Journal of Social Research

P-ISSN:2827-9832
E-ISSN: 2828-335x
Vol. 5, No. 7, June 2026
http://ijsr.internationaljournallabs.com/index.php/ijsr

Implementation of Data Logging and Historical Graphs of Furnace Temperature at the Electrical Engineering Department Laboratory, Manado State Polytechnic

Mohamad Fathan Masloman*, Gilang Ramadhan S. Luawo, Jim Mardin Wanimbo, Sintya Paula Junaedy, Franky Gerald Cliford Manoppo
Politeknik Negeri Manado, Indonesia
Email: fathan.masloman@gmail.com*, gilangluawo33@gmail.com, Jimmardin.wanimbo@gmail.com

Keywords:
Data Logging;
Historical Graph;
Furnace Temperature;
Arduino;
MAX6675

ABSTRACT

Accurate temperature monitoring is essential in laboratory and industrial operations involving furnaces to ensure process consistency, prevent overheating, and support safety compliance. Traditional monitoring methods, such as analog gauges or standalone digital thermometers, provide only instantaneous readings and fail to store historical data or visualize trends, limiting experimental reproducibility and quality control. This study aims to design, implement, and evaluate a low-cost, real-time data logging and historical graphing system for furnace temperature monitoring at the Electrical Engineering Laboratory of Politeknik Negeri Manado. The system integrates Type K thermocouples, MAX6675 thermocouple-to-digital modules, Arduino Uno microcontrollers, and a Python-based visualization application. The research employed an applied R&D approach, including stages of literature review, requirements analysis, system design, implementation, testing, and evaluation. Controlled experiments were conducted at five temperature set points, and a four-hour continuous logging test was performed to assess accuracy, logging continuity, and graphical performance. Results indicate that the system achieved a mean absolute error of 1.1°C, maintained uninterrupted data logging over 14,400 samples, and rendered real-time graphs with stable CPU usage. The findings confirm that the integrated system meets design specifications, providing a reliable and educationally valuable tool for laboratory use. This study concludes that low-cost, modular data logging solutions can effectively enhance instrumentation education and support precise thermal monitoring, with potential for further improvements in multi-channel and cloud-based monitoring systems.

INTRODUCTION

The user prompt is empty, so I cannot determine the primary language. However, based on the thinking block being in English, here is my summary: Identified formatting requirements and italicization targets systematically

Temperature monitoring is a critical aspect of laboratory and industrial operations involving furnaces (Chen, 2023; Groenewald et al., 2018; Laciak et al., 2011; Pyszko et al., 2015; Zhou et al., 2020). Precise and continuous temperature recording enables operators to ensure process consistency, prevent overheating, and comply with safety standards (Cană et

al., 2025; Edler, 2023; Prasad et al., 2024; Rahman & Abi Hamid, 2025). In the context of the Electrical Engineering Laboratory at Politeknik Negeri Manado, the furnace is used for materials testing and electrical component characterization experiments that require controlled thermal environments.

Traditional temperature monitoring methods rely on analog gauges or standalone digital thermometers, which provide instantaneous readings but lack the capability to store historical data or visualize temperature trends over extended periods. The absence of a logging mechanism makes it difficult to trace temperature behavior during experiments, identify anomalies, or produce documentation for quality control and academic reporting purposes (Sarkar, 2022; Viola et al., 2022; Wickramasinghe et al., 2023).

Data logging systems coupled with graphical interfaces have emerged as an effective solution to these limitations. By integrating digital sensors, microcontrollers, and computer-based software, it is possible to build low-cost, flexible monitoring systems that record, display, and store temperature data in real time. Several prior studies have addressed similar needs: Ref (Cecchi et al., 2024) demonstrated an Arduino-based thermocouple logging system for kiln monitoring with data storage to SD card; Ref (Hunter, 2007) presented a Python/Matplotlib solution for real-time sensor data visualization in educational settings (Fadhila et al., 2026). evaluated MAX6675-based temperature acquisition accuracy across a range of industrial furnace conditions. Building upon these foundations, this work integrates acquisition, logging, and visualization into a unified system tailored to the laboratory environment.

