



Development of an IoT-Based Temperature and Humidity Monitoring System with MySQL Query Optimization for Efficient Data Storage and Access

Dewa Ayu Indah Cahya Dewi ¹, I Ketut Swardika ²,
Putri Alit Widyastuti Santiary ³

^{1,2,3} Electrical Engineering Department, Politeknik Negeri Bali, Bali, Indonesia
ayuindahcahyadewi@pnb.ac.id

Abstract. Environmental monitoring systems are crucial in smart and sustainable infrastructure, particularly in industrial settings where temperature and humidity must be continuously monitored in real-time. This research presents the design and implementation of an Internet of Things (IoT)-based monitoring system that utilizes the ESP32 microcontroller paired with a DHT22 sensor to collect temperature and humidity data. The collected data are transmitted to a MySQL database server through the HTTP protocol and visualized on a responsive web-based dashboard. As the volume of sensor data grows over time, efficient data management becomes critical. To address performance issues such as slow query response and delayed dashboard loading, the system incorporates two key database optimization strategies: indexing and horizontal partitioning based on timestamps. These optimizations significantly enhance the speed of SQL query execution and reduce page load latency. The findings suggest that integrating IoT with structured database optimization offers a scalable and efficient solution for real-time environmental monitoring in smart industry applications.

Keywords: Indexing, Internet of Things (IoT), MySQL, Partitioning, Web

1 Introduction

In the era of Industry 4.0, environmental monitoring plays a critical role in smart infrastructure systems, particularly in ensuring the safety and sustainability of indoor environments. The integration of Internet of Things (IoT) technologies into environmental sensing allows real-time data acquisition, remote access, and improved responsiveness for decision-making processes (Abidin et al., 2022; Tsani & Subardono, 2019). Prior research has demonstrated the effectiveness of IoT-based systems in monitoring environmental parameters such as temperature and humidity using low-cost sensors and microcontrollers (Agbulu & Kumar, 2021; Narayana et al., 2024; Putri et al., 2019). Previous studies have utilized the DHT22 sensor to measure temperature and humidity in specific environments, such as organic waste bins. The data collected by

© The Author(s) 2025

A. A. N. G. Sapteka et al. (eds.), *Proceedings of the International Conference on Sustainable Green Tourism Applied Science - Engineering Applied Science 2025 (ICOSTAS-EAS 2025)*, Advances in Engineering Research 280,

https://doi.org/10.2991/978-94-6463-878-3_6

the sensor is processed by an ESP32 microcontroller and transmitted to a PHP and MySQL server using HTTP over a WiFi connection.

The recorded information is then displayed in real time on a web-based dashboard, allowing users to monitor environmental conditions remotely. This approach aims to provide an early warning system to prevent potential hazards caused by rising temperatures, humidity levels, or the accumulation of harmful gases (Novantri & Oktiawati, 2022). A separate study developed a real-time environmental and motion monitoring system for server rooms using the Thingier.io application. The system supports remote tracking of temperature, humidity, and object movement, allowing for early detection of environmental fluctuations and potential intrusions (Bakri et al., 2022). The research implemented a web-based IoT Smart Farm monitoring system using the Laravel 10 framework, PHP, and MySQL for backend development, while employing a Bootstrap-based dashboard template to provide an intuitive and responsive user interface for real-time monitoring and data visualization (Al-Adhim & Dewi, 2024). A critical limitation in many existing IoT monitoring systems is the inefficiency in handling large-scale time-series data, particularly in MySQL-based architectures. As data accumulates over time, unoptimized queries can lead to slow response times, increased memory consumption, and delayed user interface rendering. This issue is particularly pronounced in dashboards or web-based visualizations that must retrieve and display recent data in real-time. To address these challenges, this study proposes an enhanced IoT-based monitoring system that not only collects temperature and humidity data using ESP32 and DHT22 sensors but also integrates MySQL query optimization techniques, namely indexing and horizontal partitioning. By focusing on backend optimization, the system aims to improve query performance, reduce dashboard loading time, and enhance overall user experience. The contribution of this research is the implementation of database indexing and partitioning to reduce SQL query execution time and the evaluation of performance metrics, including execution time, page rendering time, and data retrieval efficiency, before and after optimization.

