

# A Real-Time Monitoring IoT-Based Weather Station: A Comparative Analysis of Web-Based and Cloud Storage Solutions

AKPAKWU A. GODFREY<sup>1</sup>, TERKUMA CALEB<sup>2</sup>, TERWASE I. AONDONA<sup>3</sup>

<sup>1, 2</sup>*Department of Electrical and Electronic Engineering, Joseph Sarwuan Tarka University, Makurdi, Benue State, Nigeria.*

<sup>3</sup>*Department of Mechanical Engineering, Heriot Watt University, Edinburgh UK.*

**Abstract-** *The advancement of IoT (Internet of Things) technology has revolutionised environmental monitoring, enabling real-time data collection and analysis across broader geographical contexts. This paper presents the design and implementation of a cost-effective IoT-based weather station, integrating widely available sensors to measure temperature, humidity, pressure, and rainfall. The system transmits data to both a local display and a cloud platform for data visualisation and storage. A comparative analysis between traditional web-based data presentation using a local host and Google Cloud-based solutions highlights the superior flexibility, accessibility, and storage capacity offered by Google Cloud's Sheets. By leveraging Google Sheets' capabilities, this study provides an innovative approach to weather data management, overcoming limitations inherent to local hosting methods. The proposed weather station system is not only affordable and scalable but also adaptable for diverse applications, including agriculture, disaster management, and urban planning. The findings demonstrate the potential of integrating cloud technology into IoT weather stations, offering robust access for use by different applications, real-time monitoring with low operational costs, and thus making it highly suitable for deployment in remote or resource-constrained environments.*

**Keywords:** *Internet of Things (IoT), Weather Station, Environmental Monitoring, Cloud Storage, Sensor Technology.*

## I. INTRODUCTION

In today's rapidly advancing technological world, the ability to monitor and predict environmental conditions in real-time has become essential for numerous applications, including agriculture, urban planning, disaster management, and meteorology. Weather stations collect climatological data regarding the present weather conditions in a certain region and predict the local meteorological patterns. In addition, weather differs every day. For accurate result analysis, recording of current weather updates

is critical [1]. Weather stations, which measure parameters like temperature, humidity, atmospheric pressure, and rainfall, serve as critical tools in gathering data that enables informed decision-making and accurate weather forecasting [2]. Traditionally, local data collection has been the primary use of weather stations, limiting their utility in broader geographical contexts.

Clouds maintain climate characteristics. This means that anyone who has been validated can access climate data online from anywhere. In the event of a disaster such as a significant rainstorm, fire, temperature, terrible wind, or soaker the information is relayed to certified individuals, which is highly beneficial in restoring the prior situation. The weather in any environment can be influenced by several factors, including rainfall, moisture, air temperature, and air pressure. It is easier to quickly presume how the external world is being affected by taking these factors into account [3].

The advent of the Internet of Things (IoT) has transformed how data is collected, transmitted, and applied across various sectors. IoT is a network of interconnected physical devices, or "things", embedded with sensors, software, and communication technologies to collect and exchange data over the Internet [4-5]. IoT involves low-power devices that communicate with one another via the Internet. The IoT has captivated the research community, aiming to connect wearables, sensors, smart appliances, washing machines, tablets, smartphones, smart transportation systems, and other entities to a unified interface for intercommunication [6-11]. IoT interlinks "things" and facilitates machine-to-machine (M2M) communication, allowing data exchange across diverse devices without human involvement [12]. These tasks can be accomplished via an uninterrupted communication

medium, as stated in [13]. The IoT is anticipated to foster an environment that will affect various facets of daily life and business applications, hence contributing to the expansion of the global economy through Massive and Critical IoT, contingent upon the specific applications implemented [5]. This technology has allowed for real-time environmental monitoring from remote locations, making it useful in fields such as traffic management, healthcare, and industrial safety [14].

Massive IoT applications necessitate the connection of a vast number of smart devices, which may be implemented in shipping environments, smart homes, smart cities, smart power systems, and real-time and agricultural monitoring environments, among others, demanding frequent updates to the cloud at a low end-to-end cost [5]. Critical IoT applications, such as remote healthcare systems (for clinical monitoring and assisted living), traffic management, industrial control (drones/robots/vehicles), and the tactile Internet, necessitate enhanced availability, reliability, safety, and reduced latency to ensure optimal end-user experience, as failures in these applications could lead to significant repercussions. The myriad application opportunities afforded by IoT are extensive, and their full potential will only be realised by connecting a greater number of smart devices to the Internet [5].