The specific problem addressed in this study emerges from the Electrical Engineering Laboratory at Politeknik Negeri Manado, where a muffle furnace is used for materials testing and electrical component characterization. Existing methods rely on standalone thermometers that cannot continuously log data or visualize historical temperature trends, which limits the capacity for quality control, experimental reproducibility, and teaching laboratory students modern instrumentation practices.

Previous studies have explored digital temperature acquisition and visualization techniques. Anand et al. (2016) demonstrated an Arduino-based data logging system for kiln monitoring with SD card storage, while Hunter (2007) showcased Python and Matplotlib for real-time sensor data visualization. Ibrahim et al. (2018) evaluated MAX6675 thermocouple modules for industrial temperature sensing, confirming their accuracy under variable conditions. These studies provide a foundation for integrating acquisition, logging, and graphical visualization in a laboratory context.

Despite these contributions, a gap remains in implementing a fully integrated, low-cost, real-time data logging system tailored to educational laboratories (Al-Zoubi et al., 2023; Doloi et al., 2025; Guerrero-Osuna et al., 2024; Moradi et al., 2026; Tariq et al., 2024). Most prior work focuses on industrial-scale setups or single-function logging systems, which are either cost-prohibitive or technically complex for student use. Moreover, the combination of Arduino-based acquisition with Python-driven graphical outputs, validated against NIST-traceable instruments, has not been systematically applied in academic laboratory teaching environments.

Addressing this gap is urgent due to the increasing reliance on digital instrumentation in engineering education and industrial practices. By providing students with hands-on

experience in end-to-end data acquisition, processing, and visualization, laboratories can enhance practical competencies, reinforce theoretical concepts, and reduce operational errors. Additionally, accurate monitoring supports experimental reproducibility, safety compliance, and process documentation.

The novelty of this research lies in its implementation of a three-layer system integrating Type K thermocouples, MAX6675 modules, Arduino microcontrollers, and Python-based visualization software. This system achieves continuous logging with ±2°C accuracy, real-time historical graphing, and automated data export, creating an accessible, low-cost, and educationally relevant tool. Its modular architecture allows students to modify each layer independently, fostering active learning and experimentation.

The purpose of the study is to design, implement, and validate a digital furnace temperature logging system capable of recording and graphing data continuously over extended periods. This purpose encompasses both practical engineering objectives—accuracy, reliability, and system stability—and educational goals, including enhancing student understanding of instrumentation principles and data analysis techniques.

The expected contribution of this research includes providing a validated, open-source system suitable for laboratory teaching environments, as well as empirical evidence of its accuracy and stability through controlled experiments. The design demonstrates a cost-effective alternative to commercial data loggers, with total hardware costs significantly lower while maintaining performance adequate for educational purposes.

Finally, the research objectives are to develop an end-to-end temperature monitoring system, evaluate its accuracy against reference instruments, assess data logging continuity and graphical performance, and demonstrate its applicability in an academic setting. The anticipated benefit is multifold: improving laboratory pedagogy, increasing student competence in digital instrumentation, and establishing a replicable model for other laboratories seeking affordable and robust monitoring solutions.

METHOD

Research Type and Approach

This study employed an applied research approach combined with a research-and-development (R&D) methodology. The research is classified as applied because it directly addresses a practical instrumentation problem in the Electrical Engineering Laboratory of Politeknik Negeri Manado, and as R&D because it produces a tangible engineering artifact — an integrated hardware-software data logging system — that is subsequently subjected to empirical evaluation under realistic operating conditions. The development of the system follows the engineering design process: requirements analysis, conceptual design, detailed design, implementation, testing, and evaluation, with quantitative metrics used to verify that each design specification has been met.

Research Location and Object

The research was conducted at the Electrical Engineering Laboratory, Department of Electrical Engineering, Politeknik Negeri Manado, where the target furnace is installed and routinely used for materials testing and electrical component characterization. The object of the research is the muffle furnace itself (maximum rated temperature 1000 °C) together with

the proposed data logging and historical graphing system. All hardware assembly, firmware development, software development, calibration, and performance testing were carried out on-site so that the operating environment of the data logging system replicates the actual conditions under which it will be used by students and laboratory staff.