2 Methodology

The monitoring system developed in this study is centered around an ESP32 microcontroller, which is integrated with a DHT22 sensor to capture real-time environmental data, specifically temperature and humidity (Firmansyah, 2019). The ESP32 device, programmed using the Arduino IDE, reads the sensor values at regular intervals and transmits the data to a remote MySQL server via HTTP protocol. Each data record is stored in a dedicated MySQL table named `station1`, which contains four key fields: an auto-incremented ID, temperature, humidity, and a timestamp field (`created_at`) that logs the exact time of data entry. As the system collects data continuously, performance optimization becomes crucial, particularly for query execution and dashboard rendering. This research implemented two core optimization techniques on the database level. Firstly, indexing was applied to the `created_at` column. This technique enables the MySQL engine to quickly locate specific records that fall within a defined time range, thereby significantly accelerating query execution times, particularly for time-based searches. Secondly, we applied horizontal

partitioning on the table by introducing an additional column named `created_date`, which stores the date in YYYY-MM-DD format extracted from `created_at`. Using MySQL's RANGE partitioning feature, the table is logically divided into multiple partitions based on date intervals (e.g., by month or year). This ensures that when a query requests data from a specific time period, only the relevant partitions are scanned, thus reducing the amount of data processed and improving system efficiency. To evaluate system performance before and after optimization, we utilized a set of metrics. The SQL Query Execution Time was measured in seconds using PHP's built-in `microtime` function to precisely calculate the time needed to run a standard SELECT query on the database. Additionally, page load metrics such as `DOMContentLoaded` and total load time were recorded using browser developer tools to assess the real-time responsiveness of the web-based dashboard interface. This research compares the performance of two configurations. The baseline setup without indexing and partitioning, where the entire table is scanned for each query. The optimized setup with both indexing and horizontal partitioning, designed to limit data scanning and speed up response times.

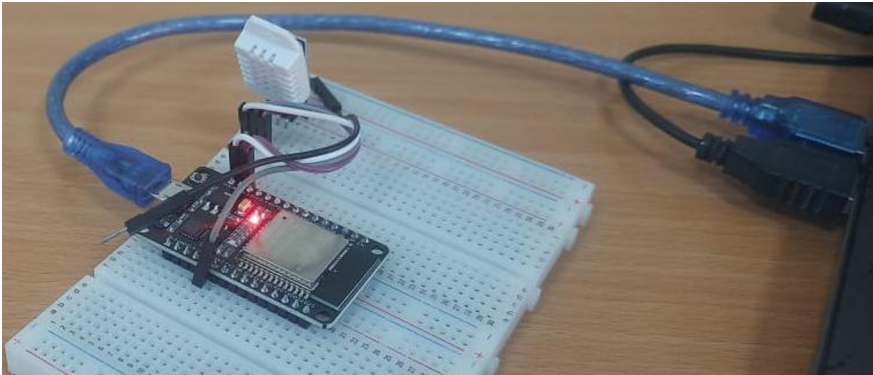


Figure 1. Assembled Circuit

The wiring diagram illustrates how the DHT22 sensor is connected to the ESP32 microcontroller to enable temperature and humidity sensing. The DHT22 sensor has three main pins: VCC (power supply), DATA (signal), and GND (ground). In this setup, the VCC pin of the DHT22 is connected to the VIN pin of the ESP32, which provides a stable 3.3V to 5V power supply required by the sensor. The DATA pin is connected to GPIO4 on the ESP32, which is used as the digital input to read temperature and humidity data. The GND pin of the sensor is connected to one of the GND pins on the ESP32, establishing a common ground between the devices. Optionally, jumper wires are color-coded for clarity: red for VCC, yellow or orange for DATA, and black for GND. This simple three-wire configuration allows efficient communication between the ESP32 and the DHT22 sensor using the DHT library in Arduino. The wiring diagram table can be seen in Table 1.