Most IoT-enabled weather stations publish their collected data on dedicated web pages, allowing users to access real-time information from anywhere in the world. However, this approach can present limitations in terms of flexibility, data storage, and advanced analytics [14]. In this study, we introduce an alternative method by sending collected weather data to Google Sheets. This approach not only simplifies data management but also provides greater flexibility in data visualisation, analysis, and sharing across multiple platforms. In addition, the performance of Google Sheets will be compared with traditional website-based data presentation in terms of storage capacity, accessibility, and ease of use.

## II. LITERATURE REVIEW

Several studies have explored the design, functionality, and application of IoT-based weather stations, with each contributing to the evolution of these systems in unique ways. Recent advancements

further highlight the evolution of IoT-based weather stations. Jabbar et al. (2024) [15] investigated the use of LoRaWAN (Long Range Wide Area Network) for enhanced communication capabilities, addressing data transmission challenges in remote areas.

Iswanto et al. (2020) [16] identifies the core components required for an IoT-based weather station, outlining the need for an input unit, power supply unit, control unit, communication unit, and output unit. This modular design ensures efficient data capture and transmission, which enables continuous weather monitoring.

Since the accuracy and reliability of IoT weather stations heavily depend on the calibration of the sensors used in data collection, Mohammed et al. (2020) [17] note that this step is necessary to reduce errors and also demonstrate the necessity of sensor calibration in their study on temperature and humidity measurements, where they utilised standard calibration techniques with conventional devices as reference points, thus ensuring the precision of their IoT-based weather station.

Comparative studies have also been conducted to assess the suitability of different microcontrollers for IoT weather station implementation. Dinkar et al. (2017) [18] compared the Arduino, Raspberry Pi, and ESP8266 development boards, concluding that while Arduino is excellent for handling analog signals from sensors, Raspberry Pi stands out for fulfilling most IoT system requirements due to its processing capabilities. On the other hand, ESP8266 remains the most cost-effective and efficient choice for wireless sensor networks, offering seamless connectivity in a compact form factor. This comparative investigation offers helpful advice for selecting the appropriate hardware based on the specific requirements of a weather monitoring system.

In a similar manner, Vivek et al. (2017) [19] took this concept further by incorporating a 16x2 LCD for weather data visualisation using a Raspberry Pi. Their weather forecasting system demonstrated the potential of using advanced microcontrollers for more complex weather data processing. By leveraging the processing power of Raspberry Pi, they enabled a more robust and scalable weather station that could be easily adapted for forecasting purposes.

Since the visualisation of environmental data remains a key challenge in the design of IoT weather stations, Ravi et al. (2016) [2] proposed a solution by developing an IoT-based weather station using the NodeMCU microcontroller. Their design included a GPS module to provide location-based measurements, allowing users to pinpoint the exact location of the data collected. Additionally, their study utilised a Liquid Crystal Display (LCD) to present real-time data at the base station, making the system both user-friendly and highly functional. Their approach highlights the importance of both remote access and local visualisations in weather monitoring systems.

Bulipe et al. (2016) [4] demonstrated the use of the ESP8266 Wi-Fi module for cloud-based weather data storage, utilising the ThingSpeak platform to store and retrieve data via the internet. This approach provides a low-cost solution for global access to weather data, using HTTP protocols over local area networks. Their system showcased the growing trend of integrating cloud platforms into IoT weather stations, enabling remote access to data from any internet-connected device in the world.

Bruce (2015) [20] highlights the far-reaching benefits of IoT weather stations, noting their capacity to provide real-time data and improve the safety and efficiency of operations in various industries. He highlighted possible applications in sectors such as transport, medical, industrial, and military, where real-time weather data is crucial for operational success, and notes that the ability to monitor hazardous environments remotely reduces the risk to human life and enhances the accuracy of environmental readings, thus contributing to energy conservation and better resource management.

### III. MATERIALS AND METHODS

Figure 1 shows the block diagram of the developed real-time monitoring IoT-based weather station system. The power supply unit provides a stable 5V regulated voltage to all components of the system, ensuring consistent operation. The DHT11 sensor module is responsible for measuring both temperature and humidity. It utilises its built-in Analog-to-Digital Converter (ADC) to convert the analog readings into digital values, which are then sent to the microcontroller for further processing.

