

AN AUTOMATED DATA LOGGER SYSTEM FOR REAL-TIME MONITORING AND ANOMALY DETECTION IN INDUSTRIAL IOT ENVIRONMENT

N. Harum^{1*}, N.A. Emran¹, M.H.F. Md Fauadi², E. Hamid¹,
M.N.M. Khambari¹, M.M. Ridzuan³, and M. Kchouri⁴

¹Fakulti Teknologi Maklumat dan Komunikasi,
Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian
Tunggal, Melaka, Malaysia.

²Fakulti Kecerdasan Buatan Dan Keselamatan Siber
Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian
Tunggal, Melaka, Malaysia.

²Product Development Division, Konica Minolta Business Technologies (M),
Sdn. Bhd, Ayer Keroh, Melaka, Malaysia.

³Department of Computer Science,
Modern University of Business and Science University,
113-7501 Beirut, Lebanon.

*Corresponding Author's Email: norharyati@utem.edu.my

Article History: Received 18 June 2024; Revised 15 November 2024;
Accepted 2 December 2024

©2024 N. Harum et al. Published by Penerbit Universiti Teknikal Malaysia Melaka. This is an open article under the CC-BY-NC-ND license (<https://creativecommons.org/licenses/by-nc-nd/4.0/>).

ABSTRACT: In Industrial Internet-of-Things, a data logger must possess critical features such as real-time data acquisition, scalable storage capabilities, robust anomaly detection, and efficient dashboard integration for user-friendly monitoring, ensuring comprehensive data management and system reliability across industrial environments. Nevertheless, current data loggers offer very little data storage, have few intelligent features, and frequently have an interface that is difficult to use. Additionally, these loggers struggle with efficient data management, leading to storage issues and poor user experience. The integration of Industrial Internet of Things technology facilitates efficient mass data collection by enabling seamless connectivity and real-time

monitoring. In this work, a system that features a user-friendly dashboard, enhanced with Grafana for advanced data visualization and management, built on Node-RED for flexible and streamlined development was proposed. A Raspberry Pi was chosen as a gateway due to its capability to process real-time data and send the data to the database. The system is capable of reading data from multiple sensors, which is stored in InfluxDB, a reliable time-series database. Moreover, the dashboard supports factory workflow and environmental monitoring from any location. The system also alerts users when an anomaly is detected, enabling proactive management and timely response. The anomaly message was sent directly from Raspberry Pi to reduce processing time, as demonstrated in the performance test results. The developed product underwent user evaluation, scoring grade A in usability testing with an impressive score of 91.25%, indicating a high level of user satisfaction and effectiveness.

KEYWORDS: *Data logger; IIoT; Smart Factory; Automation*

1.0 INTRODUCTION

A data logger plays a crucial role in smart applications by continuously collecting, processing, and storing real-time data from various sensors, enabling efficient monitoring, decision-making, and automation in systems like smart agriculture, smart homes, and industrial IoT environments [1][2]. This real-time data is essential for optimizing performance, detecting anomalies, and enhancing system efficiency in a variety of smart applications. The data logger is widely utilized in smart factory environments, playing a crucial role in ensuring workplace safety, optimizing mass production, and promoting energy efficiency through various intelligent applications. In Industrial Internet of Things (IIoT) applications, data loggers play a critical role by continuously capturing and storing sensor data, deemed essential for monitoring equipment performance and detecting anomalies [3]. This real-time data acquisition supports predictive maintenance and operational efficiency improvements by providing actionable insights from historical and current data [4].

However, industries frequently lack comprehensive data logging systems because the existing loggers offer only limited functionalities. Without incorporating intelligent features, these systems fail to meet the advanced requirements for effective industrial monitoring and data analysis. This data can be used for predictive maintenance, efficiency

improvements, and compliance adherence to industry regulations. There are works that present data logger [5]-[14] to meet the IIoT expectation. However, multiple problems such as limited data storage [4]-[8] and low-portability connection [12] for the system to communicate have remained unsolved as shown in Table 1.

Table 1: Functional comparison of existing data loggers and the proposed data logger.

Microcontroller/ microprocessor	Communication media	Storage	User Monitorin g	Anomaly Detection	Reference
NodeMCU	Wi-Fi	Database	-	-	[5]
NodeMCU	Wi-Fi	MicroSD	-	-	[6]
Microcontroller	WiFi	MicroSD	LCD Screen	-	[7]
NodeMCU	WiFi	MicroSD	ThinkSpea k	-	[8]
Arduino Mega	Wired	MicroSD	-	-	[9]
Raspberry Pi	Wi-Fi	MicroSD	-	-	[10]
Not Stated	Wi-Fi	Database	-	-	[11]
Not Stated	Wired	-	-	-	[12]
Raspberry Pi	Wi-Fi	Database	Grafana Dashboar d	Telegram	Proposed

The table presents a comparison of various microcontrollers and microprocessors used in existing data logging systems. The table also focuses on data logger features in terms of communication media, storage, user monitoring, and anomaly detection capabilities. Arduino Uno and NodeMCU are commonly used with Wi-Fi and either databases or MicroSD for storage, but they lack user monitoring and anomaly detection features. Some setups incorporate an LCD screen for local display, while others use ThinkSpeak for basic monitoring. The lack of anomaly detection and notification is unacceptable, as it can lead to a waste of company resources.

