Sistem Monitoring Panel Surya *Off-grid* Berbasis Aplikasi Mobile di Wilayah Pertanian

Muhammad Alfath Hakim
Computer Engineering
Telkom University
Bandung, Indonesia
alfathhakim9@gmail.com

Agus Virgono
Computer Engineering
Telkom University
Bandung, Indonesia
avirgono@telkomuniversity.ac.id

Randy Erfa Saputra
Computer Engineering
Telkom Universtiy
Bandung, Indonesia
resaputra@telkomuniversity.ac.id

***Abstrak* —Keterbatasan akses jaringan listrik di kawasan pertanian terpencil sering kali menghambat pemantauan sistem Pembangkit Listrik Tenaga Surya (PLTS), sehingga pengecekan kondisi operasional menuntut kehadiran fisik di lokasi. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem monitoring berbasis perangkat lunak guna mendukung operasional PLTS pada pompa irigasi *off-grid*. Arsitektur sistem dikembangkan menggunakan *mobile application* berbasis React Native yang diintegrasikan dengan *backend* Node.js serta basis data PostgreSQL. Dalam pengoperasiannya, parameter kelistrikan yang mencakup tegangan, arus, dan daya dikirimkan melalui *gateway* ESP32 ke *broker* HiveMQ Cloud menggunakan protokol MQTT, sebelum akhirnya diproses oleh *backend* dan disajikan melalui REST API. Aplikasi ini dirancang untuk menarik data secara periodik setiap lima detik, dengan fitur utama mencakup dasbor visual, log riwayat, serta sistem peringatan dini tiga level (kritis, peringatan, dan normal). Evaluasi sistem dilakukan melalui pengujian *white-box* yang memastikan seluruh alur logika pada sebelas modul utama berjalan sesuai rancangan, pengujian *black-box* pada lima halaman aplikasi, serta *User Acceptance Test* (UAT) yang menunjukkan hasil positif dari mitra. Selain itu, sistem visualisasi data riwayat telah dioptimalkan agar mampu menyesuaikan diri secara dinamis dengan berbagai ukuran layar, disertai mekanisme proteksi data untuk menjaga stabilitas aplikasi saat terjadi transmisi data yang tidak lengkap. Berdasarkan hasil pengujian tersebut, perangkat lunak yang dikembangkan terbukti layak dan efektif sebagai solusi pemantauan mandiri untuk menunjang keberlangsungan operasional irigasi berbasis PLTS *off-grid*.**

Kata kunci— aplikasi *mobile*; React Native; Node.js; MQTT; monitoring PLTS *off-grid*; sistem peringatan dini.

I. PENDAHULUAN

Air Ketahanan pangan nasional sangat bergantung pada ketersediaan air irigasi yang berkelanjutan. Namun, para petani di kawasan *off-grid* masih menghadapi hambatan serius terkait akses air akibat tingginya biaya operasional pompa serta minimnya jangkauan jaringan listrik konvensional. Penggunaan pompa berbasis bahan bakar diesel maupun listrik prabayar PLN sering kali dinilai kurang ekonomis, terutama jika mempertimbangkan beban biaya operasional yang besar serta kendala teknis dalam penarikan jaringan kabel ke lokasi lahan yang terpencil. Kondisi tersebut memicu urgensi penggunaan Pembangkit Listrik Tenaga Surya (PLTS) *off-grid* sebagai solusi energi mandiri yang lebih efisien bagi sistem irigasi pertanian.

Optimalisasi pemanfaatan PLTS *off-grid* mensyaratkan adanya sistem pemantauan yang memungkinkan pengguna memonitor kondisi kelistrikan tanpa harus melakukan peng

awasan secara langsung ke lokasi. Kebutuhan akan sistem pemantauan jarak jauh menjadi krusial, mengingat jarak geografis antara lahan pertanian dan tempat tinggal petani yang cenderung jauh. Oleh karena itu, diperlukan sebuah sistem berbasis perangkat lunak yang mampu melakukan akuisisi, pemrosesan, penyimpanan, serta visualisasi data parameter kelistrikan secara *real-time*, disertai fitur peringatan dini (*early warning system*) untuk mendeteksi anomali pada suplai energi.