Research Stages

The research was executed in six sequential stages. The first stage, literature study, reviewed prior work on thermocouple-based temperature acquisition, MAX6675 signal conditioning, microcontroller serial communication, and Python-based real-time plotting, providing the theoretical and technical basis presented in Section 2. The second stage, requirements analysis, identified functional requirements (continuous logging, real-time visualization, CSV export, and ±2 °C accuracy) and non-functional requirements (low cost, ease of student use, and compatibility with the existing laboratory PC) through informal discussions with laboratory technicians and instructors.

The third stage, system design, produced the three-layer architecture (sensing, acquisition, and application) described in Section 4, including the selection of the Type K thermocouple, MAX6675 module, Arduino Uno, and Python toolchain. The fourth stage, implementation, comprised physical construction of the sensing front-end, wiring of the MAX6675 to the Arduino, development of the Arduino firmware in C/C++, and development of the Python host application using PySerial, Matplotlib, and pandas, as described in Section 5. The fifth stage, testing and data collection, consisted of controlled experiments at five furnace temperature set points and a four-hour continuous logging trial, as detailed in Section 6.1. The sixth and final stage, data analysis and reporting, involved quantitative comparison of logged values against a NIST-traceable reference, computation of error metrics and performance indicators, and preparation of this paper.

Tools and Materials

The complete inventory of hardware and software components is provided in Table 1 (Section 5.1). The principal hardware components are the muffle furnace (maximum 1000 °C), a Type K thermocouple probe (stainless-steel sheath, 6 mm × 300 mm), the MAX6675 thermocouple-to-digital converter module, and the Arduino Uno R3 microcontroller. The host PC ran Windows 10, with the Arduino IDE 2.3.2 used for firmware development and Python 3.11 (PySerial 3.5, Matplotlib 3.8, pandas 2.1) used for the host application. For accuracy benchmarking, a Fluke 51-II digital thermometer with a NIST-traceable calibration certificate was employed as the reference instrument. Supporting tools included a digital multimeter for wiring verification, a stopwatch for independent sampling-interval verification, and high-temperature ceramic fiber insulation at the thermocouple entry point of the furnace door.

Data Collection Method

Primary data were collected through direct measurement during controlled experiments on the laboratory furnace. For accuracy testing, the furnace was heated to each of five target set points (100 °C, 250 °C, 400 °C, 600 °C, and 800 °C) and held for fifteen minutes at each point to achieve thermal stability before paired readings were recorded from both the logged system output and the Fluke 51-II reference thermometer; ten consecutive logged samples were

averaged at each set point to suppress short-term noise. For continuity testing, the system was operated without interruption over a four-hour furnace heating-and-cooling cycle, with all timestamped records written automatically to the CSV log file. The 1-second nominal sampling interval was independently verified using a stopwatch over a window of one hundred consecutive samples. Secondary data, used to support the literature study and the design rationale, were obtained from peer-reviewed journal articles, IEEE conference proceedings, and manufacturer datasheets as listed in the References section.

Data Analysis Method

The collected data were analyzed using both quantitative and graphical techniques. Quantitative analysis comprised computation of the absolute error and mean absolute error (MAE) between the logged and reference temperatures across the five set points, evaluation of sampling-interval stability through the mean and standard deviation of the measured interval, verification of the total record count against the expected sample population over the four-hour test, and measurement of host-PC CPU usage and graph refresh rate during sustained plotting. Graphical analysis was performed by inspecting the rendered Matplotlib trend graph for continuity, anomalies, and overall visual interpretability throughout the heating and cooling phases. The analytical outcomes are reported in Section 6 and interpreted against the original design specifications in Section 7. The system was considered to meet its objectives if the MAE was within ±2 °C, the logging session produced no data gaps, and the graph remained responsive at the 1 Hz refresh target.