The proposed system using Raspberry Pi is highlighted for its superior

features, including Wi-Fi communication, expandable database storage, and enhanced user interaction through a Grafana dashboard and Telegram for real-time anomaly detection. This system offers a more robust and comprehensive solution compared to others that lack these advanced functionalities. For anomaly detection messages, they are sent directly to the user to minimize the processing time required for message retrieval purposes.

2.0 METHODOLOGY

This paper proposes Data Logger for IIoT System to overcome the weaknesses and problems of existing projects. As provided in Table 1 and Figure 1, the proposed data logger was designed to improve functionality of the existing data logger by adding expandable data storage, high capability microprocessor, user monitoring and anomaly detection. This project developed a data logger by connecting multiple sensors directly to the Raspberry Pi 4 microprocessor. The sensors captured data from the facility environment based on their types, and the microprocessor automatically uploaded and, subsequently, recorded the received data into the database. The real time data was to be viewed by users via a dashboard and notifications were sent to users during anomaly cases. The recorded data was also stored in databases to be used for future forensic purposes.

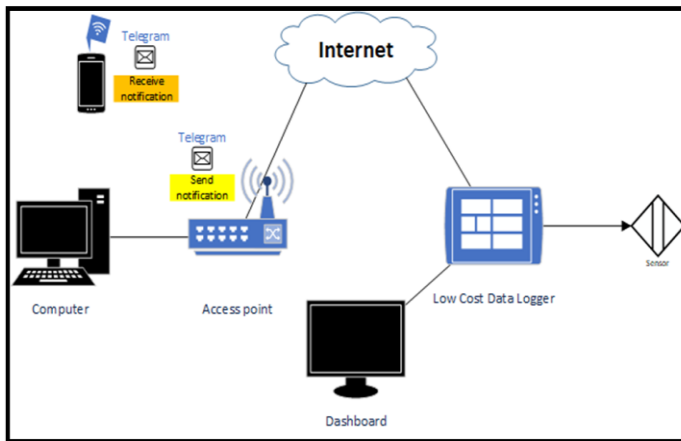

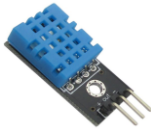




Figure 1: The architecture of the proposed data logger

This project methodology comprised 3 phases: Requirements analysis, design and implementation, testing and validation. Requirements analysis refers to gathering and defining functional and non-functional

needs of a system to ensure that the project aligns with stakeholders' goals. Design and Implementation refers to the process of creating a detailed blueprint for the system architecture and translating that design into actual code or hardware that performs the required functions. Testing and Validation involve verifying that the system works as intended by detecting and fixing errors, ensuring that the final product meets all the specified requirements and performs reliably in real-world conditions.

Table 2: Component requirements

Hardware/Software	Functionality	Picture
Raspberry Pi 4 Model B	Used as microprocessor to process detected data from sensors. The process includes: <ul style="list-style-type: none"> Anomaly detection Storing the data into database 	
DHT11 Temperature and humidity sensor	To detect temperature and humidity in the surroundings. <ul style="list-style-type: none"> Temperature Range: 0 to 50°C with ±2°C accuracy. Humidity Range: 20% to 90% relative humidity with ±5% accuracy 	
MQ-2 Gas Sensor	To detect any gas concentration in the air including methane (CH ₄), Propane (C ₃ H ₈), Butane (C ₄ H ₁₀), Hydrogen (H ₂), LPG, and other combustible gases.	
MCP3008 AC to DC Converter	Converting analog data into digital data. It can read data from analog environmental sensors, such as temperature, humidity, gas, and light sensors, converting their signals into digital data for processing in smart devices or monitoring systems.	

2.1 Phase 1: Requirement gathering

In this section, there are three types of requirements analysis as follows: Data requirements, functional requirements, hardware and software requirements.

The data and functionality requirements had been determined by several interview sessions with an industrial expert in a factory. Several visits were made to Konica Minolta factory to investigate the real scenario and requirements for data loggers in a factory (Figure 2).



Figure 2: Discussion and interview session at Konica Minolta.