Sejumlah platform pemantauan yang tersedia secara komersial, seperti Blynk atau berbagai situs web pemantau standar, sering kali memiliki keterbatasan dalam hal skalabilitas, kustomisasi fitur, serta ketergantungan pada layanan pihak ketiga. Sebaliknya, pendekatan berbasis aplikasi *mobile* menawarkan fleksibilitas pengembangan fitur yang lebih luas, kontrol penuh terhadap arsitektur sistem, efektivitas notifikasi kondisi kritis, serta kemudahan aksesibilitas bagi pengguna di lapangan. Keunggulan inilah yang menjadi landasan pemilihan aplikasi *mobile* sebagai platform utama dalam penelitian ini [3], [4].

Penelitian ini memfokuskan pengembangan pada aspek perangkat lunak sistem *monitoring* PLTS *off-grid*, yang meliputi perancangan dan implementasi aplikasi *mobile* berbasis *framework* React Native, serta integrasi *backend* Node.js dengan basis data PostgreSQL. Tujuan utama dari riset ini adalah menghadirkan perangkat lunak yang mampu menyajikan data tegangan, arus, dan daya secara *real-time*, mengelola penyimpanan data historis, memvisualisasikannya dalam format grafik, serta mengimplementasikan sistem peringatan dini berbasis tiga level ambang batas tegangan. Lingkup pembahasan dalam artikel ini difokuskan pada aspek arsitektur, desain sistem, implementasi teknis, serta pengujian kualitas perangkat lunak yang dibangun.

II. KAJIAN TEORI

A. *Aplikasi Mobile*

React Native dipilih sebagai framework pengembangan karena kemampuannya dalam menghasilkan aplikasi lintas platform (*Android* dan *iOS*) melalui basis kode tunggal (*single codebase*). Dengan memanfaatkan mekanisme native bridge, framework ini menawarkan performa yang mendekati aplikasi native dengan siklus pengembangan yang lebih efisien. Selain itu, integrasi dengan ekosistem Expo mempermudah proses konfigurasi dan deployment, sehingga sangat relevan untuk kebutuhan aplikasi monitoring yang menuntut pembaruan data real-time yang cepat dan responsif [5], [6].

B. *Backend* dan *Databese*

Sisi *backend* bertanggung jawab atas pemrosesan logika sistem, manajemen basis data, serta penyediaan REST API untuk aksesibilitas data. Pemilihan Node.js sebagai *runtime* didasarkan pada arsitektur *event-driven* yang ringan dan efisien, sehingga sangat optimal untuk pertukaran data secara *real-time*. Sementara itu, penyimpanan data historis serta manajemen notifikasi dikelola menggunakan basis data relasional PostgreSQL, yang menjamin integritas data pengukuran sistem [9].

C. Protokol MQTT dan *Broker* HiveMQ

Komunikasi data antara perangkat lapangan dan *backend* memanfaatkan protokol MQTT (*Message Queuing Telemetry Transport*), sebuah standar komunikasi ringan dengan mekanisme *publish–subscribe* yang dioptimalkan untuk perangkat IoT. Dalam arsitektur ini, HiveMQ Cloud bertindak sebagai *broker* pusat yang mengelola transmisi data dari sensor ke *subscriber*. Sistem ini juga mengimplementasikan fitur *retained message* untuk menyimpan status data terakhir, sehingga setiap klien yang baru terhubung dapat langsung mengakses informasi terkini tanpa harus menunggu siklus pengiriman data berikutnya [7], [8].

D. Sistem Monitoring dan Peringatan Dini

Sistem ini difokuskan pada fungsi *monitoring* dan diseminasi informasi tanpa mencakup kendali otomatis terhadap komponen daya. Melalui mekanisme peringatan dini, sistem akan mengklasifikasikan status operasional berdasarkan ambang batas parameter kelistrikan yang telah ditentukan. Hal ini memungkinkan pengguna untuk melakukan tindakan tanggap secara cepat terhadap penurunan performa maupun potensi gangguan, sehingga tercipta sistem yang sederhana namun tetap memiliki keandalan operasional yang tinggi [2], [11].