Overall System Architecture

The data logging and historical graphing system is structured in three functional layers. The sensing layer consists of a Type K thermocouple probe inserted into the furnace chamber and connected to a MAX6675 module for cold-junction compensation and analog-to-digital conversion. The acquisition layer is implemented by an Arduino Uno microcontroller, which reads the digitized temperature value from the MAX6675 via SPI at configurable sampling intervals and transmits the data over a USB-to-serial link. The application layer runs on a Windows-based personal computer, hosting a Python script that reads incoming serial data, appends timestamped records to a CSV log file, and renders a continuously updated temperature trend graph on screen.

Figure 1. Block diagram of the three-layer data logging and historical graphing system for furnace temperature monitoring.

Hardware Design

The sensing front-end comprises a Type K thermocouple probe (stainless steel sheath, 6 mm diameter, 300 mm insertion length) connected to the MAX6675 breakout module. The MAX6675 communicates with the Arduino Uno using three-wire SPI (SCK, CS, SO) without the MOSI line, as the MAX6675 is a read-only device. The Arduino Uno is powered via its USB connection to the host PC, eliminating the need for an external power supply for the acquisition electronics.

The furnace used in this study is a small muffle furnace with a maximum rated temperature of 1000°C. The thermocouple probe is inserted through a dedicated port in the

furnace door, with high-temperature ceramic fiber insulation at the entry point to prevent heat loss and protect the cable insulation.

Software Design

The Arduino firmware reads the MAX6675 output at a 1-second interval using bit-banged SPI and transmits the temperature value along with a sample counter over the serial port at 9600 baud in comma-delimited format. Error checking includes detection of the MAX6675 open-thermocouple fault bit.

The Python host application uses the PySerial library to open and read the COM port. Each incoming data line is parsed and a timestamp is appended using Python's datetime module. The record is then written to a CSV file and appended to an in-memory list used for plotting. Matplotlib's FuncAnimation class is used to refresh the graph at 1-second intervals, providing a real-time view of temperature evolution. The graph includes gridlines, axis labels, a title, and a moving window of the last 300 data points (5 minutes) to maintain readability during long experiments.

Hardware Implementation

The MAX6675 module was connected to the Arduino Uno as follows: VCC to 3.3V, GND to GND, SCK to pin D13, CS to pin D10, and SO to pin D12. The Type K thermocouple leads were connected to the MAX6675's T+ and T− terminals, observing correct polarity. The assembled system was mounted on a plastic enclosure fixed to the side of the furnace, with only the thermocouple probe cable passing through a cable gland into the furnace door.
The Arduino was connected to the laboratory PC via a standard USB-A to USB-B cable. The COM port assignment (COM3) was confirmed in Windows Device Manager and configured identically in both the Arduino IDE serial monitor and the Python script.

Table 1. Hardware and software components of the temperature data logging system.

Parameter	Specification / Value
Microcontroller	Arduino Uno R3 (ATmega328P)
Temperature Sensor	Type K Thermocouple, 6mm × 300mm stainless sheath
Signal Conditioner	MAX6675 Thermocouple-to-Digital Converter (0–1024°C, 12-bit)
Communication (MCU–PC)	USB-to-Serial (UART), 9600 baud, COM3
Host PC OS	Windows 10, Intel Core i5, 8 GB RAM
Programming Language	Arduino C/C++ (firmware), Python 3.11 (host application)
Python Libraries	PySerial 3.5, Matplotlib 3.8, pandas 2.1
Data Storage Format	CSV (Comma-Separated Values), timestamped records

Parameter	Specification / Value
Sampling Interval	1 second (configurable)
Furnace Type	Muffle Furnace, max. 1000°C

Firmware Implementation

The Arduino firmware was developed in the Arduino IDE 2.3.2. A custom SPI bit-banging routine reads the 16-bit SPI data frame from the MAX6675 on each sampling cycle. The 12-bit temperature value is extracted from bits [14:3] and multiplied by 0.25 to obtain the temperature in degrees Celsius. The open-thermocouple fault flag (bit 2) is checked; if asserted, the firmware transmits an error code ('ERR') instead of a numeric value, which the Python host application logs and displays as a gap in the trend graph.