Based on the findings from the visit, it was determined that the dashboard integration, anomaly detection, expandable storage, and data history were essential functionalities that must be incorporated into a data logger design to ensure comprehensive monitoring and analysis. Based on the functions, hardware or components requirements have been identified as shown in Table 2. For software, the following items were selected:

- Grafana – a dashboard to display data collected from the sensor to user. It is a multi-platform open-source analytics and has an interactive web application interface.
- InfluxDB - InfluxDB is an open-source time series database used to save data detected by the sensor. It is one of the best and most reliable databases to save data.
- Node RED - To connect all the hardware devices used in this project, Node-Red APIs are used for programming. In this project, Node-Red was to create a configuration, develop, and create multiple extensions to connect with the data collected from the sensor.
- Telegram – Simple chat platform that can control the system and receive anomaly notification. It also allows multiple users to control the system.

2.2 Phase 2: Design and Implementation

In phase 2, design, configuration and integration between data logger with user system (dashboard and notification) is discussed. Figure 3 shows a process flowchart for the developed data logger. The data logger acquired data from the sensor and subsequently transmitted it to the system for processing. All collected data, including the associated

date and time, was stored in a database, InfluxDB. Despite any data exceeding preset limits, it was retained in the database. If the data surpassed the maximum threshold established by the system, an alert notification was dispatched to user via Telegram application. This notification included details of the sensor readings that exceeded the set limit. The data that exceeded the threshold limit needed to be processed before storing the value into the database. This can reduce processing time because the data will be processed without database waiting time [13]. During the anomaly case detection, the notification would increase monitoring efficiency. Hence, eliminating the user requirement to be present 24 hours to monitor the system. The user presence was crucial to detect any incident that required any immediate action. For dashboard view, once the data is stored in the database, it will be retrieved by the system and presented to the user through a web-based interface, using Grafana as presented in Figure 4. This interface allowed users to view detailed information about the collection time and sensor readings.

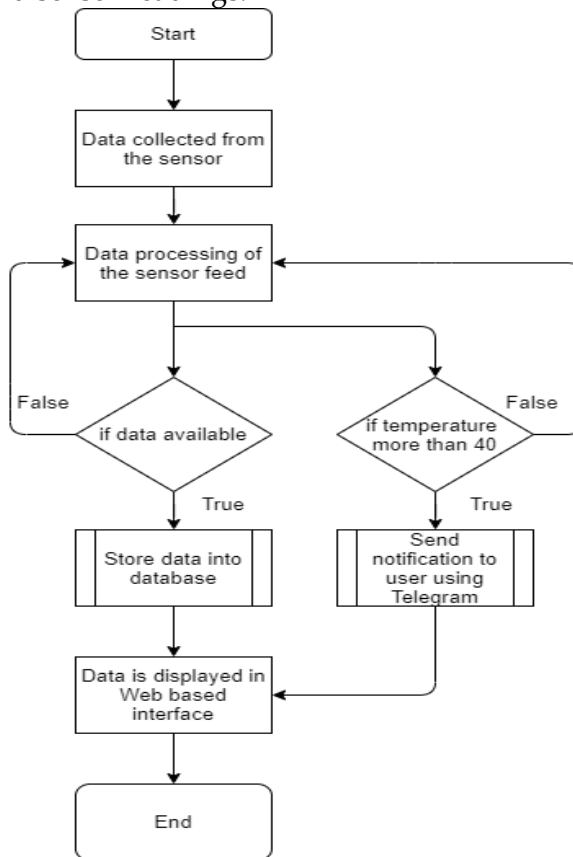


Figure 3: Block Diagram for the developed prototype

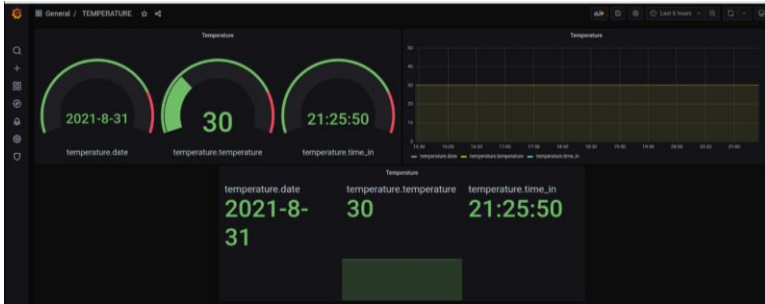


Figure 4: Grafana Dashboard

Since the data logger often collected data from several sensors, the developed prototype was simulated using two different sensors. For the prototype, the DHT11 sensor was to collect temperature and humidity data, whereas the MQ-2 Gas sensor gathered gas data, including LPG, smoke, alcohol, propane, hydrogen, methane, and carbon monoxide. Both sensors were connected to the Raspberry Pi microprocessor via jumper wires. Figure 5 displays the developed prototype, and Table 2 outlines the hardware and GPIO pin connections.