III. METODE

Pengembangan Metode penelitian dilakukan melalui tahapan analisis kebutuhan, perancangan arsitektur perangkat lunak, perancangan model sistem, perancangan antarmuka, implementasi, dan pengujian. Bagian ini menguraikan setiap tahapan pada sisi perangkat lunak.

A. Arsitektur Sistem Perangkat Lunak

Arsitektur sistem dibangun melalui pendekatan alur data berjenjang yang memastikan efisiensi transmisi dan pemrosesan informasi. Data kelistrikan yang diakuisisi oleh sensor di lapangan ditransmisikan menggunakan protokol LoRa menuju *gateway* ESP32, yang kemudian mempublikasikannya ke *broker* HiveMQ Cloud melalui protokol MQTT pada *topic* lora/data. Selanjutnya, *backend* Node.js berperan sebagai *subscriber* utama yang bertugas melakukan ekstraksi, pemrosesan, dan persistensi data ke dalam basis data PostgreSQL. Dalam model ini, aplikasi *mobile* berfungsi secara eksklusif sebagai antarmuka pengguna (*client*), sementara seluruh logika pemrosesan, evaluasi ambang batas kondisi, dan manajemen data dikonsolidasikan pada sisi *backend*. Pemisahan peran ini tidak hanya meningkatkan stabilitas sistem, tetapi juga mengoptimalkan beban kerja pada sisi klien .

B. Pemilihan *Platform* dan *Framework*

Penentuan platform pemantauan dilakukan melalui studi komparatif antara tiga alternatif utama, yakni Blynk, *website monitoring*, dan aplikasi *mobile*. Evaluasi dilakukan berdasarkan pembobotan terhadap beberapa parameter kinerja sistem, dengan hasil perbandingan yang disajikan secara rinci pada Tabel I..

TABEL 1
Penilaian *Platform* Monitorinh

Parameter	Blynk	Website	Apl. Mobile
Keandalan pemantauan	8	9	9
Skalabilitas sistem	6	9	9
Kemudahan penggunaan	9	8	9
Dukungan notifikasi	8	9	9
Skor total	**73%**	**89%**	**90%**

Berdasarkan hasil analisis komparatif, aplikasi *mobile* menempati posisi teratas dengan perolehan skor 90%, sehingga ditetapkan sebagai platform utama pengembangan. Tahap selanjutnya melibatkan pemilihan *framework* pada sisi *front-end* dan *back-end*. Pada lapisan *front-end*, React Native terpilih dengan skor 92% berkat efisiensi pengembangan serta dukungan *rendering* data *real-time* dan sistem notifikasi yang superior dibandingkan Flutter (90%) maupun Ionic (72%). Sementara itu, di sisi *back-end*, Node.js menjadi pilihan utama dengan skor 87%, mengungguli Laravel dan Django (83%), karena keunggulannya dalam kemudahan implementasi, responsivitas API, integrasi ekosistem IoT, serta skalabilitas sistem yang tinggi. Matriks penilaian pemilihan *framework* tersebut disajikan pada Tabel II.

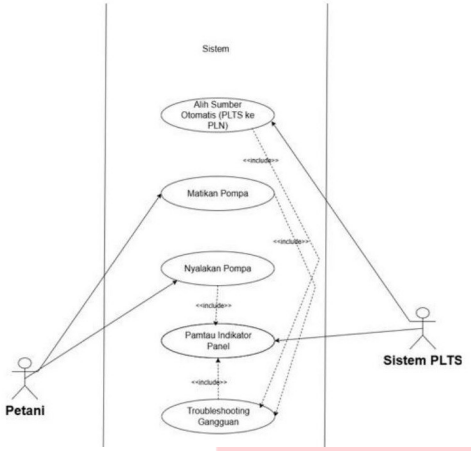
TABEL 2
Penilaian *Framework Front-end* dan *Back-end*

