Jurnal Ilmu Siber dan Teknologi Digital (JISTED)
ISSN 2986-7312, Vol 3, No 2, 2025, 181-202 https://doi.org/10.35912/jisted.v3i2.5393

Aplikasi Web Sistem Irigasi dan Monitoring Lahan Reklamasi Bekas Tambang Batubara

(Web-Based Irrigation and Monitoring System for Post-Coal Mine Reclamation Land)

Nazar Abdul Fattah[1*], Wahyuni Eka Sari [2], Irwansyah Irwansyah[3]
Politeknik Negeri Samarinda, Samarinda, Indonesia[1,2,3]
nazar.af.03@gmail.com[1*], wahyunisari52@gmail.com[2], irwansyah@polnes.ac.id[3]

Riwayat Artikel:
Diterima pada 05 April 2025
Revisi 1 pada 10 April 2025
Revisi 2 pada 01 Mei 2025
Revisi 3 pada 05 Mei 2025
Disetujui pada 11 Mei 2025

Abstract

Purpose: This study aimed to develop a web application for PT Insani Baraperkasa to streamline vegetation management on post-reclamation mining land. The system enhances efficiency via real-time monitoring, automated scheduling for irrigation and fertilization, and integration with Internet of Things (IoT) device control.

Methodology/approach: Built on the Waterfall model, the system uses a Laravel backend, Vue.js frontend, and MySQL with MQTT for Internet of Things communication. As a foundational study, validation focused on core functionality using white-box (PHP Unit) and black-box (Postman) testing to verify the logic and API integrity. This approach covered all 25 application endpoints, ensuring comprehensive functionality for the initial development phase.

Results/findings: Backend stability was validated by 25 successful endpoint tests, executed in 2.61s (PHPUnit) and 4.242s (Postman). Real-time IoT communication via MQTT proved to be highly responsive with low average latencies (7ms native; 89ms WebSocket), ensuring effective remote monitoring and control.

Conclusion: This study successfully developed a centralized digital tool that optimizes vegetation maintenance. The application enhances the operational efficiency of PT Insani Baraperkasa and improves the consistency of environmental management.

Limitations: The scope of this study was confined to the development of a bespoke web application and not its commercialization. Consequently, it lacks a multi-tenant architecture and native mobile app.

Contribution: This study contributes a functional and reliable IoT-based application for enhancing vegetation maintenance at PT Insani Baraperkasa. The successful deployment serves as a practical, scalable proof of concept, demonstrating the effectiveness of digital tools in managing post-reclamation land.

Keywords: *Internet of Things, Land Reclamation, Laravel, Vue.js, Website.*

How to Cite: Fattah, N. A., Sari, W. E., Irwansyah, I. (2025). Aplikasi Web Sistem Irigasi dan Monitoring Lahan Reklamasi Bekas Tambang Batubara. *Jurnal Ilmu Siber dan Teknologi Digital*, 3(2), 181-202.

1. Pendahuluan

Pertambangan merupakan sektor strategis yang memiliki kontribusi besar terhadap pertumbuhan ekonomi dan pembangunan nasional di Indonesia. Namun, di balik kontribusinya, kegiatan pertambangan juga menjadi salah satu penyebab utama kerusakan lingkungan, khususnya kerusakan hutan. Secara nasional, Indonesia terus menghadapi tantangan deforestasi, di mana data Badan Pusat

Statistik (BPS) menunjukkan rata-rata laju deforestasi netto nasional mencapai 248,419 hektar per tahun pada periode 2017-2022. Fenomena ini juga sangat relevan di Provinsi Kalimantan Timur, lokasi penelitian ini, yang menurut data analisis kehutanan mengalami deforestasi seluas 28,633 hektar sepanjang tahun 2024 (bps.go.id, 2024). Untuk mengurangi dampak tersebut, reklamasi lahan bekas tambang menjadi suatu kewajiban yang harus dilakukan oleh perusahaan tambang. PT Insani Baraperkasa, sebagai perusahaan yang bergerak di sektor pertambangan, menunjukkan komitmennya terhadap tanggung jawab lingkungan melalui pelaksanaan kegiatan reklamasi di area bekas tambangnya. Dalam pelaksanaan reklamasi, tantangan yang dihadapi tidak hanya pada aspek teknis operasional, tetapi juga pada faktor lingkungan seperti suhu, intensitas cahaya, dan kelembaban tanah yang secara langsung berpengaruh terhadap pertumbuhan tanaman dan keberlangsungan mikroorganisme di dalam tanah (Maulana Syahaddan et al., 2024). Oleh karena itu, pemantauan dan pengelolaan kondisi lingkungan tersebut secara real-time sangat diperlukan untuk mendukung keberhasilan program reklamasi.