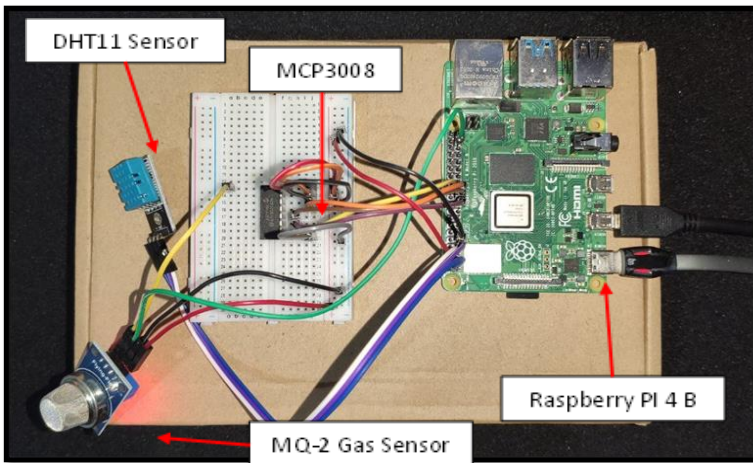


Figure 5: Prototype of IoT Integrated Data Logger System

Table 3: Hardware vs. GPIO pin

Hardware	Wire	Pins
----------	------	------

DHT11Sensor	OUT	GPIO 4
MQ-2 Gas Sensor	AO DO	CH0 GPIO 26
MCP8002 AC to DC Converter	CH0 ABOUT DIN CS/SHDN	AO GPIO 9 MISO (SPI0) GPIO 10 MOSI (SPI0) GPIO 25

The integration between the developed data logger with user dashboard and Telegram is done by using Node RED as presented in Figure 6, consists of the following parts:

- a. Input data from sensor
- b. Input processing of temperature, humidity
- c. Input processing of gas data
- d. Threshold setup for anomaly detection and notification

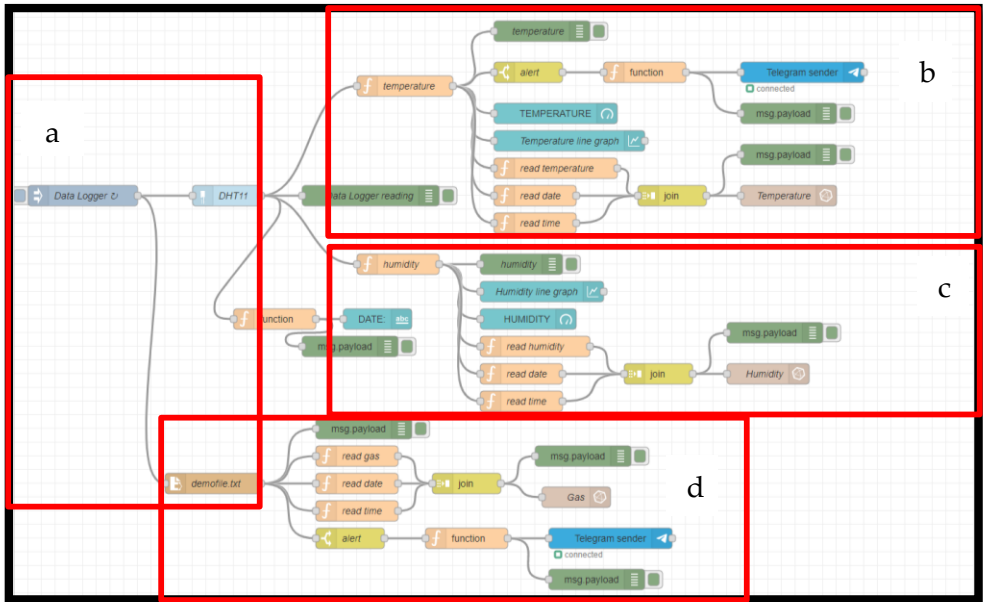


Figure 6: Integration between data logger and user interface using Node RED

2.3 Phase 3: Testing and Validation

The testing and validation are done and shown in Figure 7, based on functionality testing, performance testing and usability testing, elaborated in Section 3.



Figure 7: Testing and Validation process in Konica Minolta

3.0 RESULT AND DISCUSSION

Three types of testing were performed to validate the performance of the developed data logger: functionality, performance and usability tests. Functionality testing, performance testing, and usability testing were deemed critical for the prototype development and system validation process, each serving distinct purpose.

Functionality testing focuses on verifying that the prototype behaves according to its specifications and correctly performs the expected tasks under normal conditions. Performance testing, on the other hand, assesses the processing time used to detect, process and transfer the data to dashboard and notification system. Low processing time is required to ensure the data logger can play its role when anomaly case is detected, and notification sent to user platform. The usability testing examines user experience, ensuring the developed system is intuitive, user-friendly, and meets the needs of data monitoring workers, often involving real users to evaluate ease of navigation and overall satisfaction. Together, these tests ensure that a system is not only functional and efficient but also accessible and easy to use.