Kategori / Kandidat	Terpilih	Skor
Front-end (React Native, Flutter, Ionic)	React Native	92%
Back-end (Node.js, Laravel, Django)	Node.js	87%

C. Perancangan Sistem

Aspek perancangan sistem direpresentasikan melalui *Use Case Diagram*, *Sequence Diagram*, *Data Flow Diagram* (DFD), dan *Entity Relationship Diagram* (ERD). Pada Gambar 1, *Use Case Diagram* memvisualisasikan interaksi antara dua entitas, yaitu petani sebagai pengguna dan sistem PLTS sebagai aktor otomatis. Relasi *include* diterapkan pada fungsi pemantauan indikator panel, yang wajib diakses sebelum pengguna dapat mengoperasikan pompa.

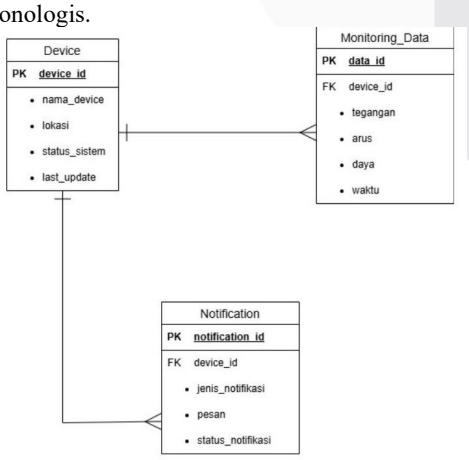
Sebaliknya, mekanisme alih sumber daya (*automatic source switching*) dikelola secara otonom oleh sistem tanpa melibatkan intervensi manual dari pengguna.



GAMBAR 1
Use Case Diagram

Alur komunikasi sistem yang diilustrasikan dalam *Sequence Diagram* bermula dari inisiasi aplikasi oleh pengguna, yang kemudian mengirimkan permintaan data ke *server*. Secara sistematis, *server* akan memproses serta memverifikasi data sebelum menyimpannya ke dalam basis data, lalu mengirimkan respons berupa data pengukuran aktual beserta status peringatan untuk ditampilkan pada antarmuka aplikasi. Melalui pemodelan *Data Flow Diagram* (DFD) Level 0 dan Level 1, ditegaskan kembali bahwa aplikasi *mobile* beroperasi sebagai media interaksi pengguna, sementara fungsi akuisisi, evaluasi logika, dan manajemen penyimpanan data dikonsolidasikan sepenuhnya di sisi *server*.

Struktur data diatur melalui *Entity Relationship Diagram* (ERD) yang melibatkan tiga entitas utama, yaitu *Device*, *Monitoring_Data*, dan *Notification* (sebagaimana ditunjukkan pada Gambar 2). Entitas *Device* berfungsi sebagai representasi fisik sistem PLTS di lapangan, yang memiliki relasi *one-to-many* terhadap entitas *Monitoring_Data* dan *Notification*. Skema ini memastikan bahwa satu unit perangkat mampu mengakomodasi riwayat pengukuran dalam jumlah besar serta pengelolaan notifikasi secara kronologis.



GAMBAR 2
Entity Relationship Diagram

D. Perancangan Antarmuka Aplikasi

Antarmuka aplikasi dirancang sederhana agar mudah digunakan di lapangan dan dibagi menjadi tiga bagian utama. Halaman *dashboard* menyajikan ringkasan nilai tegangan, arus, dan daya, serta menampilkan notifikasi Sensor *Offline* apabila data tidak masuk lebih dari batas waktu tertentu. Halaman riwayat menampilkan rekam jejak pembacaan sensor dalam bentuk grafik dengan opsi filter waktu Hari Ini, 7 Hari, dan 30 Hari. Halaman notifikasi menyampaikan peringatan kondisi sistem yang dibedakan melalui indikator warna dan ikon sesuai tingkat urgensinya.