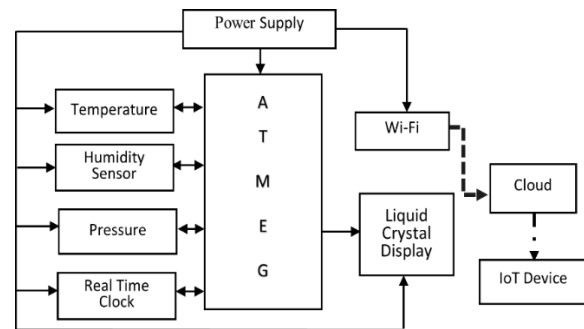


Figure 1: Block Diagram of Real-Time Monitoring IoT-Based Weather Station.

The implementation of this research is majorly divided into two parts: the hardware and software. The Arduino language, which is an abstraction of the C language, was used in the Arduino IDE (Integrated Development Environment) to write the required code. The DHT sensor library for ESPx by beegoe\_tokyo for temperature was used. The HTTPS redirect library was used for publishing results to Google Sheets and the Google API library. A dedicated library was not used for the rain sensor. However, an algorithm was developed to decode the on-and-off logic of the module.

Also, instead of using an Arduino board and an ESP8266 shield separately, the study leveraged the capability of the Wemos D1 microcontroller board.

The Wemos D1's ESP8266 chip (the WiFi shield) operates in station mode, allowing it to connect to an access point for internet connectivity. In this mode, the Service Set Identifier (SSID) and password of the access point are hardcoded into the microcontroller's program. When powered on, the WiFi shield scans for the specified SSID. Upon locating it, the device sends a connection request to the access point. The access point then verifies the provided password. If the password is correct, a connection is established; otherwise, the request is denied.

To ensure a stable connection, the program is designed to attempt reconnection up to five times if it is lost. It is important to note that the access point must have internet access for the Wemos D1 (through the ESP8266) to successfully transmit data to the webserver. The webserver itself is created using HTML and CSS, embedded directly in the Arduino code. Once the ESP8266 is connected to the network, its IP address can be used to access the server. Any device connected to the same access point can open a web browser and enter the ESP8266's IP address to view the web page and monitor data.

Sending the measured sensor values to Google Sheets involved writing two scripts: one on the microcontroller and another on the server side of Google Sheets. On the microcontroller (Wemos D1), data was collected and transmitted, while the server-side script in Google Sheets received and stored the data. Figure 2 shows a sample of a Google Sheet.

	A	B	C	D	E
1	Date	Time	value0	value1	value2
2					
3					

Figure 2: A Custom Google Sheet Format Showing the Columns.

#### A. The Circuit Diagram

The circuit diagram presented in Figure 3 was designed using Proteus software (version 8.10). The voltage supplied by the solar panel is about 7V-9V. This voltage is conditioned by appropriate components to charge a 5V battery, which then supplies 5V DC to the circuit.

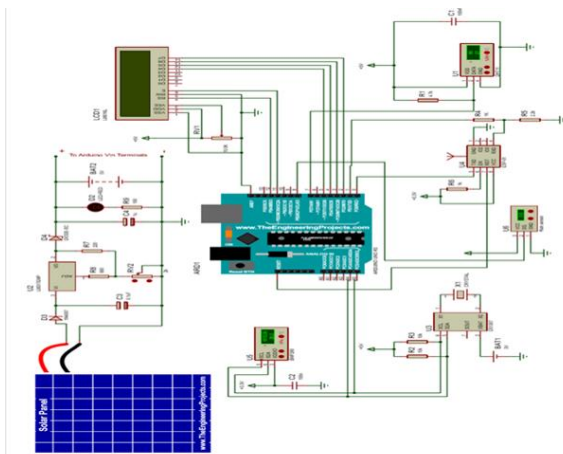


Figure 3: The Circuit Diagram of the Real-Time Monitoring IoT-Based Weather Station.

DHT11 is labelled U1, BMP280 is labelled U2, DS1307 is labelled U3, ESP8266 is labelled U4, and the rain sensor module is labelled U7. All these sensors and modules are connected to the microcontroller.