Table 1. Wiring Diagram ESP32 to DHT22

DHT22 pin	ESP32 pin	Function
VCC	VIN	Power supply (3.3V/5V)
DATA	GPIO4	Temperature and humidity data
GND	GND	Ground

The research flow can be seen in Figure 2.

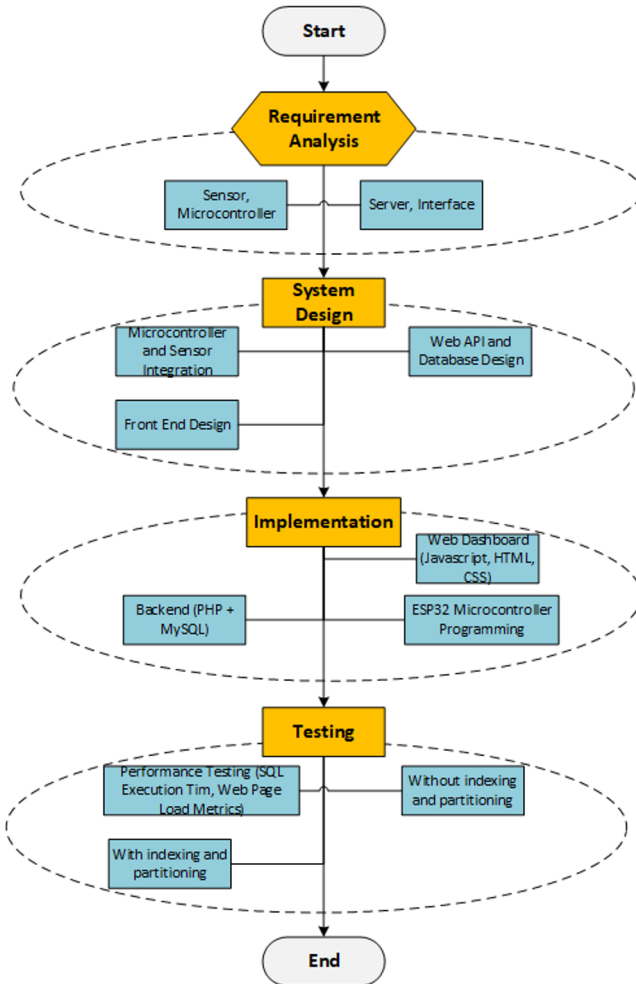


Figure 2. Research Flow

3 Result and Discussion

3.1 Result

The developed monitoring system presents real-time temperature and humidity data through a responsive web-based dashboard. The website interface is built using PHP and integrated with a MySQL database that stores sensor readings collected from the ESP32 microcontroller, which is connected to a DHT22 sensor. The dashboard displays two interactive line charts generated with Chart.js, one for temperature (in °C) and another for humidity (in %), both plotted against timestamps to show environmental changes over time. In addition to the graphical representation, the dashboard also includes a tabular view listing the latest 50 sensor readings with corresponding timestamps, values, and an option to delete individual records. This dual-mode visualization enables both quick overviews and detailed analysis of historical environmental data. The frontend design employs Bootstrap to ensure mobile responsiveness and an intuitive user experience. This web-based system facilitates continuous environmental monitoring and can be accessed remotely, making it suitable for applications in smart rooms, greenhouses, or industrial environments (Sari et al., 2022). The architecture of this system can be seen in Figure 3.

