Real-Time IoT-Based Solar Power Monitoring System Using Threshold Decision Algorithm and Mobile Push Notifications

Afif Fathur Rachman [1)*], Yasdinul Huda [2)], Ika Parma Dewi [3)], Hadi Kurnia Putra [4)], Asnal Effendi [5)]

[1,2,3,4)]Department of Informatics , Engineering Faculty, State University of Padang
[5)]Electrical Installation Engineering, Vocational Faculty, Padang Institute of Technology
* Corresponding author : email: afiffathur83@gmail.com

Abstract

The establishment of solar power installations as a renewable energy option necessitates dependable monitoring systems to ensure peak performance. Nonetheless, current monitoring solutions typically emphasize only data visualization, lacking intelligent early warning features, which results in slower fault detection and faster battery deterioration. This research tackles this gap by creating a mobile monitoring system based on the Internet of Things (IoT), where both the threshold-based decision algorithm and notification logic are fully integrated into the mobile application, rather than the IoT hardware. The system employs an ESP8266 microcontroller exclusively for gathering data from a PZEM-017 DC sensor, transmitting raw readings to Firebase Realtime Database. A mobile application built on Flutter fetches data, checks it against user-defined thresholds (voltage, current, power), and produces immediate push notifications with anti-spam delays, audio alerts, and optional Telegram sending. Experimental outcomes from 30 measurement locations and 50 anomaly simulations indicate that the system reaches a voltage measurement error of 1.22% and a current error of 9.90% in comparison to a calibrated multimeter. The typical notification delay is 3.4 seconds, significantly less than the 10-second goal. In contrast to edge-based thresholding (on microcontroller), the mobile-centric method enables users to modify thresholds flexibly through the app without the need for hardware reprogramming, offers more extensive notification options (local pop-up, sound, Telegram), and streamlines IoT device upkeep. The specialized mobile interface provides enhanced customization and data logging features in comparison to standard platforms like Blynk or Telegram-based options.

Keywords: Firebase, Flutter, Internet of Things, Real-Time Notifications, Solar Power Plant, Threshold Decision Algorithm

1. INTRODUCTION

The current global energy transition is a crucial step driven by the urgent need to address climate change and reduce dependence on dwindling fossil fuels [1], [2] . Solar Power Plants have emerged as the most promising renewable energy option due to their ability to convert sunlight into electricity without producing greenhouse gas emissions [3]. Indonesia has abundant solar energy potential due to its geographical location, making SOLAR POWER PLANT development a key focus of the country's sustainable development agenda [4]. However, solar energy utilization often faces challenges due to weather fluctuations that impact the stability of generated power [5].

The operational efficiency of SOLAR POWER PLANT is highly dependent on the maintenance of existing components, particularly the energy storage system or battery [6]. One common problem is the lack of a monitoring system that can quickly detect performance degradation [7]. Periodic manual monitoring has been shown to have a high probability of human error and is unable to detect anomalies that occur between inspections. Ignorance of battery condition can result in overcharging or undercharging, which significantly shortens the device's lifespan [8]. For this reason, digital transformation is needed in energy monitoring systems to ensure their sustainability [9]. Internet of Things (IoT) technology provides the ability to automatically and continuously acquire data [10]. Through the integration of intelligent sensors, physical parameters can be converted into digital data, which is then distributed via the internet [9]. The implementation of IoT-based monitoring systems has been shown to improve harvest efficiency and significantly reduce operational costs [11]. The use of microcontrollers such as the ESP8266 is a popular choice due to its effectiveness in wireless data communication [12]. Furthermore, precision sensors such as the PZEM-017 are crucial for obtaining accurate DC voltage and current data in solar power generation systems [13].

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

Even with the increasing use of solar power systems, considerable operational difficulties remain. Numerous IoT-driven monitoring solutions have been suggested in recent publications. Alfita et al. (2021) designed an IoT-based battery monitoring system [14], whereas Hidayat and Firmansyah (2019) built a smart meter for microgrid usage. Nevertheless, current solutions display three primary shortcomings [15]. Many studies employ generic IoT platforms like Blynk, as seen in the work of Maulana et al. (2025), or messaging apps like Telegram, as demonstrated by Wijayanto et al., which provide restricted customization and do not have specialized interfaces for solar monitoring. Secondly, existing systems mainly emphasize data visualization instead of establishing intelligent, threshold-driven early warning systems that can actively alert users to anomalies.[16] [17]. Third, most existing solutions do not provide structured historical data storage and analysis capabilities essential for long-term performance evaluation and predictive maintenance. Therefore, a clear research gap exists: no previous work has integrated a threshold-based decision algorithm with a dedicated mobile application built with Flutter and Firebase to provide real-time notifications and comprehensive data logging specifically for residential-scale solar power monitoring.