Perkembangan teknologi informasi, khususnya Internet of Things (IoT) dan aplikasi berbasis web, membuka peluang untuk menciptakan solusi yang lebih efisien dalam pengelolaan lahan reklamasi. Penelitian ini mengembangkan aplikasi Web berbasis framework Laravel sebagai backend karena Laravel mampu meningkatkan kualitas perangkat lunak, menyederhanakan sistem otentikasi, memudahkan proses routing, serta meningkatkan performa aplikasi (Noor Saputra et al., 2023). Di sisi frontend, digunakan framework Vue.js yang memiliki kemampuan dalam pengembangan Single Page Application (SPA) dengan pengalaman pengguna yang baik dan proses caching yang cepat (Herman & Frederick, 2022). Untuk komunikasi data, digunakan protokol HTTP dan MQTT. HTTP dinilai efektif untuk komunikasi standar antara aplikasi dan microcontroller (Nugraha et al., 2024), sedangkan MQTT dipilih karena keunggulannya dalam komunikasi data ringan dan efisien, bahkan dalam kondisi jaringan yang terbatas, serta terbukti 93 kali lebih cepat dalam pengiriman data pada jaringan 3G (Utamy et al., 2023). Untuk memastikan bahwa klaim performa tersebut relevan dalam konteks sistem yang diimplementasikan, penelitian ini tidak hanya bersandar pada studi literatur tetapi juga melakukan validasi kinerja melalui serangkaian pengujian eksperimental. Pengujian dirancang untuk mengevaluasi responsivitas dan latensi komunikasi dari arsitektur yang diusulkan. Dengan dukungan dua protokol tersebut, aplikasi yang dikembangkan dapat bekerja secara efisien dan responsif.

Menindaklanjuti celah penelitian yang teridentifikasi, di mana platform digital terpadu untuk manajemen reklamasi lahan pascatambang masih langka, penelitian ini menguraikan perancangan sistem komprehensif untuk mentransformasi praktik konvensional menjadi operasional berbasis data. Studi ini secara spesifik bertujuan menjawab bagaimana arsitektur teknologi *Internet of Things* (IoT) dapat diimplementasikan untuk pemantauan biofisik lahan secara *real-time*, bagaimana aplikasi dapat dirancang untuk menjamin integritas data penggunaan sumber daya, serta sejauh mana sistem ini mampu meningkatkan efisiensi operasional dan efektivitas ekologis. Ruang lingkup penelitian difokuskan secara eksklusif pada aspek teknis, mencakup perancangan arsitektur, implementasi *backend* menggunakan *framework Laravel*, pengembangan *frontend* interaktif dengan *Vue.js*, serta pemanfaatan protokol HTTP dan MQTT untuk komunikasi, dengan mengecualikan analisis kelayakan finansial. Dengan demikian, kontribusi utama dari penelitian ini adalah mewujudkan sebuah artefak teknologi berupa aplikasi berbasis situs web yang *robust*, fungsional, dan telah teruji, yang menyediakan modul terintegrasi seperti penjadwalan penyiraman dan pemupukan yang adaptif, pencatatan sumber daya yang akurat, dasbor pemantauan visual, dan kontrol perangkat jarak jauh. Secara kolektif, sistem ini diharapkan tidak hanya menjadi aset operasional strategis bagi PT Insani Baraperkasa, tetapi juga berfungsi sebagai sebuah cetak biru teknologi yang dapat direplikasi untuk mendukung akselerasi transformasi digital dalam industri pertambangan yang lebih luas.