The DHT11 power pin (pin 1) is connected to the 5V power rail, while the data pin (pin 2) is connected to the Arduino IO7 pin. Meanwhile, pin 3 is not connected, and the ground pin is connected to the ground rail. A 4.7k $\Omega$  connected between the power supply and the data pin is used as a pull-up resistor. A 100nF capacitor connected between the power supply

and the ground pin is used to filter the input voltage. Temperature and humidity measured are processed and sent by the IC on the DHT11 sensor module to the microcontroller via the data pin.

The BMP280 power pin (pin 3) is connected to the 3.3V supply of the Arduino board since this sensor operates on an input voltage of 3.3V. The GND pin is connected to the circuit's ground rail. The SCL (Serial Clock) pin, which is pin 5, is connected to Arduino analogue pin A4, while the SDA (Serial Data) pin, which is pin 6, is connected to Arduino analogue pin A5; the SCL provides the clock pulses, while SDA is used to send the measured pressure and altitude values, which can be read by the microcontroller by use of the "bmp.readPressure()" and "bmp.readAltitude()" codes, respectively. A 100nF capacitor is connected between the Vin and the ground pins to filter the input voltage.

The rain sensor module consists of the sensing pad and the analog to digital converter (ADC). The sensing pad's two pins are connected to the ADC's input pins. These values are converted to digital ones by ADC. The ADC module's Vin pin is connected to the 5V power rail. The GND pin is connected to the circuit's ground rail. The ADC has two outputs: analog and digital. This study specifically used the digital pin, which is connected to the Arduino IO8 digital pin. A 100nF capacitor is connected between the Vin pin and the ground pin to filter the input voltage.

With the unavailability of the ESP8266 NodeMCU development board on the Proteus software used to design the circuit, the NodeMCU's chip was modelled to represent it on the circuit. The VCC pin (pin 8) is connected to the 3.3 V supply rail, while the GND pin (pin 1) is connected to the ground rail. The transmit pin (pin5) is connected to the receive (RX) pin of the Arduino, while the receive pin (pin4) is connected to the transmit (TX) pin of the Arduino through a voltage divider network comprising 1k $\Omega$  and 2.2k $\Omega$  resistors since the ESP8266 chip works on 3.3V. The enable pin (pin 6) is connected to VCC through a 1 k $\Omega$  resistor, while the reset pin (pin 7) is connected to the Arduino reset pin through a voltage divider comprising 1k $\Omega$  and 2.2k $\Omega$  resistors. Pin2 and pin3 are non-connected.

The rain clock module (DS1302), which is represented as U3, has its ground pin connected to the

ground rail of the circuit. Pin5 (SDA) and pin6 (SCL) are connected to the Arduino's analogue pins A4 and A5, respectively. 10k $\Omega$  resistors are connected between the 5V supply and the SCL and SDA pins, respectively, as pull-up resistors. The VBAT terminal (pin 3) has a 3V battery connected to it to maintain time measurements even when the circuit's power supply is disrupted. Furthermore, a 32.67 kHz crystal oscillator is connected between pin 1 and pin 2.

Lastly, the Liquid Crystal Display (LCD) was connected using the 4-bit connection mode, where data bits D4, D5, D6, and D7 are connected to Arduino digital pins IO5, IO4, IO3, and IO2, respectively. The LCD's reset pin is connected to Arduino digital pin IO12, while its enable pin is connected to Arduino pin IO11. The VSS pin is connected to the ground, while the VDD is connected to the circuit's 5V supply rail. A 10k $\Omega$  potentiometer was used to enable contrast adjustment. Therefore, the middle terminal of the potentiometer is connected to the LCD's VEE terminal. It must be noted that these data pins can be changed to any digital pin of one's choice in the program code.

The circuit works on a compiled code uploaded to the ATmega328p microcontroller IC via the serial programming port of the Arduino Uno R3 and the ESP8266 NodeMCU using a computer system. The code format recognised by the microcontroller is the "hex" file format.