The serial output format is: <sample_number>,<temperature_celsius> (e.g., '125,352.25'), transmitted as a newline-terminated ASCII string at 1-second intervals using the Arduino millis() non-blocking timing approach to maintain accurate sampling periods without using delay().

Python Application Implementation

The Python host application is structured as a main script with three primary functions: (1) serial_reader(), which opens the PySerial port and continuously reads incoming data lines into a thread-safe queue; (2) data_logger(), which dequeues records, appends timestamps, writes to the CSV file, and updates the in-memory data lists; and (3) update_plot(), called by Matplotlib's FuncAnimation every 1000 ms, which redraws the line chart from the current in-memory data list.

The CSV file is opened in append mode at application startup with a header row (Timestamp, Sample_No, Temperature_C). Each new record is written as a separate row. The plot displays temperature (°C) on the Y-axis and elapsed time in minutes on the X-axis, with a rolling window of the last 300 samples. Axis limits auto-scale to the visible data range, with a fixed minimum Y-axis margin of 10°C to prevent graph distortion during stable temperature periods.

RESULT AND DISCUSSION

Test Methodology

System performance was evaluated through three test categories: (1) accuracy testing, comparing logged temperature values against a calibrated reference thermometer (Fluke 51-II digital thermometer with NIST-traceable calibration) at five temperature set points (100°C, 250°C, 400°C, 600°C, 800°C); (2) data logging continuity testing, verifying uninterrupted record capture over a 4-hour heating cycle; and (3) graph rendering performance testing, measuring CPU usage and frame update rate during continuous plotting operations.

The furnace was heated to each target temperature and held for 15 minutes to achieve thermal stability before reference and logged values were compared. Ten consecutive logged samples were averaged for each set point comparison. A stop-watch was used to verify the 1-second sampling interval independently.

Temperature Measurement Accuracy

Table 2 summarizes the accuracy test results. Across all five set points, the maximum absolute error between the logged value and the Fluke reference was 1.8°C (at 800°C), and the minimum was 0.5°C (at 100°C). The mean absolute error (MAE) across all set points was 1.1°C, well within the ±2°C specification target. These results are consistent with MAX6675 datasheet accuracy specifications of ±3°C at full range.

Table 2. Temperature measurement accuracy comparison between logged values and calibrated reference

Set Point (°C)	Reference (°C)	Logged Value (°C)	Absolute Error (°C)
100	100.3	99.8	0.5
250	250.7	249.5	1.2
400	401.2	400.0	1.2
600	600.9	599.7	1.2
800	800.5	798.7	1.8
Mean Absolute Error			**1.1**

Data Logging Continuity

Over the 4-hour continuous test, the system logged 14,400 records with zero data gaps or file corruption events. The sampling interval was verified to be 1.00 s ± 0.02 s (measured over 100 consecutive samples using an external stopwatch), confirming that the Arduino millis()-based timing approach maintains consistent intervals without drift. The resulting CSV file size was 523 KB, well within practical storage constraints.

Graph Rendering Performance

The Matplotlib FuncAnimation update loop maintained a stable 1-second refresh rate throughout the 4-hour test with a mean CPU usage of 8.3% on the host PC (Intel Core i5, 8 GB RAM, Windows 10). No frame drops or application freezes were observed. The rolling 300-sample display window (5 minutes) provided sufficient temporal context for monitoring the furnace heating and cooling phases without visual clutter from earlier data points.

Table 3. Summary of system performance test results.

Parameter	Result
Mean Absolute Error (Temperature)	1.1°C (target: ≤ ±2°C)
Maximum Absolute Error	1.8°C at 800°C
Sampling Interval Accuracy	1.00 s ± 0.02 s

Parameter	Result
Total Records Logged (4 hours)	14,400 records, 0 gaps
CSV File Size (4-hour session)	523 KB
Graph Refresh Rate	1 frame/second (stable)
Mean CPU Usage (Plotting)	8.3% (Intel Core i5, Windows 10)
Data Logging Latency	< 500 ms (target met)

The results confirm that the implemented system successfully fulfills its design objectives. The mean absolute temperature error of 1.1°C is well within the ±2°C target and consistent with MAX6675 specifications, validating the sensor-conditioner selection. The dominant source of error is the inherent non-linearity of the Type K thermocouple at higher temperatures, which could be further reduced by implementing polynomial calibration correction in the Python application using standard NIST thermocouple reference tables.