2. Tinjauan Pustaka

2.1 Kajian Ilmiah

Penelitian sebelumnya telah banyak mengeksplorasi pengembangan sistem monitoring berbasis Internet of Things (IoT) dan memberikan fondasi teknologi yang penting. Penelitian berjudul "Implementasi RESTful API Pada Laravel dan Simulator IoT Wokwi Untuk Pengukuran Suhu dan Kelembaban Menggunakan Metode Waterfall" berhasil mendemonstrasikan kelayakan teknis perancangan RESTful API menggunakan framework Laravel, namun validasinya terbatas pada lingkungan simulator menggunakan IoT Wokwi (Ardhana et al., 2023). Pendekatan serupa dalam pemanfaatan API juga dilakukan dalam penelitian "Development of Application Programming Interface (API) for AMIKOM Purwokerto Handsanitizer (AMPUH) Data Logger Visualization", yang berfokus pada pengolahan data mentah dari platform data logger pihak ketiga (ThingSpeak) menjadi API menggunakan Laravel, sehingga data dapat disajikan dalam visualisasi web yang lebih informatif bagi tim frontend. Di sisi penerapan praktis (Nur et al., 2022). Di sisi penerapan praktis, penelitian "Pendampingan penerapan teknologi sistem monitoring dan penyiraman berbasis IoT pada budidaya tanaman obat keluarga" berhasil mengimplementasikan sistem penyiraman otomatis yang menggabungkan Arduino dan NodeMCU, akan tetapi, sistem ini bergantung sepenuhnya pada aplikasi smartphone pihak ketiga, yaitu Blynk, sebagai antarmuka untuk memantau dan mengendalikan perangkat (Sayekti et al., 2022). Penelitian lain berjudul "Monitoring Suhu Dan Kelembaban Tanah Tanaman Buah Naga Berbasis IoT" juga berfokus pada monitoring kelembaban tanah yang dikontrol oleh mikrokontroler untuk melakukan penyiraman otomatis secara real-time, dengan data sensor yang dikirim ke server untuk pemantauan jarak jauh (Hidayat et al., 2020). Sementara itu, penelitian "Perancangan dan Implementasi Protokol MQTT pada Sistem Parkir Cerdas Berbasis IoT" secara spesifik mengimplementasikan protokol MQTT untuk komunikasi data dari sensor ke website, dengan penekanan pada pengujian waktu respons sistem untuk terhubung ke broker (R. F. Pratama et al., 2023). Berdasarkan analisis terhadap kajian tersebut, terlihat bahwa penelitian-penelitian sebelumnya telah membuktikan efektivitas komponen teknologi secara terpisah, namun menyisakan celah. Terdapat kebutuhan untuk sebuah sistem yang tidak hanya bergerak dari simulasi ke penerapan fisik di dunia nyata, tetapi juga mengintegrasikan perangkat keras di lapangan dengan sebuah ekosistem aplikasi web yang komprehensif, custom-built, dan tidak bergantung pada platform pihak ketiga. Oleh karena itu, penelitian ini berfokus pada pengembangan sebuah sistem aplikasi web end-to-end dengan arsitektur headless (Laravel dan Vue.js) yang fleksibel dan scalable. Sistem ini dirancang untuk menjawab kebutuhan operasional industri yang kompleks pada lahan reklamasi pascatambang dan kinerjanya divalidasi secara kuantitatif untuk memastikan keandalannya.

2.2 Dasar Teori

2.2.1 Website

Website adalah salah satu aplikasi yang berisikan dokumen-dokumen multimedia (teks, gambar, suara, animasi, video) di dalamnya yang menggunakan protokol HTTP (hyper transfer protokol) dan untuk mengakses menggunakan perangkat lunak yang disebut browser. Fungsi website diantaranya yaitu Media promosi, pemasaran, informasi, pendidikan dan komunikasi (Syahril & Ramdhani, 2024). Pengembangan website untuk sistem monitoring dan pemyiraman lahan reklamasi bekas tambang dipilih karena memberikan akses universal melalui browser tanpa instalasi aplikasi khusus, memungkinkan visualisasi data monitoring dalam bentuk dashboard yang intuitif, serta dapat diakses dari berbagai perangkat baik desktop maupun mobile untuk memudahkan monitoring di lapangan.

2.2.2 Linux

Linux merupakan sebuah sistem operasi berbasis *kernel* yang *script* pertamanya diciptakan oleh seorang mahasiswa asal Finlandia, Linus Torvalds, untuk arsitektur Intel 80386. Setelah kemunculannya di *Internet* pada tahun 1991, banyak individu dari berbagai belahan dunia turut memainkan peran krusial dalam proses pengembangan dan ekspansi *Linux* secara global (Handayani et al., 2023). Linux dipilih sebagai sistem operasi server monitoring lahan reklamasi karena sifatnya yang open source dan gratis, memiliki stabilitas tinggi untuk operasi 24/7, mendukung berbagai layanan server seperti web server

dan MQTT broker, serta menyediakan keamanan yang baik dengan kontrol akses yang ketat untuk melindungi data monitoring yang sensitif.