#### B. The Flow Chart

Figure 4 shows the Flow chart of the real-time monitoring IoT-based weather station system. The system starts just as power is supplied to it. The data values for temperature, pressure, humidity, altitude, and rain status are then measured with the aid of DHT11, BMEP280, and raindrop sensors. The measured values are then converted from analog to digital signals by the DHT11 and the microcontroller, which has an inbuilt Analog to Digital Converter (ADC). The received signals are then calibrated as specified by the written codes and stored in the microcontroller's flash memory. The needed results, which are temperature values measured in degrees Celsius, pressure values measured in Pascal, humidity values presented in percentage, altitude measured in meters, and rain status presented as either being "raining" or "no rain," are then sent over to the LCD for visualisation at the base station as well as to the cloud via the ESP8266 standalone Wi-Fi

module. The above process continues until the system is Reset or shut down by removing the power supply to it.

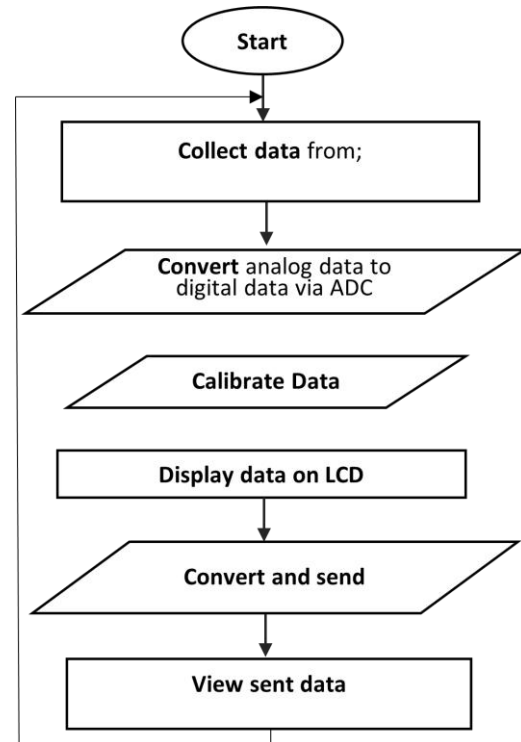


Figure 4: Flow Chart of the Real-Time Monitoring IoT-Based Weather Station.

#### IV. RESULTS AND DISCUSSION

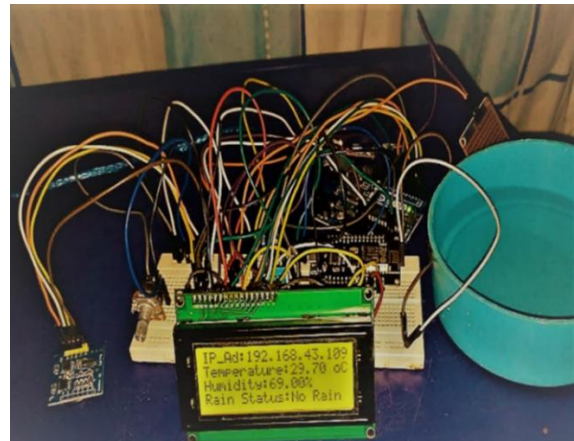
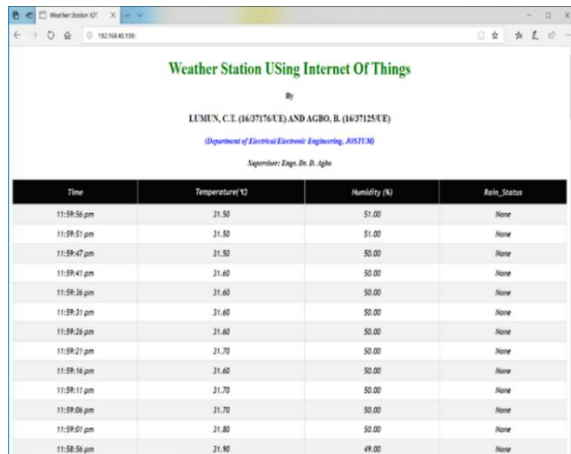


Figure 5: Temperature, Humidity and Rain Status Values Displayed on the LCD.