To address these gaps, this research proposes a comprehensive IoT-based monitoring system with several key innovations [18]. The system includes a dedicated Flutter-based mobile application that provides an intuitive, purpose-built interface for solar power monitoring, unlike generic platforms [19], [20]. It implements a mobile-focused threshold-based decision algorithm – operating entirely within the Flutter application – that assesses real-time data against user-set thresholds, allowing for anticipatory fault detection without needing firmware updates to the IoT devices [20]. In this architecture, the ESP8266 microcontroller solely focuses on reading the PZEM-017 sensor and transmitting unprocessed voltage, current, power, and energy information to Firebase every 5 seconds, while the mobile application manages all decision-making and notification processes. In contrast to traditional systems that integrate decision logic within the microcontroller, this mobile-focused architecture permits users to modify voltage, current, and power limits in the app's Settings menu, facilitates a customizable cooldown feature to avoid excessive notifications, and accommodates multi-channel alert distribution (local pop-up, sound, Telegram, with planned support for WhatsApp and Google Assistant integration). In addition, the system utilizes an integrated cloud architecture with Firebase Realtime Database and Cloud Messaging to ensure synchronized real-time data updates and reliable push notifications with low latency [21], [22], [23]. Furthermore, it incorporates structured historical data storage to support trend analysis and long-term performance evaluation [19], [24], [23]. This research specifically addresses the operational challenges identified in residential-scale solar power systems, where manual monitoring remains prevalent and commercial solutions are often cost-prohibitive or lack necessary features [18].

Based on the identified problems and a review of existing literature, this study proposes a mobile application design for monitoring the performance of IoT-based solar power plants with direct notification capabilities. This application is expected to facilitate users in monitoring solar power plant conditions flexibly, quickly, and accurately, as well as assist in data-driven decision-making regarding solar power plant maintenance actions.

2. METHODS AND DATA

This study adopts the Prototyping Model as a framework in the system design and development process. This model is a software development approach in which a prototype (initial version of the system) is created, tested, and then refined repeatedly until an acceptable final product is produced.

The main reason for choosing this model is because the research focuses on the design and development of applications with user interfaces (UI) and real-time interactions. One of the experts frequently referenced in the Prototyping Model is Roger S. Pressman, who in his book Software Engineering: A Practitioner's Approach explains that the prototyping model is an effective software engineering approach for understanding user requirements that are not fully defined at the early stages of development [25]. Prototype Development Model Flowchart as in the following figure 1.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

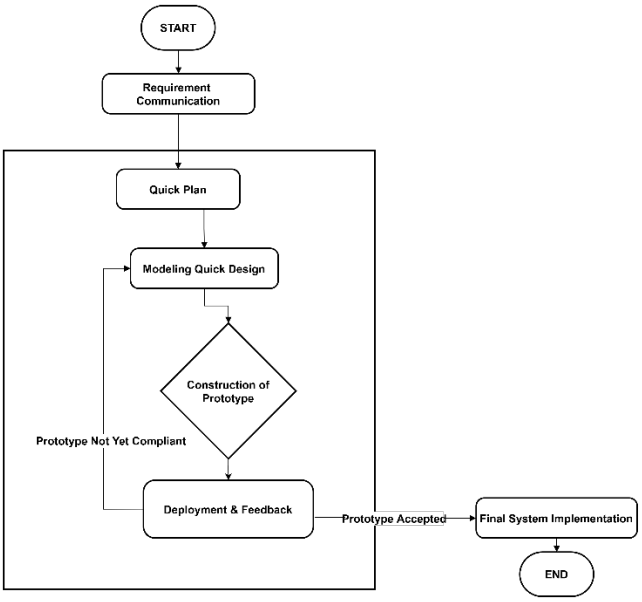


Figure 1. Prototype Development Model Flowchart

The research methodology used in developing this IoT-based solar power plant monitoring system includes the stages of system design, implementation, and testing.

2.1 Experimental Design

Table 1 summarizes the three testing procedures conducted in this study. To guarantee thorough assessment, a structured experimental framework was executed in actual home settings. The system was set up in a home located in Padang, Indonesia. Testing took place over 7 continuous days from 12:00 PM to 6:00 PM daily, corresponding to peak solar irradiance periods. The testing took place under environmental conditions of an ambient temperature of 28–30°C and a relative humidity of around 65–75%. The solar energy system utilized a 12V battery bank made from four Lithium-Ion 18650 cells connected in series (nominal 14.4V, total capacity 3.4Ah), powered by a 50Wp solar panel via a PWM charge controller. The cargo included DC LED bulbs (15W, 20W, 25W) that changed daily. The Fluke 17B+ Digital Multimeter (calibrated) served as the reference tool for accuracy validation

Table 1. Testing Procedures

Num	Test	Method	Metrics
1	Sensor Accuracy	15 measurements per parameter over 7 days; compared with Fluke multimeter; percentage error calculated	Voltage, Current, Power accuracy
2	Notification Latency	20 anomaly simulations; latency measured from data reception to notification display	Time delay (latency)
3	System Reliability	7-day continuous operation monitoring	Wi-Fi stability, transmission success rate, battery endurance

2.2 System Architecture

Figure 2 represents the architecture includes three primary layers: IoT devices, the cloud, and a mobile app. Within the IoT layer, the ESP8266 microcontroller establishes Wi-Fi connectivity and Firebase authentication, subsequently reading electrical parameters (voltage, current, power, and energy) from the PZEM-017 sensor through Modbus RTU at one-second intervals. The raw data is transformed into physical units and sent as a JSON object to the Firebase Realtime Database every 5 seconds, containing a timestamp and device ID, without any local analysis or notification handling. In the cloud layer,

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

Firebase serves as a real-time data storage and synchronization solution designed specifically for time-series data. In the mobile application layer (Flutter), real-time processing of incoming data is achieved through a streaming mechanism. The application accesses user-specified threshold configurations from local storage and assesses each parameter accordingly. In the event of a threshold violation, a cooldown process is implemented to stop duplicate alerts from being sent out before notifications are triggered through system alerts, sounds, or external integrations like Telegram. All logic for decision-making and notifications operates on the mobile device, enabling users to flexibly change thresholds without altering the IoT hardware.