The 14,400-record, gap-free logging performance over 4 hours demonstrates the reliability of the serial communication link and the robustness of the PySerial-based data capture approach. No buffer overflow or data loss events were observed, attributable to the use of a dedicated background thread for serial reading decoupled from the slower plotting thread — a design pattern recommended in Ref (Fadhila et al., 2026). for real-time Python instrumentation applications.

The graph rendering performance (8.3% CPU, stable 1 fps) indicates that the Matplotlib FuncAnimation approach is adequate for this application. This is consistent with findings by (Hunter, 2007), who reported similar CPU loads for real-time sensor data plots at comparable update rates. For applications requiring higher update rates (e.g., 10 fps) or multi-channel display, a transition to a more performant plotting library such as PyQtGraph or Bokeh would be recommended.

From an educational perspective, the system provides students with direct experience of the complete data acquisition chain — from physical sensor to digital readout to stored data to visual graph — reinforcing theoretical concepts in instrumentation, digital communication, and data analysis. The open-source, component-based nature of the implementation (Arduino + Python) also allows students to examine and modify each layer of the system independently, supporting active learning approaches.

Compared to commercial data loggers such as the Fluke 1744 or Hioki LR8400, the implemented system offers significantly lower cost (estimated component cost: IDR 450,000 vs. > IDR 10,000,000 for commercial units) at the expense of fewer built-in safety features and lower official calibration traceability. For educational and non-critical monitoring purposes, this trade-off is acceptable.

CONCLUSION

The implementation of the three-layer furnace temperature monitoring system successfully demonstrated the integration of Type K thermocouples, MAX6675 modules, Arduino microcontrollers, and Python-based visualization to achieve continuous, accurate,

and reliable data logging. Experimental validation showed that the system maintained a mean absolute error of 1.1°C across multiple temperature set points and recorded 14,400 uninterrupted samples during a 4-hour test cycle, confirming both accuracy and stability. The graphical interface provided real-time historical visualization with minimal CPU usage, offering an effective educational tool for students to observe and analyze temperature trends, thereby enhancing practical understanding of instrumentation, digital communication, and data analysis principles.

For future research, it is recommended to explore higher update-rate visualization libraries, such as PyQtGraph or Bokeh, to accommodate multi-channel monitoring and faster sampling needs. Additionally, incorporating polynomial calibration or machine-learning-based error correction could further improve accuracy at higher temperature ranges. Expanding the system to support networked monitoring and cloud-based storage would enable remote access and longitudinal data analysis, making it applicable to larger-scale laboratory or industrial environments. Such developments would not only refine technical performance but also broaden the pedagogical and practical applications of digital temperature monitoring in academic and professional settings.

REFERENCE

Al-Zoubi, A., San Cristobal, E., Shahroury, F. R., & Castro, M. (2023). The middle east higher education experience: Implementing remote labs to improve the acquisition of skills in industry 4.0. *IEEE Transactions on Learning Technologies*, *17*, 982–991.

Cană, P., Ripeanu, R. G., Diniță, A., Tănase, M., Portoacă, A. I., & Pătîrnac, I. (2025). A review of safety valves: Standards, design, and technological advances in industry. *Processes*, *13*(1), 105.

Cecchi, R., Catania, A., Sbrana, C., Macucci, M., Strangio, S., & Iannaccone, G. (2024). Data loggers for high-temperature industrial environments. *IEEE Access*, *12*, 180726–180737.

Chen, X. (2023). Temperature control in electric furnaces: Methods, applications, and challenges. *Journal of Physics: Conference Series*, *2649*(1), 12032.

Doloi, S., Das, M., Li, Y., Cho, Z. H., Xiao, X., Hanna, J. V, Osvaldo, M., & Tat, L. N. W. (2025). Democratizing self-driving labs: advances in low-cost 3D printing for laboratory automation. *Digital Discovery*, *4*(7), 1685–1721.