3.1 Functionality Test

A functionality test was conducted to ensure the developed prototype was fully integrated among three modules: sensors detection, database, notification and dashboard modules. Figure 8 shows functionality test results for temperature and humidity sensors. A message payload showing temperature and humidity level is presented in the figure.

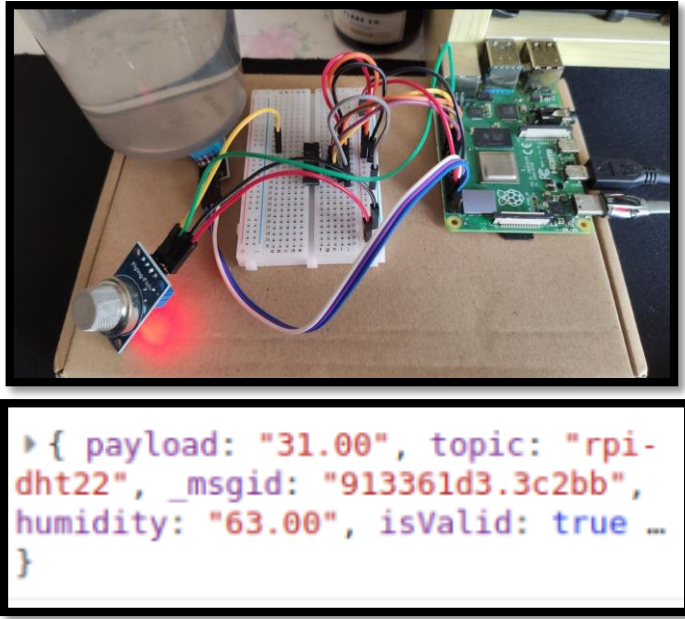


Figure 8: Sensors Functionality Testing

Figure 8 shows that detected data is successfully recorded into the database. The detected temperature and time were successfully recorded into the database table as shown in Figure 9.

name: temperature	time	date	temperature	time_in
1624470759378560353	2021-6-24	30.00	1:52:39	
1624470798884493021	2021-6-24	30.00	1:53:18	
1624470838878380517	2021-6-24	30.00	1:53:58	
1624470885147766359	2021-6-24	30.00	1:54:45	
1624470925138656689	2021-6-24	30.00	1:55:25	
1624470965155639222	2021-6-24	30.00	1:56:05	
1624471005153172994	2021-6-24	30.00	1:56:45	
1624471045168245775	2021-6-24	30.00	1:57:25	

Figure 9: Database Testing

Figure 10 displays the dashboard developed via Grafana, which effectively presents the detected data. This dashboard enabled the monitoring of data across the entire factory from the monitoring room. A worker was required

to address or rectify an issue only if an anomaly alert was received via Telegram. Thus, the dashboard provided real-time views of the current temperature, time, date, and a graph illustrating temperature variations.

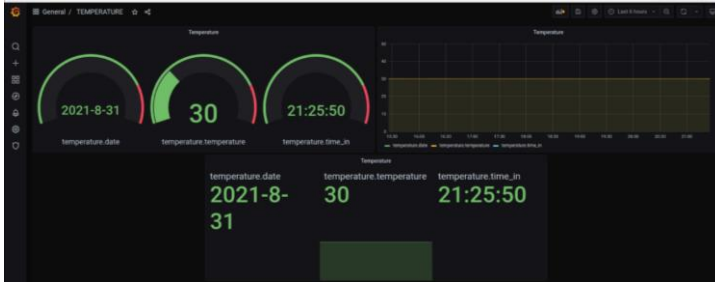
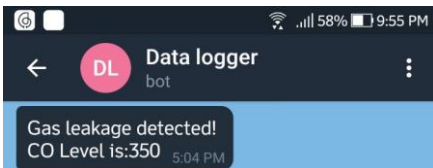


Figure 10 Grafana Dashboard

Figure 11 illustrates the alert message that on-duty staff received upon detecting an anomaly. When the system identified any gas leakage or high-temperature reading as shown in Figure 11 (a) and (b) respectively, it sent a user notification including the specific reading values immediately via Telegram, including the specific reading values.



(a) Gas leakage detection notification



(b) Temperature detection notification

Figure 11: Notification Received by a user using Telegram for Anomaly Detection.