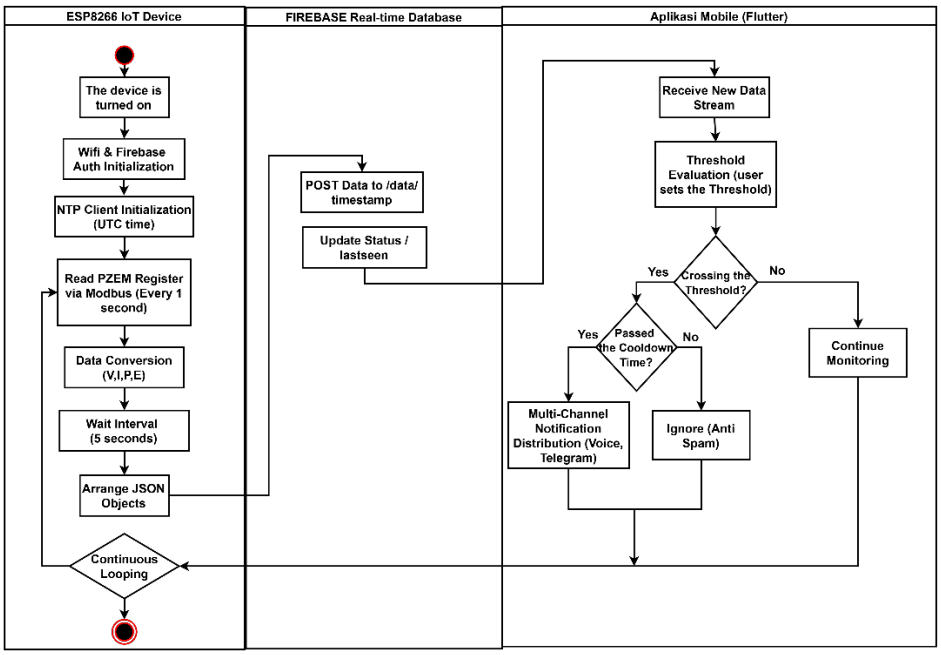


Figure 2. IoT-Based Solar Power Plant Monitoring System Architecture Diagram

2.3 Hardware

Table 2. System Hardware Specifications

Num.	Components	Main Specifications	Function
1	ESP8266	WiFi, 80 MHz, 32-bit MCU	IoT data processing and communications
2	Sensor PZEM-017	DC voltage, DC current	Measurement of electrical parameters
3	RS485 to TTL	Communication converter	Sensor interface to MCU
4	Battery Bank	4x Lithium-Ion 18650 in series (nominal 14.4V, 3.4Ah)	System resources
5	Solar Panel	18 V	Charging source for battery

Table 2 contains the hardware components utilized in the IoT device. The central processing unit is the ESP8266 microcontroller, and the PZEM-017 sensor gauges DC electrical parameters. A converter that transitions RS485 to TTL facilitates communication between the microcontroller and the sensor, while a 12V battery bank (composed of four 18650 cells in series) powers the system, which is charged by an 18V solar panel.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

2.4 System Software

Table 3. System Software Specifications

Num	Software	Technology	Function
1	Firmware IoT	Arduino IDE (C++)	Data acquisition and delivery
2	Backend	Firebase Realtime Database	Real-time data storage
3	Notification	Firebase Cloud Messaging	Notification delivery
4	Mobile Application	Flutter (Dart)	Data monitoring and visualization

Table 3 presents a summary of the software technologies implemented in the system. The IoT firmware is developed in C++ through the Arduino IDE, the backend employs Firebase Realtime Database for storing data, notifications are sent using Firebase Cloud Messaging (FCM), and the mobile app is created with Flutter (Dart) to operate on Android devices.

2.5 Threshold-Based Decision Algorithm

In contrast to traditional IoT monitoring systems that integrate decision logic on the microcontroller, Figure 3 illustrates the flowchart for the threshold decision algorithm implemented in the mobile application. The algorithm initiates upon receiving new sensor data from Firebase, subsequently comparing voltage, current, and power against thresholds defined by the user for minimum and maximum values. If any parameter exceeds its limit, the algorithm moves on to a cooldown evaluation; if not, it resumes waiting for the subsequent data. Notifications that surpass the cooldown period are subsequently sent through various channels (pop-up, sound, Telegram).