E. Perancangan Antarmuka Aplikasi

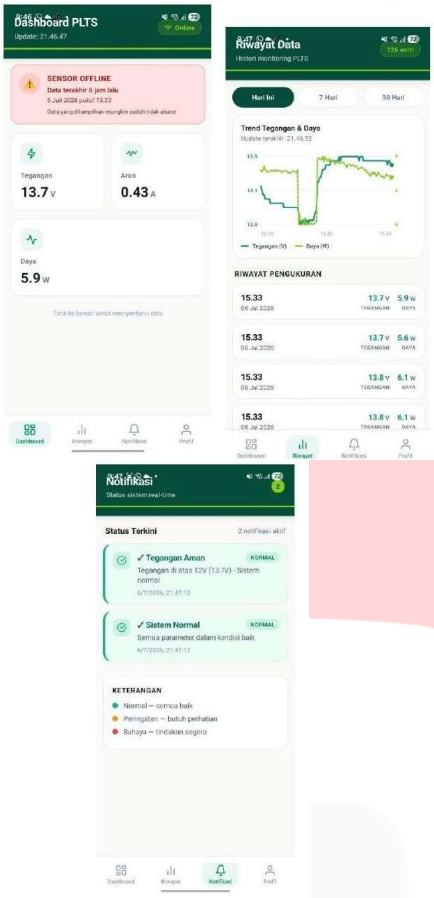
Implementasi perangkat lunak dilakukan melalui pendekatan modular guna menjaga skalabilitas dan konsistensi antarmuka, di mana konfigurasi *theme* dikonsolidasikan pada berkas Colors.js. Pada sisi *gateway*, mikrokontroler ESP32 melakukan akuisisi data dari *node* melalui komunikasi serial sebelum dipublikasikan ke *broker* HiveMQ *Cloud*. Fungsi client.publish() digunakan dengan parameter *retained* bernilai *true*, yang menjamin ketersediaan data terakhir bagi *client* saat inisialisasi koneksi.

Pada sisi aplikasi, sinkronisasi data *dashboard* diotomatisasi melalui metode setInterval dengan interval transmisi lima detik. Visualisasi data historis direpresentasikan melalui grafik berbasis vektor yang dikembangkan dalam berkas HistoryChart.js. Sistem ini menerapkan komputasi matematis untuk memetakan nilai minimum dan maksimum ke dalam koordinat layar, sehingga visualisasi bersifat responsif terhadap berbagai dimensi perangkat. Terkait sistem peringatan dini, logika generateNotifications mengklasifikasikan status kelistrikan menjadi tiga level ambang batas tegangan: kondisi kritis (< 10 V), status peringatan ($10 - 12$ V), dan kondisi normal (> 12 V). Interaksi dengan *backend* dikelola oleh api.js yang menangani pengiriman data format JSON serta injeksi token JWT pada *header* otentikasi. Selain itu, sistem telah dilengkapi dengan mekanisme *error handling* untuk mencegah kegagalan aplikasi (*crash*) apabila terjadi ketidaklengkapan metrik data dari *server*.

IV. HASIL DAN PEMBAHASAN

A. Hasil Implementasi Antarmuka

Implementasi menghasilkan aplikasi *mobile* dengan tiga halaman utama yang berfungsi sesuai rancangan. Gambar 3 menampilkan halaman *dashboard*, halaman riwayat, dan halaman notifikasi. Halaman dashboard menyajikan nilai tegangan, arus, dan daya beserta indikator Sensor *Offline*, halaman riwayat menampilkan grafik tren tegangan dan daya dengan filter waktu, dan halaman notifikasi menampilkan status sistem berdasarkan tingkat peringatan.