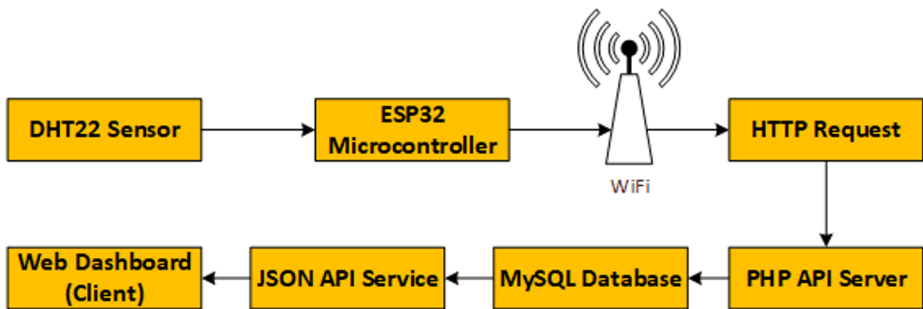


Figure 3. IoT System Architecture for Temperature and Humidity Monitoring

Figure 3 illustrates the architecture of an IoT-based system designed to monitor temperature and humidity. The system uses a DHT22 sensor connected to an ESP32 microcontroller, which transmits the sensor data via WiFi to a PHP API server. The server stores the data in a MySQL database and provides it in JSON format to a web-based dashboard, allowing users to visualize the data in real-time. The Arduino code can be seen in Figure 4.

```
Arduino IDE 2.2.1
DOIT ESP32 DEVKIT V1
suhu_esp32.ino
35 void loop() {
36   float kelembaban = dht.readHumidity();
37   float suhu = dht.readTemperature();
38
39   if (isnan(kelembaban) || isnan(suhu)) {
40     Serial.println("❌ Gagal membaca data dari sensor DHT!");
41     delay(5000);
42     return;
43   }
44
45   Serial.printf("🌡️ Suhu: %.2f °C | 💧 Kelembaban: %.2f%%\n", suhu, kelembaban);
46
47   if (WiFi.status() == WL_CONNECTED) {
48     HTTPClient http;
49     WiFiClient client;
50
51     String url = String(serverUrl) + "?suhu=" + String(suhu, 2) + "&kelembaban=" + String(kelembaban, 2);
52     Serial.println("📡 Mengirim data ke: " + url);
53   }
}

Output Serial Monitor x
Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM11')
📡 Mengirim data ke: http://192.168.1.6/suhu/simpan_data.php?suhu=27.00&kelembaban=79.90
📄 Respons server: Data berhasil disimpan
🌡️ Suhu: 27.00 °C | 💧 Kelembaban: 80.00 %
📡 Mengirim data ke: http://192.168.1.6/suhu/simpan_data.php?suhu=27.00&kelembaban=80.00
📄 Respons server: Data berhasil disimpan
🌡️ Suhu: 26.90 °C | 💧 Kelembaban: 80.00 %
📡 Mengirim data ke: http://192.168.1.6/suhu/simpan_data.php?suhu=26.90&kelembaban=80.00
📄 Respons server: Data berhasil disimpan
```

Figure 4. ESP32 Code for IoT Data Transmission Over WiFi

Figure 4 shows the Arduino code implemented on the ESP32 microcontroller to read temperature and humidity data from the DHT22 sensor. The code connects to a WiFi network and sends the collected data to a web server using the HTTP protocol. This enables real-time data logging into a MySQL database for further monitoring and analysis. The display of temperature and humidity data can be seen in Figure 5.

