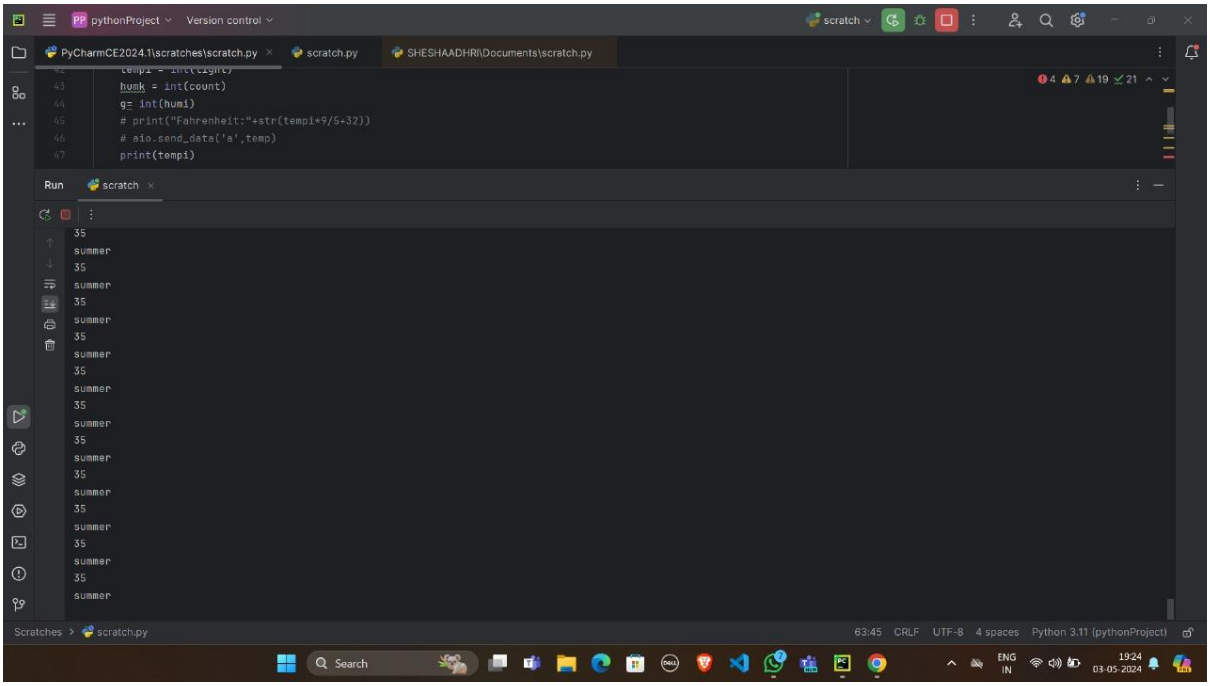
This screenshot illustrates the successful collection and storage of temperature and humidity data in the Firebase Cloud. The data is transmitted from the edge layer (sensors and Arduino ESP32) to the fog layer (Raspberry Pi) and then to the cloud layer (Firebase), where it is securely stored for further analysis and access.

- **Real-Time Data Transmission to the Cloud**



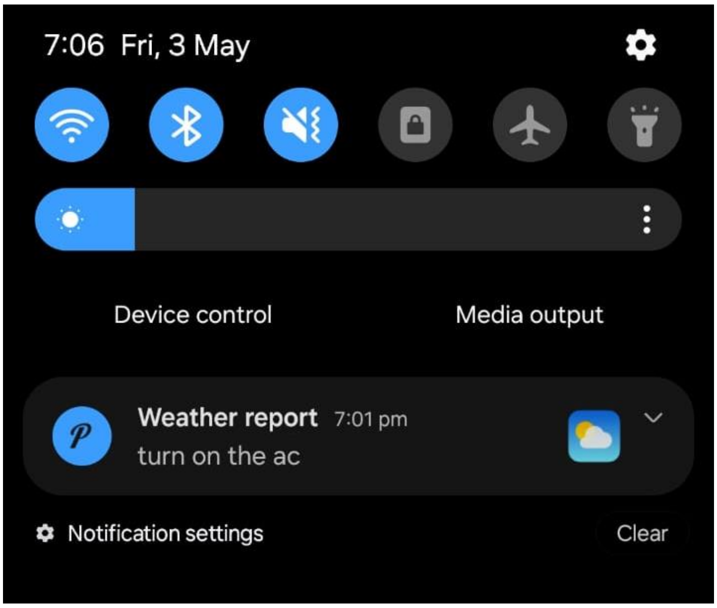
This screenshot showcases the real-time transmission of temperature data from the fog layer (Raspberry Pi) to the cloud layer (Firebase). The continuous flow of data ensures timely updates and enables remote monitoring and analysis of environmental conditions.

- **Temperature Analysis on PC**



In this screenshot, the PC performs temperature analysis based on the received data to determine whether the current temperature indicates summer or winter conditions. This analysis is crucial for making informed decisions and taking appropriate actions, such as adjusting heating or cooling systems.

- **Pushover Notification on Phone**



The final screenshot demonstrates the receipt of a push notification on the user's smartphone via the Pushover notification API. This notification informs the user about the temperature analysis results and prompts them to take action, such as turning on or off the air conditioning system, based on the detected weather conditions.

7. Conclusion

The proposed system offers a comprehensive solution for real-time temperature and humidity monitoring, addressing the limitations of existing systems. By leveraging wireless communication and cloud-based storage solutions, the system ensures continuous and reliable data collection and analysis. The successful implementation of the system underscores its potential for various applications, ranging from agriculture to industrial automation. Future developments may further enhance the system's capabilities to meet evolving needs and challenges in temperature and humidity monitoring.

8. References

1. B. Satyanarayana, V. V. B. Reddy, and M. V. N. K. Prasad, "A Comprehensive Study on Fog and Edge Computing: Concepts, Frameworks, Applications, and Research Directions," IEEE Access, vol. 8, pp. 34289-34310, 2020.

2. R. K. Singh and S. Singh, "Smart Weather Monitoring and Forecasting System Using IoT," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-6.
3. V. Karuppiah et al., "An IoT-Based Smart Weather Monitoring and Prediction System Using Machine Learning Techniques," 2021 International Conference on Information and Communication Technology for Intelligent Systems (ICTIS), Bangkok, Thailand, 2021, pp. 1-6.

4. K. Sharma and A. Pachauri, "Development of a Low-Cost IoT Based Weather Monitoring System Using Raspberry Pi," 2021 International Conference on Communication Systems, Computing and IT Applications (CSCITA), Ghaziabad, India, 2021, pp. 1-6.