GAMBAR 3
Aplikasi Monitoring

B. Hasil Implementasi Antarmuka

Evaluasi kualitas perangkat lunak dilakukan melalui pengujian *white-box* untuk memvalidasi alur eksekusi logika dan integritas struktur kode pada aplikasi. Cakupan pengujian ini meliputi sebelas modul fungsional utama, yaitu: sistem autentikasi dan manajemen sesi pengguna, akuisisi data sensor *real-time*, pembaruan data periodik, mekanisme *refresh* manual, filterisasi rentang waktu data historis, serta sistem peringatan dini tiga level tegangan. Selain itu, dilakukan pengujian mendalam pada modul deteksi anomali daya dan baterai, visualisasi grafik, animasi indikator status baterai, hingga prosedur keamanan *logout*. Hasil pengujian menunjukkan bahwa seluruh jalur logika program telah tereksekusi secara utuh (100%) sesuai dengan rancangan spesifikasi, tanpa ditemukan cacat logika maupun kesalahan sintaksis. Efektivitas penanganan kesalahan (*error handling*) pada kondisi gangguan jaringan juga terverifikasi dengan munculnya notifikasi sistem saat terjadi kegagalan koneksi.

C. Pengujian *Black-Box*

Pengujian *black-box* difokuskan pada fungsionalitas antarmuka dan pengalaman pengguna terhadap lima halaman utama aplikasi. Seluruh skenario yang diuji memperoleh status sesuai, sebagaimana dirangkum pada Tabel III.

TABEL 3
Ringkasan Hasil Pengujian *Black-Box* Aplikasi

Halaman	Aspek yang Diuji	Status
Login	Validasi kredensial, penerbitan token JWT, penolakan akses tidak valid	Sesuai
Dashboard	Tampilan data real-time, notifikasi Sensor Offline, pull-to-refresh, pembaruan tiap 5 detik	Sesuai
Riwayat	Filter waktu, grafik tegangan (solid) dan daya (dashed), daftar riwayat	Sesuai
Notifikasi	Klasifikasi status tiga tingkat dengan indikator warna	Sesuai
Profil	Informasi akun dan mekanisme logout	Sesuai

D. *User Acceptance Test (UAT)*

Pengujian UAT dilakukan bersama mitra untuk memastikan sistem memenuhi kebutuhan pengguna dan mudah dioperasikan. Lima skenario diuji, yaitu kemudahan penggunaan, kejelasan informasi, manfaat sistem, kesesuaian dengan kebutuhan, dan keandalan aplikasi. Seluruh skenario memperoleh status OK. Mitra menyatakan aplikasi mudah digunakan tanpa pendampingan khusus, informasi mudah dipahami, dan sistem membantu pemantauan operasional pompa air secara *real-time*.

E. Pengujian Keamanan

Pengujian keamanan perangkat lunak dilakukan menggunakan metode *Software Composition Analysis* (SCA) melalui utilitas *npm audit* untuk mengidentifikasi potensi kerentanan pada pustaka pihak ketiga. Pemindaian dilakukan terhadap backend dan aplikasi mobile dengan mengklasifikasikan tingkat risiko ke dalam empat kategori keparahan, yaitu *low*, *moderate*, *high*, dan *critical*, sebagai dasar evaluasi dan pemutakhiran dependensi.

F. Pembahasan

Seluruh rangkaian pengujian yang dilakukan menunjukkan bahwa perangkat lunak yang dikembangkan mampu memenuhi kebutuhan pemantauan jarak jauh bagi sistem PLTS *off-grid*. Validasi *white-box* mengonfirmasi kematangan logika kode, sementara pengujian *black-box* menegaskan responsivitas antarmuka terhadap berbagai skenario penggunaan. Selain itu, *User Acceptance Test* (UAT) memberikan bukti empiris mengenai akseptabilitas sistem oleh pengguna akhir. Keunggulan teknis sistem ini meliputi mekanisme pembaruan data setiap lima detik yang mendekati *real-time*, proteksi aplikasi melalui *error handling* pada layanan API untuk menjaga stabilitas saat transmisi data tidak lengkap, serta visualisasi grafik vektor yang dinamis terhadap resolusi layar.