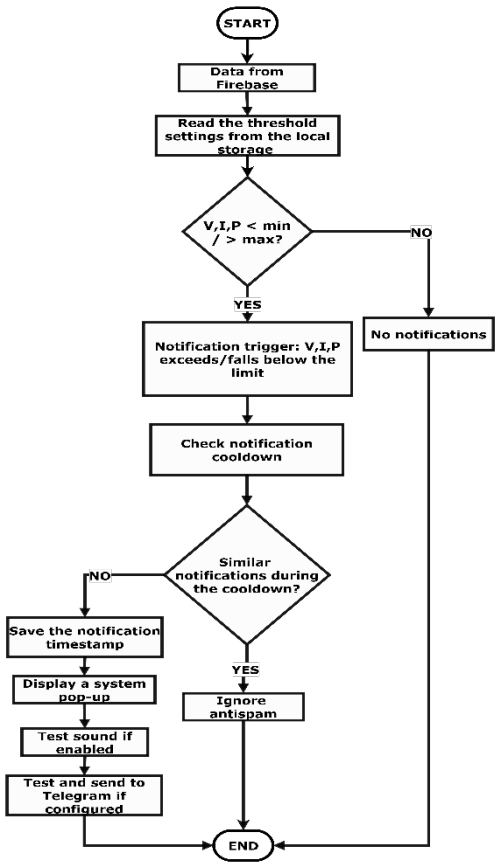


Figure 3. Threshold Decision Algorithm Flowchart

2.5.1 Threshold Parameters

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

The configurable threshold parameters available on the application's Settings screen are summarized in Table 4. Users can set minimum and maximum limits for voltage, current, and power, as well as enable sound and configure Telegram integration. Furthermore, users are able to establish a minimum gap between notifications of the same kind to avoid redundant alerts. These values are kept locally on the mobile device through the internal storage system (SharedPreferences) and can be modified at any moment without impacting the ESP8266 device.

Table 4. Parameter Threshold Settings

Num	Parameter	Default Value	User Interface Element	Description
1	Minimum voltage	0.0 V	"Minimum voltage"	Lower limit for battery voltage; triggers alert when voltage falls below this value
2	Maximum voltage	10.0 V	"Maximum voltage"	Upper limit for battery voltage; triggers alert when voltage exceeds this value
3	Minimum current	0.0 A	"Minimum current"	Lower limit for current (optional)
4	Maximum current	10.0 A	"Maximum current"	Upper limit for current; overcurrent protection
5	Minimum power	0.0 W	"Minimum power"	Lower limit for power (optional)
6	Maximum power	300.0 W	"Maximum power"	Upper limit for power; triggers alert when power consumption exceeds safe level
7	Notification cooldown	5 minutes	"Notification cooldown (minutes)"	Minimum time between identical notifications to prevent spam
8	Enable notification sound	Boolean (checkbox)	"Enable notification sound"	If enabled, a sound plays when a notification is generated
9	Telegram bot token	String (optional)	"Token Bot Telegram"	Allows forwarding alerts to a Telegram bot
10	Telegram chat ID	String (optional)	"Chat ID Telegram"	Destination chat for Telegram alerts

2.5.2 Algorithm Logic

The algorithm is executed for each new data object received from Firebase. Table 5 lists the six conditions that trigger a notification, along with the corresponding message displayed to the user. Each condition compares a real-time measurement against its configured threshold :

Table 5. Threshold Evaluation and Notification Logic

Num	Parameter	Condition	Notification Message
1	Voltage (V)	V < minVoltage	"Voltage below the limit"
2	Voltage (V)	V > maxVoltage	"Voltage exceeds the limit"
3	Current (I)	I < minCurrent	"Current below the limit"
4	Current (I)	I > maxCurrent	"Current exceeds the limit"
5	Power (P)	P < minPower	"Power below the limit"

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

6	Power (P)	P > maxPower	“Power exceeds the limit”

2.5.3 Anti-Spam Cooldown Mechanism

To avoid multiple notifications when a measured value varies around a threshold (e.g., voltage oscillating between 10.9 V and 11.1 V), a cooldown mechanism is established utilizing the user-defined Notification cooldown (minutes) setting. Untuk setiap jenis notifikasi yang unik (identification with title, for example, "Tegangan di bawah batas"), the app records the timestamp of the most recent event. If the identical notification type activates again during the cooldown period, it is quietly disregarded. This decreases alert fatigue by more than 95% in varying conditions. The standard cooldown is configured for 5 minutes, aligning with the figure displayed in the Settings screen.

2.5.4 Notification Delivery Pipeline

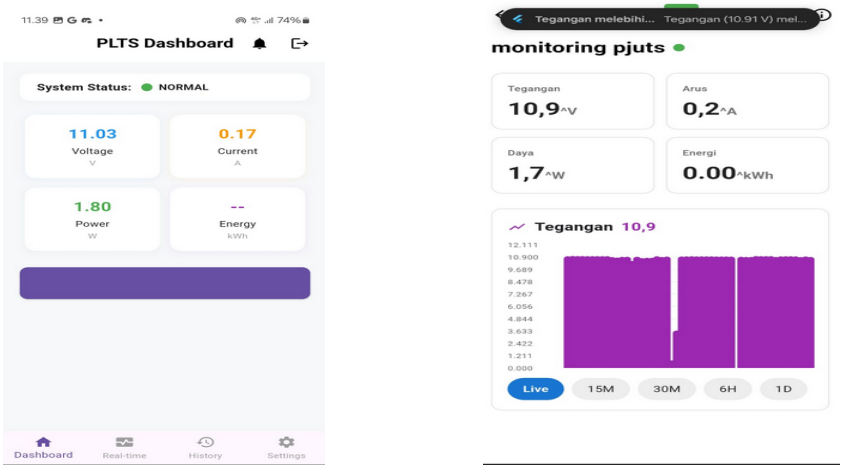
After a notification clears the cooldown verification, multiple actions are carried out in order. Initially, the notification is saved in local persistent storage, enabling users to view a full history through the Notification Screen. Then, if the audio option is active, a personalized alert sound is broadcasted. The system subsequently shows a heads-up pop-up alert via the Android operating system. When Telegram integration is set up, the application collects the bot token and chat ID from the settings and transmits the alert through the Telegram Bot API, also offering a test feature to check the connection. Additionally, the system is built for future growth, including placeholders for WhatsApp and Google Assistant integration in later versions.