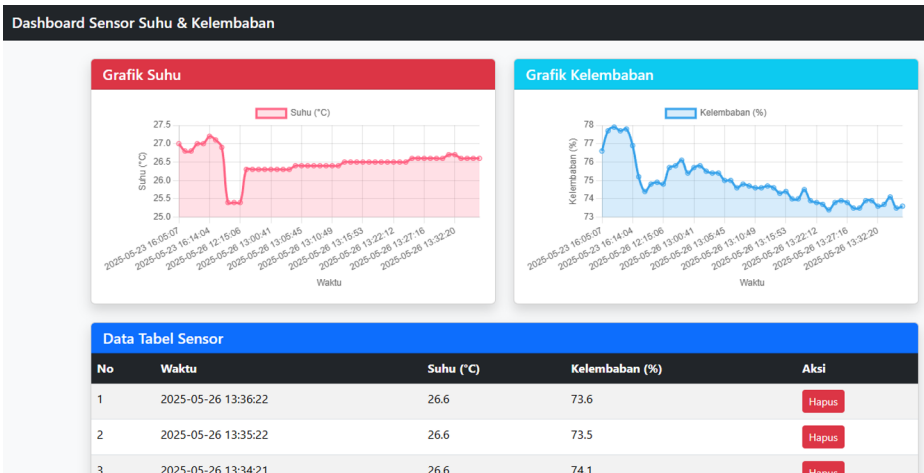


Figure 5. Web-Based Dashboard Interface for Real-Time Temperature and Humidity Monitoring Using IoT and MySQL

Figure 5 displays a web-based dashboard interface that visualizes real-time temperature and humidity data collected from an IoT system. The dashboard retrieves data from a MySQL database via a JSON API, enabling users to dynamically and efficiently monitor environmental conditions through graphical representations, such as charts and data tables. The value of the page load time is shown in Figure 6.

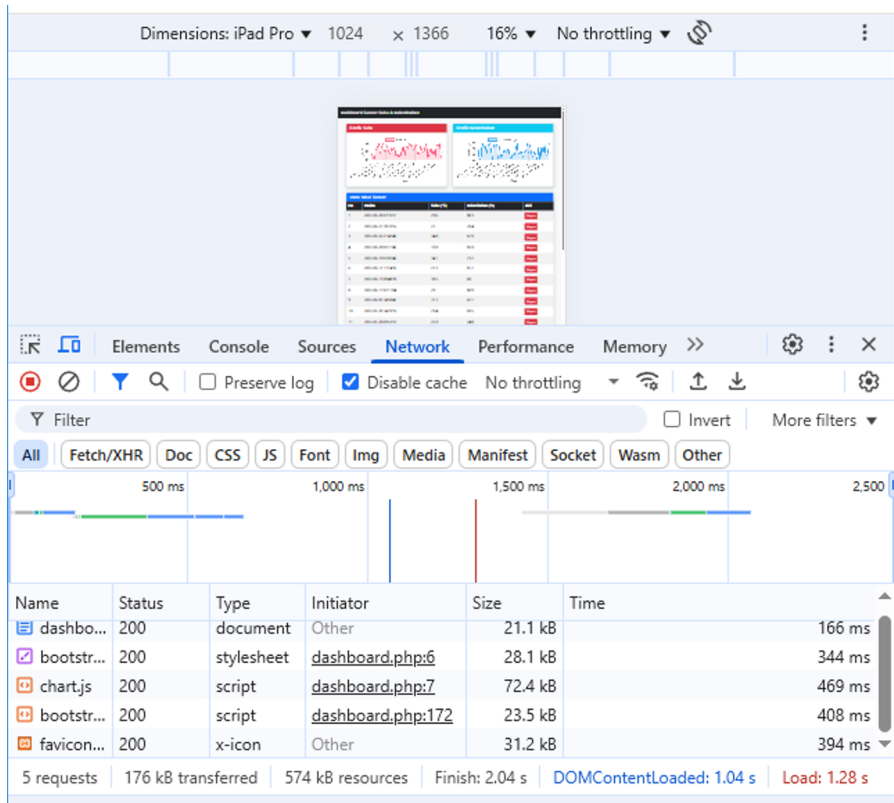


Figure 6. Page Load Time Measurement After Applying Index and Partition on the Sensor Data Table

The Performance Improvement Percentage, or Enhancement (%), is a metric used to quantify the relative improvement achieved after applying an optimization technique. It compares the performance of a system before and after optimization, using a proportional difference relative to the initial (before) state. This formula is commonly applied in performance analysis, benchmarking, and efficiency evaluation. It can be seen in Equation 1.

$$\text{Enhancement (\%)} = \frac{\text{Without index partition} - \text{With index partition}}{\text{Without index partition}} \times 100\% \quad (1)$$

A total of 1,000 data records of temperature and humidity readings were successfully collected during the monitoring process. These readings were captured in real time using a DHT22 sensor connected to an ESP32 microcontroller. Each reading was transmitted periodically via WiFi using the HTTP protocol and stored in a MySQL database. This dataset served as the basis for analyzing system performance, including evaluating the effectiveness of indexing and partitioning techniques in improving SQL query execution time and web dashboard loading responsiveness. The results are summarized below in Table 2.