3.2 Performance Test

The processing time in a data logger system, particularly the time taken from data detection on a Raspberry Pi to its visualization on a dashboard and notification via external platforms, plays a critical role in the overall performance and effectiveness of the system. A short processing time ensures that real-time data is promptly captured, transmitted, and displayed, allowing for immediate decision-making and timely responses to anomalies. Conversely, long processing delays can lead to outdated information, reducing the ability to monitor and react to critical events in a timely manner. For industrial applications

where real-time monitoring is crucial, the efficiency of this data flow, from detection to dashboard visualization and notification, directly impacts the reliability, accuracy, and usefulness of the data logger.

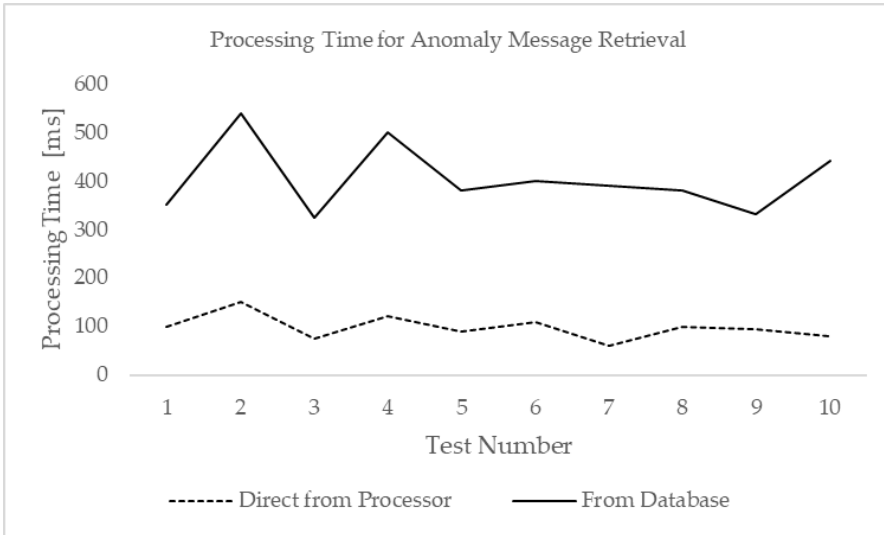


Figure 12: Processing Time for Message Retrieval

Figure 12 shows consistent anomaly detection and message retrieval times across all test cases. The proposed anomaly message transmission, which was directly sent from a processor (Raspberry Pi) performed a lower processing time, compared to the message retrieval from database. The reason for this is that once the data is stored in the database, the query time becomes part of the overall processing time [13]. Reducing processing time is crucial, as it allows for prompt action to be taken based on the message alert. Delayed actions can negatively impact factory production, potentially resulting in company losses.

3.3 Usability Test

System Usability Scale (SUS) is a realistic and accurate method for assessing perceived user friendliness, which can be used across a wide variety of digital goods and services to help programmers and developers to decide whether a design has the solution for overall problems [14][15]. According to [16], SUS is not diagnostic and is used for an overall usability assessment as described in ISO 9241-11, consisting of the following characteristics.

- Effectiveness—can users successfully achieve their objectives?
- Efficiency—how much effort and resources are expended in achieving those objectives
- Satisfaction—was the experience satisfactory?

To evaluate the developed data logger system, feedback was collected from 20 respondents whose job roles are directly related to data logger usage, such as managers and site supervisors responsible for monitoring and analyzing data. The feedback is on two modules: the dashboard using Grafana and anomaly notification via Telegram. The modules were tested by respondents working on monitoring data logger system. Figure 13 shows the usability test result of the developed data logger.

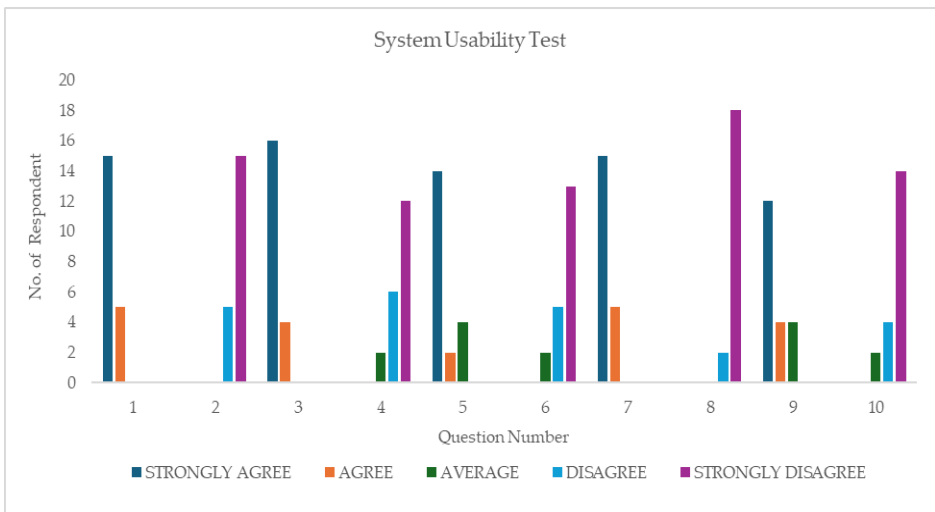


Figure 13: System usability test for developed system

Based on the data collected shown on the previous figure, the calculation score for SUS is counted. Formula used to calculate data is shown as follows:

$$\text{SUS Score} = \frac{(X+Y)25}{(R*5)} \times 100\% \quad (1)$$

where, X = Total odd-numbered questions – 50 and, Y= 150 - total even-

numbered questions. Table 5 shows a tabulated data calculation of the SUS for the developed data logger. The calculation is done based on calculation method in [16][17], and the proposed prototype achieves grade A where the System Usability Score is 91.25.

Cronbach's Alpha (α) is a widely used metric to evaluate the internal consistency or reliability of a set of items, such as survey questions or test responses, designed to measure a single construct. It reflects the degree to which individual items correlate with each other and collectively measure the intended concept. In the context of the System Usability Scale (SUS) conducted, this metric ensures that all items consistently capture aspects of usability as perceived by respondents.

Table 5: Data collection from SUS test

Number of respondents	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Raw Score	Final Score
1	4	4	4	4	4	4	4	4	4	4	40	100
2	4	4	4	4	4	4	3	4	4	4	39	97.5
3	4	4	4	4	4	4	4	4	4	4	40	100
4	3	3	4	2	4	3	3	3	4	2	31	77.5
5	4	4	4	4	4	4	4	4	4	4	40	100
6	4	4	3	3	4	3	4	4	3	4	36	90
7	3	3	3	3	2	4	3	4	2	3	30	75
8	4	4	4	3	3	4	4	4	2	3	35	87.5
9	4	4	4	4	2	2	4	4	3	4	35	87.5
10	4	3	4	4	4	4	4	4	4	4	39	97.5
11	4	4	4	4	4	4	4	4	4	4	40	100
12	4	4	4	4	4	3	4	4	4	4	39	97.5
13	4	4	4	4	4	4	4	4	4	4	40	100
14	3	3	4	2	4	3	3	3	4	2	31	77.5
15	4	4	4	4	4	4	4	4	4	4	40	100
16	4	4	3	3	4	4	4	4	3	4	37	92.5
17	3	3	3	3	2	3	3	4	2	3	29	72.5
18	4	4	4	3	3	4	4	4	2	3	35	87.5
19	3	4	4	4	2	2	4	4	3	4	34	85
20	4	4	4	4	4	4	4	4	4	4	40	100
Average:												91.25

For the SUS dataset provided, the calculated Cronbach's Alpha is 0.835, indicating good reliability. This value surpasses the widely accepted threshold of 0.7, demonstrating that the SUS items (Q1 to Q10) are highly consistent in evaluating the usability of the system. Scores in the range of 0.8 to 0.9 suggest not only reliable but also cohesive items, which collectively provide a comprehensive understanding of user satisfaction and system efficiency. Importantly, high reliability ensures that the items likely measure a single usability dimension with minimal overlap or redundancy.

This reliability assessment underscores the robustness of the SUS instrument in this context. A Cronbach's Alpha of 0.835 indicates confidence in the results, as it minimizes measurement errors and strengthens the credibility of the findings. The SUS responses can thus be relied upon to provide meaningful insights into the system's usability, facilitating data-driven improvements and validating the overall evaluation process.

4.0 CONCLUSION

In conclusion, this project successfully addresses the limitations of existing data loggers in IIoT environments by developing a system with real-time data acquisition, scalable storage, robust anomaly detection, and a user-friendly interface. Utilizing a Raspberry Pi as the gateway for processing and sending sensor data to an InfluxDB time-series database, the system ensures efficient data management and real-time monitoring. Enhanced by Grafana for advanced data visualization and Node-RED for flexible development, the dashboard facilitates easy monitoring of factory workflows and environmental conditions. The direct anomaly messaging feature reduces processing time, as validated by performance tests. Additionally, user evaluations reveal high satisfaction, with the system receiving a grade A in usability testing, scoring 91.25%, demonstrating its effectiveness and user-friendliness. Furthermore, Cronbach's Alpha of 0.835 for the System Usability Scale (SUS) indicates good reliability, confirming that the survey items consistently measure usability, provide credible insights, and support data-driven system improvements.

ACKNOWLEDGMENTS

The authors would like to thank Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia for the support.

AUTHOR CONTRIBUTIONS

N. Harum: Requirement Analysis, Result and Validation, Supervision, Original Draft Preparation; N.A. Emran: Software, Writing-Reviewing and Editing; M.H.F. Md Fauadi: Writing-Reviewing, Editing, Data Validation; E. Hamid: Data Validation; M.N.M. Khambari: Data Analysis; M. M. Ridzuan: Requirement Analysis; M. Kchouri: Data Validation.