Namun, penelitian ini juga mengidentifikasi batasan sistem yang menjadi ruang lingkup pengembangan di masa depan. Saat ini, sistem berfokus pada pemantauan parameter kelistrikan dasar dan belum menyertakan fitur analisis performa panel surya secara otomatis berdasarkan *historical data*. Selain itu, ketergantungan pada konektivitas internet antara *gateway* dan *server* menjadi hambatan saat terjadi gangguan jaringan. Untuk pengembangan selanjutnya, disarankan untuk mengintegrasikan analisis performa yang lebih mendalam, implementasi teknik *buffering* (penyimpanan sementara) pada sisi *gateway* untuk

meminimalisir dampak putusnya koneksi, serta penguatan keamanan dependensi perangkat lunak.

V. KESIMPULAN

Riset ini telah berhasil merancang dan mengimplementasikan sistem *monitoring* PLTS *off-grid* untuk irigasi pertanian yang mengintegrasikan aplikasi *mobile* berbasis React Native, *backend* Node.js, serta basis data PostgreSQL. Sistem terbukti efektif dalam melakukan akuisisi, persistensi, dan visualisasi data kelistrikan secara *real-time* melalui protokol MQTT, disertai fitur peringatan dini berbasis tiga level ambang batas tegangan.Hasil pengujian komprehensif menunjukkan bahwa seluruh alur logika pada sebelas modul utama berhasil tervalidasi, pengujian fungsional pada seluruh halaman aplikasi dinyatakan sesuai, dan *User Acceptance Test* (UAT) memperoleh status valid (*OK*) pada semua skenario pengujian. Berdasarkan hasil tersebut, perangkat lunak yang dikembangkan dinyatakan layak sebagai media pemantauan mandiri untuk mendukung keberlangsungan operasional irigasi berbasis PLTS *off-grid*. Sebagai langkah pengembangan lanjutan, disarankan untuk mengintegrasikan analisis performa yang lebih mendalam serta mekanisme penguatan sistem terhadap fluktuasi konektivitas internet.

REFERENSI

[1] M. Miyasto, "Strategi ketahanan pangan nasional guna meningkatkan kemandirian dan daya saing ekonomi," *Jurnal Kajian Lemhannas RI*, vol. 17, pp. 17–30, Mar. 2014.

[2] A. R. Al-Ali, "*IoT-solar energy powered smart farm irrigation system,*" *Alexandria Engineering Journal*, 2019.

[3] N. Rouibah, "*Smart monitoring of photovoltaic energy systems: An IoT approach," Energy Reports*, 2025.

[4] M. A. Osman, "*Solar-powered parallel irrigation with IoT monitoring system," Journal of Engineering Science and Technology*, vol. 18, no. 5, 2023.

[5] S. Sujono et al., "Implementasi teknologi LoRa untuk monitoring PJU berbasis solar panel," *Modem: Jurnal Informatika dan Sains Teknologi*, vol. 2, 2023.

[6] N. Kumar, N. Akkamahadevi, S. Kumar, and Harish, "*LoRa based photovoltaic solar energy monitoring system," International Journal of Engineering Research & Technology* (IJERT), vol. 11, no. 3, 2022.

[7] Q. Ziyuan, D. R. Thinesh, V. Ayyalusamy, B. Sivaneasan, K. T. Tan, and K. J. Tseng, "*A LoRaWAN-based solar PV condition monitoring system," Singapore Institute of Technology*, 2023.

[8] B. H. Hameed and S. Kurnaz, "*Secure low-cost photovoltaic monitoring system based on LoRaWAN network and artificial intelligence*," *Discover Computing*, vol. 27, no. 36, 2024.

[9] V.-V. Stoica, A.-V. Pălăcean, D.-C. Trancă, and F.-A. Stancu, "*Open-source smart wireless IoT solar sensor*," *Applied Sciences*, vol. 15, no. 11059, pp. 1–33, 2025.

[10] J. Li, B. Moeini, S. Nejati, M. Sabetzadeh, and M. McCallen, "*A lean simulation framework for stress testing IoT cloud systems*," IEEE *Transactions on Software Engineering*, pp. 1–20, 2024.

[11] N. A. A. B. Suwandi et al., "Implementasi pompa air berbasis teknologi digital," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 4, no. 1, 2025..