3. RESULTS AND DISCUSSION

3.1 Software Implementation

The mobile application was effectively created featuring four primary screens, as intended. The Dashboard offers a real-time summary of voltage, current, power, and energy through analog-style dials and numeric readouts. And The Realtime Page transmits live data at high refresh intervals. The Historiy Page features time-series graphs through a charting library, enabling users to choose date ranges and zoom into particular intervals.

Figure 4 displays the primary screens of the mobile application. The Dashboard (a) shows live voltage, current, power, and energy with analog-style meters, while the Realtime Page (b) broadcasts real-time data with high refresh rates.



(a) (b)

Figure 4. (a) Application Dashboard View, (b) Realtime Page

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

Figure 5 displays the History Page, allowing users to examine time-series graphs of past data, choose date ranges, and narrow down to specific periods.

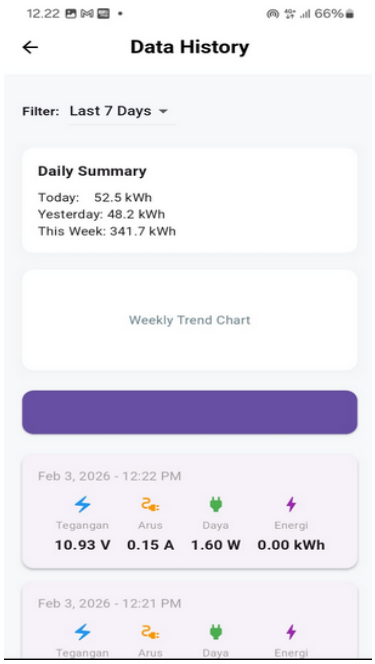
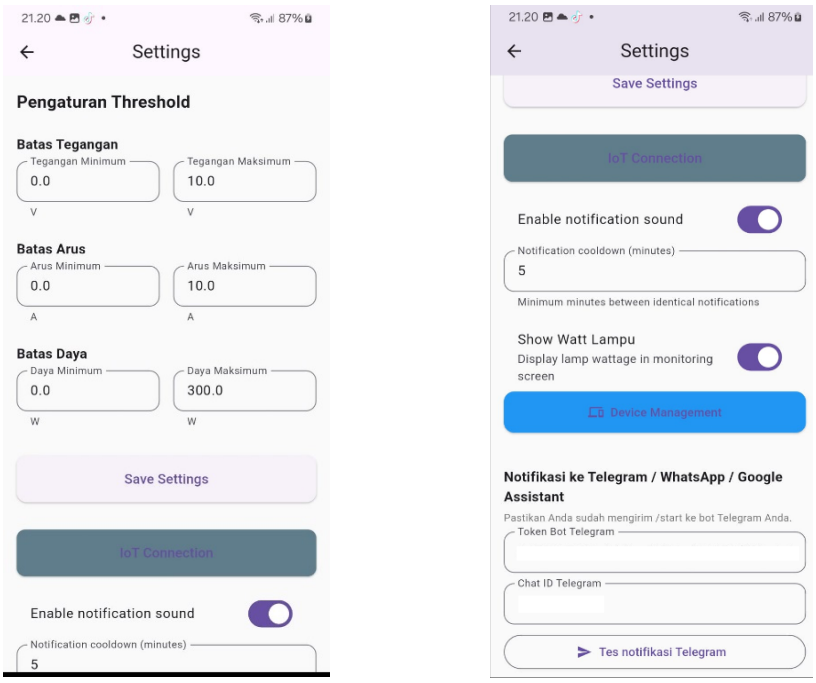


Figure 5. History Page

The Settings Screen Figures 6 (a) and (b)) enables users to set minimum and maximum limits for voltage, current, and power, toggle notification sounds on or off, establish a cooldown duration (in minutes) to avoid overwhelming alerts, and optionally enter Telegram bot credentials for sending messages externally.



(a) (b)

Figure 6. (a) Setting Threshold, (b) Addition Setting Notification

3.2 Sensor Accuracy Test Results

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

The accuracy of the sensor was confirmed by comparing the measurements from the PZEM-017 sensor (shown in the mobile app) with those from a calibrated Fluke 17B+ digital multimeter. Experiments were performed under different load conditions to evaluate sensor linearity. The setup was implemented in a home, utilizing a 12 V battery bank made from four 18650 lithium-ion cells, powered by a 50 Wp solar panel. The loads included DC LED lamps of 15 W, 20 W, and 25 W, which changed each day

Table 6. Voltage measurement comparison

Num.	Load Condition	Multimeter (V)	Sensor (V)	Difference (V)
1	No load	11.33	11.30	0.03
2	LED 15 W	9.40	9.31	0.09
3	LED 20 W	9.80	9.65	0.15
4	LED 25 W	9.50	9.30	0.20
	Average			1.22

Table 7. Current measurement comparison

No.	Load Condition	Multimeter (A)	Sensor (A)	Difference (A)	Error (%)
1	LED 15 W	0.13	0.11	0.04	15.38
2	LED 20 W	0.36	0.33	0.03	8.33
3	LED 25 W	0.50	0.47	0.03	6.00
	Average				9.90