CONFLICTS OF INTEREST

The manuscript has not been published elsewhere and is not under consideration by other journals. All authors have approved the review, agree with its submission and declare no conflict of interest in the manuscript.

REFERENCES

- [1] S. Sendari, Y. Rahmawati, F. Ramadhan, F. Alqodri, T. Tibyani, T. Matsumoto, A. Fujiyama, and I. Rachman, "Energy Usage Monitoring System For Environmental Mobile Station", *Journal of Advanced Manufacturing Technology*, vol. 16, no. 3, pp. 29-41, 2022.
- [2] I. Zehra, R. Mehrotra, M. A. Ansari, R. Agrawal, and A. Pathak, "A. Real-Time Tracking of Health Parameters Using IoT and Data Logger: Application to COVID-19", in *Modern Electronics Devices and Communication Systems*, R. Agrawal, C. K. Singh, A. Goyal, D.K. Singh. *Lecture Notes in Electrical Engineering*, vol. 948, pp. 303-313, 2023.
- [3] O. Peter, A. Pradhan, and C. Mbohwa., "Industrial internet of things (IIoT): opportunities, challenges, and requirements in manufacturing businesses in emerging economies," *Procedia Computer Science*, vol. 217, pp. 856-865, 2023.
- [4] M. M. Aljarrah, F. H. Zawaideh, M. Magableh, H. Al Wahshat, R. R. Mohamed and A. K. V, "Internet of Thing (IoT) and Data Analytics with Challenges and Future Applications," in *2023 International Conference on*

Computer Science and Emerging Technologies (CSET), Bangalore, India, pp. 1-8, 2023.

- [5] D. Dobrilovic, V. Brtko, Z. Stojanovic, G. Jotanovic, D. Perakovic, and G. Jausevac "A Model for Working Environment Monitoring in Smart Manufacturing" *Applied Sciences*, vol.11, no.5, pp.28-50, 2021.
- [6] M. Sajjad, M. Z. Yusoff, J. Ahmad, "Design of a Low-Cost High-Speed and Large-Capacity Data Logger Using MicroSD Card," *Springer Lecture Notes in Electrical Engineering*, vol 758, pp 651–657, 2022.
- [7] P. Mahdi, H. Ghadamian, M. Meisam, M. Masoud, "Design, Fabrication, and Experimental Study of a Low-cost and Accurate Weather Station Using a Microcontroller System," *Journal of Renewable Energy and Environment*, vol. 10, no. 4, pp. 35 – 43, 2023.
- [8] B.E. Demir, "A New Low-Cost Internet of Things-Based Monitoring System Design for Stand-Alone Solar Photovoltaic Plant and Power Estimation," *Applied Sciences*, vol.13, no. 24. pp.13072, 2023.
- [9] M. D. Mudaliar and N. Sivakumar, "IoT based real time energy monitoring system using Raspberry Pi," *Internet of Things*, vol. 12, no. 100292, 2020.
- [10] W. Chen, "Intelligent manufacturing production line data monitoring system for industrial internet of things", *Computer Communications*, vol. 151, pp. 31-41, 2020.
- [11] V. J. Mawson, and B. R. Hughes, "Deep learning techniques for energy forecasting and condition monitoring in the manufacturing sector", *Energy and Buildings*, vol. 217, no. 109966, 2020.
- [12] A. Yadav and N. Sakle, "Development of low-cost data logger system for capturing transmission parameters of two-wheeler using Arduino," *Materials Today: Proceedings*, vol. 72, pp. 1697-170., 2023.
- [13] Z. Hong, L. Ward, K. Chard, "Challenges and Advances in Information Extraction from Scientific Literature: a Review," *The Journal of the Minerals, Metals & Materials Society*, vol. 73, no. 7612, pp. 1-18, 2021.
- [14] N. B. Harum, N. A. A. Mahin, E.Hamid, N. A. Emran, S. Anawar, A. Asnawi, "RFID Attendance System with Contagious Disease Prevention Module using Internet-of-Things Technology", *International Journal of Interactive Mobile Technologies*, vol. 17 no. 18, pp. 4-15, 2023,
- [15] N. B. Harum, N. I. M. S. K, N. A. Emran, N. Abdullah, N. A. Zakaria, Erman Hamid, S. Anawar, "A Development of Multi-Language Interactive Device using Artificial Intelligence Technology for Visual Impairment Person", *International Journal of Interactive Mobile Technologies (ijIM)*, vol. 17 no. 18, pp. 79-92, 2021.
- [16] R. L. James and J. Sauro, "Usability And User Experience: Design And Evaluation. *Handbook of Human Factors and Ergonomics*, New Jersey: John Wiley & Sons, 2021.