Figure 5 shows the value of the temperature, humidity and rain status parameters generated by the weather station and displayed on the LCD. Line 1: Shows the Internet IP address of the ESP8266 Node MCU, which is "192.168.43.109.". Line 2: shows the value of the temperature measured in degrees Celsius "°C". Line 3: Shows the value of the humidity

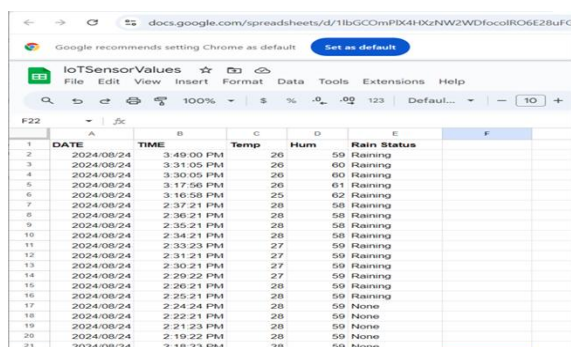
measured in percentage “%”. Line 4: indicates the rain status, which can be either "None" or "Raining."



Time	Temperature(°C)	Humidity (%)	Rain_Status
11:59:56 pm	31.50	51.00	None
11:59:51 pm	31.50	51.00	None
11:59:47 pm	31.50	50.00	None
11:59:41 pm	31.60	50.00	None
11:59:36 pm	31.60	50.00	None
11:59:31 pm	31.60	50.00	None
11:59:26 pm	31.60	50.00	None
11:59:21 pm	31.70	50.00	None
11:59:16 pm	31.60	50.00	None
11:59:11 pm	31.70	50.00	None
11:59:06 pm	31.70	50.00	None
11:59:01 pm	31.80	50.00	None
11:58:56 pm	31.90	49.00	None

Figure 6: Temperature, Humidity and Rain Status Values Displayed on the Local Host Server via the IoT Interface.

Figure 6 shows the values of the temperature, humidity and rain status generated by the weather station and sent to the internet. The results on the web server have been presented in a tabular format, comprising four columns for “Time”, “Temperature”, “Humidity” and “Rain Status”, respectively. Column 1 (Time): Shows the actual time at which the measurement was taken. Column 2 (Temperature): Shows the values of the temperature measured. Column 3 (Humidity): Shows the values of humidity measured. Column 4 (Rain Status): Shows the status of rain in the environment.



	A	B	C	D	E
1	DATE	TIME	Temp	Hum	Rain Status
2	2024/08/24	3:49:00 PM	26	59	Raining
3	2024/08/24	3:31:05 PM	26	60	Raining
4	2024/08/24	3:30:05 PM	26	60	Raining
5	2024/08/24	3:17:56 PM	26	61	Raining
6	2024/08/24	3:16:56 PM	25	62	Raining
7	2024/08/24	2:37:21 PM	28	58	Raining
8	2024/08/24	2:36:21 PM	28	58	Raining
9	2024/08/24	2:35:21 PM	28	58	Raining
10	2024/08/24	2:34:21 PM	28	58	Raining
11	2024/08/24	2:33:23 PM	27	59	Raining
12	2024/08/24	2:31:21 PM	27	59	Raining
13	2024/08/24	2:30:21 PM	27	59	Raining
14	2024/08/24	2:29:22 PM	27	59	Raining
15	2024/08/24	2:26:21 PM	28	59	Raining
16	2024/08/24	2:25:21 PM	28	59	Raining
17	2024/08/24	2:24:24 PM	28	59	None
18	2024/08/24	2:22:21 PM	28	59	None
19	2024/08/24	2:21:23 PM	28	59	None
20	2024/08/24	2:19:22 PM	28	59	None
21	2024/08/24	2:18:23 PM	28	59	None

Figure 7: Screen Shot of Live Streamed Data on Google Sheet.

Figure 7 shows a screenshot of live-streamed data on Google Sheets, while Figure 8 shows a graphical plot of measured temperature and humidity values generated from stored Google Sheets data.

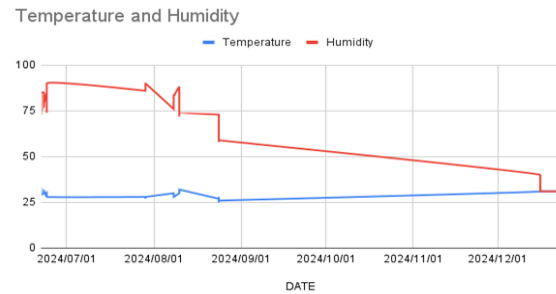


Figure 8: A Graphical Plot of 6-Months Measured Temperature and Humidity Values.