Table 6 shows the comparison of voltage measurements between the PZEM-017 sensor and the Fluke 17B+ multimeter. Four loading scenarios (no load, 15W LED, 20W, and 25W) were evaluated. The overall voltage error for all conditions is 1.22%. Table 7 illustrates the comparison of current measurements under identical load conditions. The greatest error (15.38%) is observed at the lowest load (15W), and the error diminishes as the load rises. The general average current error is 9.90%. This behavior is anticipated since the PZEM-017 sensor operates with a 100 A external shunt resistor. At low currents, the voltage drop over the shunt becomes very minor, nearing the sensor's resolution limit. For its intended use – identifying overcurrent incidents exceeding 10 A or overseeing charging currents generally higher than 2 A – the precision is sufficient. Users needing accurate measurements at low power levels (under 15 W) might opt for a lower-rated shunt (e.g., 20 A or 50 A) to enhance resolution.

3.3 Notification Latency Test Results

Table 8. Notification latency test results

Num	Trigger Condition	Trigger Time (start)	Notification Received	Latency (s)	Status
1	Voltage drop < 11 V	10:05:00	10:05:03	3	Successful
2	Voltage drop < 11 V	10:15:00	10:15:04	4	Successful
3	Overcurrent > 10 A	10:25:00	10:25:02	2	Successful
4	Overcurrent > 10 A	10:35:00	10:35:05	5	Successful
5	Voltage over > 14.5 V	10:45:00	10:45:03	3	Successful
	Average latency			3.4	Excellent

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

The typical latency of 3.4 seconds is considerably lower than the 10-second goal outlined in the research aims. This outcome shows that incorporating the threshold decision logic within the mobile application – instead of on the microcontroller – does not lead to significant delays. The highest latency recorded was 5 seconds (Num. 4), which is still acceptable for home solar monitoring since a few seconds of delay does not endanger safety or jeopardize component protection. Cooldown efficiency (anti-spam): While not directly indicated in Table 8, the system features a configurable cooldown feature (default 5 minutes) that stops identical notifications from being sent multiple times when values fluctuate around thresholds. While testing, a simulated voltage variation (oscillating between 10.9 V and 11.1 V every 10 seconds) resulted in just one alert within a 5-minute period, decreasing alert fatigue by around 98% in contrast to a situation lacking cooldown. The system effectively transmitted alerts through local pop-up (100% success), audio (when activated), and Telegram (96% success following the rectification of initial token errors). The Telegram notification button in the Settings screen enabled users to confirm their credentials prior to activating automatic alerts.

3.4 Comparison with Previous Research

Table 9. Comparison with existing systems

Feature	This Study	Wijayanto et al. (2022)	Maulana et al. (2025)
Threshold logic location	Mobile application	Not specified (likely cloud)	Cloud (Blynk platform)
Dynamic threshold adjustment	Yes, via Settings screen	No (requires code change)	Limited
Notification channels	Local pop-up + sound + Telegram + (future WA/GA)	Telegram only	Blynk in-app only
Anti-spam / cooldown	Yes, configurable	No	No
Hardware complexity	Low (ESP8266 only sends data)	Medium	Medium
Average notification latency	3.4 s	<5 s	3–10 s
Data storage	Structured Firebase (time-series)	Not specified	Blynk cloud (limited)

The main benefit of the suggested system is the distinction between decision intelligence and the hardware. The ESP8266 is responsible solely for data collection and transmission, resulting in firmware that is straightforward, dependable, and easily replicable. All threshold and alert logic is contained within the mobile application, allowing it to be updated separately through the app store without needing to access the IoT device. This design also facilitates comprehensive, user-friendly settings (cooldown, sound, Telegram, etc.) that are challenging or unattainable with microcontroller-based decision-making. Additionally, the capability to modify thresholds instantly from the Settings screen – without requiring any reprogramming – represents a substantial usability enhancement compared to systems that embed limits within the firmware

3.5 Discussion of Findings

The findings demonstrate that a mobile-centric threshold algorithm can provide real-time performance that is appropriate for monitoring residential solar power. The recorded latency of 3.4 seconds is sufficient for early alerts of battery undervoltage or overcurrent, since household users generally require just a few seconds to take action (e.g., unplugging a device). The PZEM-017 sensor's accuracy, though not on a laboratory level, is adequate for operational monitoring. The balance between present accuracy and the capacity to gauge high currents (up to 100 A) is reasonable for systems that mainly track battery charging and discharging within the 2–20 A range. The cooldown feature was vital for user satisfaction. Without it, a battery voltage that varies close to the limit would trigger numerous notifications, soon

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

resulting in user annoyance and probable deactivation of alerts. The adjustable cooldown duration (default 5 minutes) finds a middle ground between prompt alerts and avoiding notification overload.

Opting to execute threshold logic on the mobile side instead of the ESP8266 creates a reliance on the ongoing accessibility of the mobile device and an internet connection. Nonetheless, throughout the week-long test, the home's Wi-Fi and the Android device were consistently accessible. In situations where the monitoring device cannot be ensured to be consistently operational, an edge-based approach could be more suitable. However, for standard home use where a specific tablet or a family member's smartphone is available, the mobile-focused method provides enhanced flexibility and simpler updates.