Table 2. Performance Comparison Between Baseline and Optimized Configurations

Metric	Without index and partition	With index and partition	Performance enhancement (%)
SQL Execution Time (sec)	0.03916	0.00298	92.39%
DOMContentLoaded (sec)	2.45200	1.33267	45.65%
Page Load Time (sec)	2.44800	1.58333	35.32%

Table 2 presents a comparative analysis of system performance between the two configurations. One without optimization (no indexing or partitioning) and another with optimization (indexing on the timestamp column and horizontal partitioning based on date ranges). The evaluation was conducted using three performance metrics: SQL query execution time, DOMContentLoaded, and total page load time.

3.2 Discussion

The implementation of indexing and partitioning led to faster query execution and improved dashboard performance. This is particularly beneficial for systems with increasing data volume, as it reduces the amount of scanned rows and system workload. Additionally, visual dashboards are rendered more quickly, thereby enhancing the user experience in real-time monitoring scenarios. The SQL Execution Time reflects how long the server takes to execute a time-based SELECT query on the sensor data table. Without optimization, the average execution time was 0.03916 seconds. After applying indexing and partitioning techniques, the execution time decreased to 0.00298 seconds, resulting in a 92.39% improvement in query performance. The DOMContentLoaded metric, which measures how quickly the initial HTML content loads in the browser, was reduced from 2.45 seconds to 1.33 seconds. This indicates that faster query response times contributed to improved front-end rendering, yielding a 45.65% enhancement.

Lastly, the page load time, representing the total time required for the full dashboard to load (including charts and tables), improved from 2.44 seconds to 1.58 seconds. This equates to a 35.32% improvement in overall user interface responsiveness. These results clearly demonstrate that the use of indexing and partitioning in the MySQL database significantly enhances both backend data access efficiency and frontend dashboard performance, particularly for systems handling large volumes of time-series sensor data. Although the developed system successfully integrates IoT-based temperature and humidity monitoring with optimized MySQL data storage using

indexing and partitioning, several limitations were identified during implementation and testing. The system currently relies on a single type of sensor (DHT22), which, while cost-effective and widely used, has limited accuracy and response time compared to more advanced industrial-grade sensors. This may impact the reliability of the data in critical environments requiring high precision. The system was tested on a small-scale dataset with 1,000 entries. The scalability of the indexing and partitioning strategy for larger datasets with millions of records has not yet been validated, especially under concurrent data access conditions. The data is only used for monitoring and visualization purposes, without being integrated into automated control systems. The absence of real-time alert mechanisms or actuators such as fans, alarms, or relays. The system's performance was only evaluated under stable network conditions. Its robustness under poor connectivity, data loss, or high-latency scenarios remains unexplored.

4 Conclusion

This study presents the development of an IoT-based monitoring system for temperature and humidity using ESP32 and DHT22 sensors, with data transmission to a MySQL server via the HTTP protocol. To address the performance limitations commonly encountered in time-series data processing, the system was enhanced with database indexing and horizontal partitioning techniques applied to the timestamp column. The implementation of these optimizations led to noticeable improvements in system performance, both at the query execution level and in the overall responsiveness of the web dashboard. The optimized system demonstrated faster data retrieval and smoother user interface loading compared to the non-optimized version. These results confirm the effectiveness of combining IoT technology with structured database optimization to support real-time environmental monitoring. The approach is particularly relevant for applications that require scalable and efficient handling of continuous sensor data. The limitations of this study include the use of only one type of sensor (DHT22), a limited data scale, and the absence of system testing under extreme environmental conditions such as high temperatures or extreme humidity. Future development can explore the use of distributed databases and cloud-based analytics to further enhance system capabilities. Additionally, future research can focus on enhancing the system to support practical implementations, such as triggering automatic alarms when temperature or humidity exceeds predefined thresholds, and integrating with automated control of fans or heaters in storage rooms and greenhouses.