## V. CONCLUSION

The project successfully demonstrated the design and implementation of an Internet of Things (IoT) weather station. The system efficiently collected and transmitted environmental data using specialised sensors, processed by an Arduino Uno board, and communicated over a local area network via the ESP8266 Wi-Fi module. The data visualisation methods explored included a locally hosted webserver and Google Sheets, each with distinct advantages and limitations. The locally hosted web server provided immediate access to data but faced significant drawbacks, including data loss upon page refresh, restricted access to devices within a limited range, and higher costs for expanding accessibility. In contrast, Google Sheets offered a more robust solution by ensuring secure, persistent data storage and universal accessibility. Additionally, it enabled effective data visualisation and analysis, overcoming the limitations of the web server approach. To enhance economic efficiency of the system, it is recommended for future investigation to augment with additional environmental sensors for collecting more comprehensive weather forecasts for further analysis.

## REFERENCES

- [1] Egho-Promise E., Sitti M., Hutchful N. and Agangiba W.A. (2024). “IoT-Enhanced Weather Monitoring System: Affordable Hardware Solution for Real-Time Data Collection, Storage, and Predictive Analysis,” European Journal of Computer Science and Information Technology, 12(1), 43-56.
- [2] Ravi, K. K., & Snehashish, M. (2017). IoT Based Weather Station. Conference Paper, National Institute of Technology, Warangal. 4pp.



- [3] Shahadat, A.S.B., Ayon, S.I. and Khatun, M.R. (2020), "Efficient IoT based Weather Station," International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), pp. 227-230.
- [4] Bulipe, A., Agbo, A., & Ojo, J. (2016). Internet of Things (IoT) Based Weather Monitoring System. *International Journal of Computer Applications*.
- [5] G. A. Akpakwu, B. J. Silva, G. P. Hancke, and A. M. Abu-Mahfouz, "A survey on 5G networks for the Internet of Things: Communication technologies and challenges," *IEEE Access*, vol. 6, pp. 3619–3647, 2018.
- [6] M. R. Palattella *et al.*, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.
- [7] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Comput. Commun.*, vol. 54, no. 12, pp. 1–31, 2014.
- [8] R. Want, B. N. Schilit, and S. Jenson, "Enabling the Internet of Things," *Computer*, vol. 48, no. 1, pp. 28–35, 2015.
- [9] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
- [10] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.
- [11] O. Vermesan *et al.*, "Internet of Things strategic research agenda," in *Internet of Things—Global Technological and Societal Trends*. Gistrup, Denmark: River Publishers, 2011, ch. 2.
- [12] *Service Requirements for Machine-Type Communications*, Sophia-Antipolis Cedex, France, document 3GPP TS 22.368 V11.5.0, 3GPP, 2012.
- [13] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generat. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [14] Neha, J., Yash, S., & Rakesh, M. (2019). IoT and its Applications: A Comprehensive Review. *Journal of Emerging Technologies and Innovative Research*, 6(4), 452–458.
- [15] Jabbar, W. A., Ting, T. M., Hamidun, M. F. I., Kamarudin, A. H. C., Wu, W., Sultan, J., Alsewari, A. A., & Ali, M. A. H. (2024). Development of LoRaWAN-based IoT system for water quality monitoring in rural areas. *Expert Systems with Applications*, 242, 122862.
- [16] Iswanto, Prisma, M., & Brahmantya, A. P. (2020). IoT-Based Weather Station Using Python Interface for Measurement Technique of Educational Purpose. *AIP Conference Proceedings*, 2296, 020027: 1-10.
- [17] Mohamed, F. M. F., & Sudantha, B. H. (2020). Cloud IoT-Powered Smart Weather Station for Microclimate Monitoring. *Indonesian Journal of Electrical Engineering and Computer Science*, 17(1), 508-515.
- [18] Dinkar, M., Reddy, K., & Rao, S. (2017). A Comparative Study of Arduino, Raspberry Pi, and ESP8266 as IoT Development Boards. *International Journal of Computer Applications*.
- [19] Vivek Babu, K., Anudeep, R., Vidyapathi, C. M., & Karthiyekan, B. (2017). Weather Forecasting Using Raspberry Pi with Internet of Things (IoT). *ARPJ Journal of Engineering and Applied Science*, 12(17), 5129-5134
- [20] Bruce, S. (2015). The Internet of Things: A Reality Check. *ACM SIGCOMM Computer Communication Review*.