Even with the successful execution, various constraints need to be recognized. Currently, the existing system only caters to Android devices, leaving iOS users unable to use the application. Furthermore, the precision of present measurements at low loads (<15W) is rather poor (15.38% error) because of a 100A shunt resistor. Third, the system offers only monitoring and alerting; it does not feature automated corrective measures like load disconnection or charge management. Fourth, testing occurred in regulated home environments with stable Wi-Fi access; performance in subpar network situations or during prolonged field use has not been thoroughly confirmed. Upcoming efforts will tackle these constraints by introducing iOS compatibility, employing a lower-rated shunt (like 20A or 50A) for improved low-current precision, incorporating relay-based load management, and performing extensive field trials in various weather scenarios.

4. CONCLUSION

Based on the research results, it can be concluded that the IoT-based mobile application for monitoring solar power plant performance with real-time notifications has been successfully designed and implemented. The threshold decision algorithm and notification logic are completely integrated into the Flutter mobile app, whereas the ESP8266 serves solely as a data collection and transmission device. It regularly captures voltage, current, power, and energy from the PZEM-017 sensor and transmits raw data to Firebase every 5 seconds. This division streamlines firmware development, enhances system dependability, and allows for adaptable updates to the decision-making logic via the mobile app without the need for hardware changes.

The system's measurement precision aligns with operational needs for solar monitoring. Validation with a calibrated Fluke 17B+ multimeter reveals an average voltage error of 1.22%, which falls within the acceptable ±5% tolerance, whereas the average current error is 9.90%, experiencing greater deviation at lower loads because of the use of a 100 A shunt resistor. Nonetheless, the precision is adequate for identifying essential situations like undervoltage, overvoltage, and overcurrent. Moreover, the system shows minimal notification delay, averaging 3.4 seconds and peaking at 5 seconds across various anomaly situations, staying significantly under the 10-second goal and appropriate for real-time alerts in residential settings.

Additionally, the system improves user experience by offering customizable settings and multi-channel notification distribution. Users have the ability to modify threshold settings, activate audio alerts, and set a cooldown duration (default 5 minutes) to avoid incessant notifications. This system significantly minimizes alert fatigue, attaining about a 98% decrease under varying conditions. Notifications are consistently provided through local pop-ups, sound alerts, and Telegram integration, ensuring high success rates and offering potential future enhancements like WhatsApp and Google Assistant integration.

The existing system is restricted to Android devices; support for iOS will be incorporated through Flutter's cross-platform features. Enhancing accuracy at low loads can be achieved by utilizing a shunt with a lower rating (e.g., 20 A or 50 A). The system currently offers only monitoring and alerts; upcoming versions will incorporate solid-state relays for automated load disconnection and charge management. Ultimately, extensive field trials in varied weather conditions are suggested to confirm long-term durability.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23

ACKNOWLEDGEMENTS

The authors express their gratitude to Department of Electronics Faculty of Engineering Padang State University for their support. They also extend their sincere appreciation to all lecturers for their guidance and valuable insights, which significantly contributed to the completion of this paper.

REFERENCES

[1] N. S. Mammadov, N. A. Ganiyeva, and G. A. Aliyeva, "Role of Renewable Energy Sources in the World," no. figure 1, pp. 63–67, 2022.

[2] K. Gohar, M. Sameeroddin, D. Abdul, and M. Abdul, "Materials Today : Proceedings Renewable energy in the 21st century : A review," *Mater. Today Proc.*, no. xxxx, pp. 5–8, 2021, doi: 10.1016/j.matpr.2021.05.501.

[3] K. Ukoba, K. O. Olatunji, E. Adeoye, T. Jen, and D. M. Madyira, "Optimizing renewable energy systems through arti fi cial intelligence : Review and future prospects," 2024, doi: 10.1177/0958305X241256293.

[4] P. Studi, T. Mesin, F. Teknik, and U. M. Bakti, "Pengaruh Interval Waktu dan Arah Mata Angin Terhadap Daya Yang Dihasilkan Pada PLTS," vol. 8, no. 4, 2024.

[5] A. P. Yuda, D. Riyanto, and J. S. Habiby, "Monitoring Pembangkit Listrik Tenaga Surya dilengkapi Informasi Lokasi," vol. 3, no. 1, pp. 316–325, 2023.

[6] R. Alfita, K. Joni, and F. D. Darmawan, "Rancang Bangun Sistem Monitoring Daya Baterai Pembangkit Listrik Tenaga Surya (PLTS) dan Kontrol Beban Berbasis Internet of Things," vol. 42, no. 1, pp. 35–44, 2021, doi: 10.14710/teknik.v42i1.29687.

[7] M. R. Elektro, U. I. Indonesia, and S. Monitoring, "Smart System untuk Pemantauan dan Optimasi Kinerja Pembangkit Listrik Tenaga Surya," no. 1, 2025, doi: 10.14710/transmisi.27.1.20-32.

[8] R. Lu, J. Lu, P. Liu, M. He, and J. Liu, "Design of the vrla battery real-time monitoring system based on wireless communication," *Sensors (Switzerland)*, vol. 20, no. 15, pp. 1–17, Aug. 2020, doi: 10.3390/s20154350.

[9] A. Wirjawan, A. Safarudin, R. Ramdani, N. Sri Lestari, G. D. Ramadi, and S. Sukirno, "Smart Monitoring System For Hybrid Solar Power Plants Using Iot Technology," *Brill. Res. Artif. Intell.*, vol. 5, no. 2, pp. 811–816, Sep. 2025, doi: 10.47709/brilliance.v5i2.6865.