Edler, F. (2023). Reliable and Traceable Temperature Measurements Using Thermocouples: Key to ensuring process efficiency and product consistency. *Johnson Matthey Technology Review*, *67*(1), 65–76.

Fadhila, N., Takasumiang, F. S., Wistikhirana, H., Dewangga, Y., Andareska, R. P., Pramono, A. E., & Setiyadi, I. (2026). Measuring Thermal Conductivity and Magnetic Strength Using Low-Cost Sensors Based on an Open-Source Platform Arduino. *Recent in Engineering Science and Technology*, *4*(02), 59–72.

Groenewald, J. W. D., Nelson, L. R., Hundermark, R. J., Phage, K., Sakaran, R. L., Van Rooyen, Q., & Cizek, A. (2018). Furnace integrity monitoring using principal component analysis: an industrial case study. *Journal of the Southern African Institute of Mining and Metallurgy*, *118*(4), 345–352.

Guerrero-Osuna, H. A., García-Vázquez, F., Ibarra-Delgado, S., Mata-Romero, M. E., Nava-Pintor, J. A., Ornelas-Vargas, G., Castañeda-Miranda, R., Rodríguez-Abdalá, V. I., &

Solís-Sánchez, L. O. (2024). Developing a cloud and iot-integrated remote laboratory to enhance education 4.0: an approach for FPGA-based motor control. *Applied Sciences*, *14*(22), 10115.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95.

Laciak, M., Fazekaš, P., & Kačur, J. (2011). Monitoring and control of temperatures in electric furnace in PROMOTIC system. *2011 12th International Carpathian Control Conference (ICCC)*, 239–242.

Moradi, A., Jafari, H., Sajadinia, M., & Zamani, A. (2026). An Innovative Remote Laboratory System With Controller Hardware, Management Software, and Fault Control System. *Computer Applications in Engineering Education*, *34*(2), e70144.

Prasad, S. J. S., Sureshkumar, R., Thangatamilan, M., Nirmal, M., Sriram, N. S., & Thilagavathi, M. (2024). Automation of Oil Press Safety and Consistency Control Using Node MCU. *2024 International Conference on Smart Electronics and Communication Systems (ISENSE)*, 1–6.

Pyszko, R., Brestovič, T., Jasminská, N., Lázár, M., Machů, M., Puškár, M., & Turisová, R. (2015). Measuring temperature of the atmosphere in the steelmaking furnace. *Measurement*, *75*, 92–103.

Rahman, S. A., & Abi Hamid, M. (2025). Tempvision 1000: A Portable Temperature Measurement and Monitoring System for Boiler Combustion. *SAP Proceedings in Interdisciplinary Insights and Innovations*, *3*, 522.

Sarkar, P. R. (2022). Data-Driven Quality Assurance Systems For Food Safety In Large-Scale Distribution Centers. *ASRC Procedia: Global Perspectives in Science and Scholarship*, *2*(1), 151–192.

Tariq, R., Casillas-Muñoz, F., Hassan, S., & Ramírez-Montoya, M. (2024). Synergy of internet of things and education: Cyber-physical systems contributing towards remote laboratories, improved learning, and school management. *Journal of Social Studies Education Research*, *15*(2), 305–352.

Viola, L., Ronchieri, E., & Cavallaro, C. (2022). Combining log files and monitoring data to detect anomaly patterns in a data center. *Computers*, *11*(8), 117.

Wickramasinghe, A., Muthukumarana, S., Schaubroeck, M., & Wanasundara, S. N. (2023). An anomaly detection method for identifying locations with abnormal behavior of temperature in school buildings. *Scientific Reports*, *13*(1), 22930.

Zhou, P., Zhang, R., Xie, J., Liu, J., Wang, H., & Chai, T. (2020). Data-driven monitoring and diagnosing of abnormal furnace conditions in blast furnace ironmaking: An integrated PCA-ICA method. *IEEE Transactions on Industrial Electronics*, *68*(1), 622–631.