Acknowledgment

The author expresses sincere gratitude to the Research and Community Service Center (P3M) of Politeknik Negeri Bali for providing research funding as stipulated in Agreement Letter No. 04328/PL8/AL.04/2025 dated April 8, 2025, within the 2025 institutional budget. Appreciation is also extended to the lecturers and students of the Automation Engineering Study Program, Department of Electrical Engineering, for their valuable support throughout the implementation of this research.

References

- Abidin, Z., Nadhif, M., & Arif, M. (2022). Desain prototype alat kontrol serta deteksi suhu dan kelembaban kandang ayam broiler dengan metode Fuzzy berbasis IoT. *ELECTRA : Electrical Engineering Articles*, 3(01). <https://doi.org/10.25273/electra.v3i01.13976>.
- Agbulu, G. P., & Kumar, G. J. R. (2021). An ultra-low power IoT system for indoor air quality monitoring. *Journal of Physics: Conference Series*, 2007(1). <https://doi.org/10.1088/1742-6596/2007/1/012053>.
- Al-Adhim, M., & Dewi, G. (2024). Sistem monitoring IoT Smart Farm berbasis web dengan integrasi template dashboard Bootstrap dan Laravel 10. *COMSERVA : Jurnal Penelitian dan Pengabdian Masyarakat*, 4, 1973–1981. <https://doi.org/10.59141/comserva.v4i7.2595>.
- Bakri, M. A., Farhan, M., Sujatmiko, A., & Firasanti, A. (2022). Pemantauan suhu dan deteksi gerak obyek berbasis IoT pada ruang server menggunakan Thingier.IO. *TELKA-Telekomunikasi Elektronika Komputasi dan Kontrol*, 8(1). <https://doi.org/10.15575/telka.v8n1.74-81>.
- Firmansyah, V. (2019). Implementasi IoT system pada perekaman data sensor suhu DHT22 dengan web service native PHP untuk pemantauan kondisi ruangan laboratorium. *Insan Metrologi*, 2(4).
- Narayana, T. L., Venkatesh, C., Kiran, A., J. C. B., Kumar, A., Khan, S. B., Almusharraf, A., & Quasim, M. T. (2024). Advances in real-time smart monitoring of environmental parameters using IoT and sensors. *Heliyon*, 10(7). <https://doi.org/10.1016/j.heliyon.2024.e28195>.
- Novantri, S. O., & Oktiwati, U. Y. (2022). Rancang bangun pemantauan kadar gas metana pada pengolahan sampah organik berbasis IoT menggunakan microcontroller ESP32. *Jurnal Listrik Instrumentasi Elektronika Terapan*, 3(2).
- Putri, A. R., Suroso, & Nasron. (2019). Perancangan alat penyiram tanaman otomatis pada miniatur greenhouse berbasis IoT. *Seminar Nasional Inovasi dan Aplikasi Teknologi di Industri 2019*, 5.
- Sari, I. P., Batubara, I. H., Basri, M., & Hazidar, A. H. (2022). Implementasi internet of things berbasis website dalam pemesanan jasa rumah service teknisi komputer dan jaringan komputer. *Blend Sains Jurnal Teknik*, 1(2). <https://doi.org/10.56211/blendsains.v1i2.136>.
- Tsani, Y. R., & Subardono, A. (2019). Data logging implementation on web-based communication on Arduino devices. *Jurnal Online Informatika*, 3(2). <https://doi.org/10.15575/join.v3i2.252>.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