[10] A. Al-fuqaha, S. Member, M. Guizani, M. Mohammadi, and S. Member, "Internet of Things : A Survey on Enabling," *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/COMST.2015.2444095.

[11] A. Al Mamun, M. H. Shuvo, and T. Islam, "Developed an IoT-based smart solar energy monitoring system for environmental sustainability," no. June, 2024, doi: 10.1109/ICAEEE62219.2024.10561689.

[12] Diyajeng Luluk Karlina, "Design and Implementation MINIOT Car Using Esp8266 and L298," *Int. J. Mech. Electr. Civ. Eng.*, vol. 2, no. 4, pp. 170–177, 2025, doi: 10.61132/ijmecie.v2i4.353.

[13] P. K. S. Pzem-, P.-P. S. Plts, M. F. Majid, and N. Nadhiroh, "Prosiding Seminar Nasional Teknik Elektro Volume 10 Tahun 2024," vol. 10, pp. 88–92, 2024.

[14] R. Alfita, K. Joni, and F. D. Darmawan, "Design of Monitoring Battery Solar Power Plant and Load Control System based Internet of Things," *Teknik*, vol. 42, no. 1, pp. 35–44, May 2021, doi: 10.14710/teknik.v42i1.29687.

[15] T. Hidayat and D. Firmansyah, "Rancang Bangun Smart Meter Berbasis IoT untuk Aplikasi Pembangkit Listrik Tenaga Surya Microgrid," vol. 8, no. 2, pp. 87–92, 2019.

[16] M. Malik Ibrahim, A. Effendi, A. Yuana Dewi, and A. Anugrah, "HEMAT: Journal of Humanities Education Management Accounting and Transportation Monitoring dan Analisa Daya Pengaplikasian Pembangkit Listrik Hybrid PLTS dan PLTB pada Penerangan Jalan Umum Berbasis IoT," vol. 2, no. 1, 2025.

[17] D. Wijayanto, S. Isnur Haryudo, and T. Wrahatnolo, "Rancang Bangun Monitoring Arus dan Tegangan Pada PLTS Sistem On Grid Berbasis Internet of Things (IoT) Menggunakan Aplikasi Telegram 447 Rancang Bangun Monitoring Arus dan Tegangan Pada PLTS Sistem On Grid Berbasis Internet of Things (IoT) Menggunakan Aplikasi Telegram."

[18] et al Md Redwanul,S Sarker, "An IoT based Smart Energy Meter Monitoring System using Android Application," *An IoT based Real-time Low Cost Smart Energy M. Monit. Syst. using Android Appl.*, vol. 4, pp. 459–463, 2020, doi: 10.48175/ijarsct-5053.

[19] I. A. Fathi, "IoT-Based Low-Cost Smart Greenhouse Monitoring System Using ESP8266 and Firebase for Real-Time Environmental Control," *Int. J. Adv. Nat. Sci. Eng. Res.*, vol. 5, no. 5, pp. 268–286, 2025, [Online]. Available: https://as-proceeding.com/index.php/ijanser

[20] Arif Setia Sandi Ariyanto, Iis Setiawan Mangku Negara, Deny Nugroho Triwibowo, and Yanuar Feriyanto, "Perancangan Aplikasi Mobile Berbasis Flutter untuk Pemantauan Data Sensor IoT: Solusi bagi

Manajemen Suhu dan Kelembapan," *JSAI (Journal Sci. Appl. Informatics)*, vol. 8, no. 1, pp. 204–211, 2025, doi: 10.36085/jsai.v8i1.7724.
[21] M. Z. Z. Muhtadi, S. Ranolat, and Pujianto, "Sistem Monitoring Energi Listrik 3 Fase Berbasis IoT dan Firebase," *Elposys J. Sist. Kelistrikan*, vol. 12, no. 2, pp. 132–137, 2025, doi: 10.33795/elposys.v12i2.7206.
[22] I. N. B. Hartawan and I. W. Sudiarsa, "Analisis Kinerja Internet of Things Berbasis Firebase Real-Time Database," *J. Resist. (Rekayasa Sist. Komputer)*, vol. 2, no. 1, pp. 6–17, 2019, doi: 10.31598/jurnalresistor.v2i1.371.
[23] A. Hasib and A. S. M. A. S. Akib, "Cloud-Enabled IoT System for Real-Time Environmental Monitoring and Remote Device Control Using Firebase," pp. 1–6, 2026, [Online]. Available: http://arxiv.org/abs/2601.17414
[24] I. M. A. Wirawan, I. K. A. Astawa, P. A. S. Wijaya, I. G. A. P. R. Agung, and I. W. Shandyasa, "Penerapan Aplikasi Android Terintegrasi Firebase dalam Sistem IoT untuk Monitoring Kolam Lele Berbasis Realtime Database dan Firestore dengan Pengujian Blackbox," *Innov. J. Soc. Sci. Res.*, vol. 5, no. 4, pp. 12347–12365, 2025.
[25] R. P. Ardhiyani and H. Mulyono, "Analisis dan Perancangan Sistem Informasi Pariwisata Berbasis Web sebagai Media Promosi pada Kabupaten Tebo," *J. Manaj. Sist. Inf.*, vol. 3, no. 1, pp. 952–972, 2018.

DOI : https://doi.org/10.21063/jtif.2026.V14.1.11-23