2.2.3 API (Application Programming Interface)

API atau *Application Programming Interface* merupakan sebuah antarmuka yang sengaja dibangun oleh pengembang agar fungsionalitas sistemnya dapat diakses oleh aplikasi lain secara terprogram. Kehadiran *API* ini menjadi salah satu faktor yang turut menentukan performa sistem, karena ia menyediakan sekumpulan prosedur standar bagi aplikasi eksternal untuk memanfaatkan layanan yang ada (Barus et al., 2021). Implementasi API dalam sistem monitoring lahan reklamasi berfungsi sebagai penghubung antara frontend Vue.js dengan backend Laravel untuk pertukaran data, menyediakan fondasi yang siap untuk diintegrasikan dengan sistem eksternal atau aplikasi mobile di masa depan, serta menyediakan endpoint yang terstruktur untuk mengakses data sensor IoT dan informasi reklamasi secara programatis.

2.2.4 Protokol MQTT (Message Queuing Telemetry Transport)

Protokol *MQTT* adalah protokol komunikasi yang menggunakan mekanisme publikasikan atau berlangganan terhadap permintaan/tanggapan. Ada empat properti pada protokol *MQTT*: *IP*, *Payload*, *Topic*, dan *Port*. Protokol *MQTT*, ada tiga agen: Penerbit, Pelanggan, dan Pialang. *Publisher* bekerja dengan menghubungkan klien ke broker dan mengirimkan pesan ke broker. Pelanggan bekerja dengan menghubungkan klien ke broker dan menerima pesan tentang topik yang diinginkan (R. R. Pahlevi et al., 2019). Implementasi MQTT dalam sistem monitoring lahan reklamasi bekas tambang dipilih karena protokol ini dikenal luas sangat efisien untuk lingkungan IoT (R. F. Pratama et al., 2023). Selain efisiensinya yang telah terdokumentasi, protokol ini juga mendukung sistem publish-subscribe yang memungkinkan multiple devices mengirim data monitoring secara simultan, serta memiliki mekanisme Quality of Service (QoS) yang menjamin pengiriman data.

2.2.5 Protokol Websockets

Protokol *WebSocket* merupakan sebuah protokol yang dikembangkan dalam *HTML5*. Protokol ini memungkinkan komunikasi dua arah secara bersamaan (*bidirectional full duplex*). Hal ini berarti server dan klien dapat mengirim dan menerima data secara bersamaan tanpa harus menunggu respons dari pihak lain. *WebSocket* awalnya dirancang untuk digunakan di web browser dan web server, namun juga dapat digunakan oleh aplikasi klien atau server lainnya (Eka Putra et al., 2024). Penggunaan *WebSocket* dalam sistem monitoring memungkinkan komunikasi real-time antara server dan web application untuk menampilkan data reklamasi yang terupdate secara langsung, dan mendukung implementasi *MQTT* over *WebSocket* yang memfasilitasi integrasi data sensor *IoT* dengan web.

2.2.6 PHP (Hypertext Preprocessor)

PHP adalah sebuah bahasa pemrograman *server side* yang bersifat *open source* atau gratis, dan umumnya dimanfaatkan untuk mengolah informasi melalui halaman *web* di internet. Nama *PHP* sendiri merupakan singkatan resmi dari *Hypertext Preprocessor*, yang menegaskan fungsinya sebagai bahasa yang dieksekusi di sisi *server* (O. Pahlevi et al., 2018; Utomo et al., 2022). Pemilihan PHP sebagai bahasa pemrograman backend sistem monitoring lahan reklamasi bekas tambang didasarkan pada kemudahan integrasi dengan berbagai database untuk menyimpan data monitoring, sifat open source yang menghemat biaya pengembangan, serta dukungan komunitas yang luas dengan dokumentasi lengkap yang memudahkan pemeliharaan dan pengembangan sistem.

2.2.7 Javascript

Javascript merupakan salah satu bahasa pemrograman yang sangat dikenal dan umum digunakan untuk mengembangkan aplikasi berbasis *website*. Sejak pertama kali dibuat pada tahun 1995, bahasa ini telah diadopsi secara luas oleh banyak pengembang dalam pembuatan berbagai *website* maupun aplikasi (Christian & Hengky, 2023). JavaScript dipilih untuk pengembangan frontend sistem monitoring karena kemampuannya dalam menciptakan interface yang interaktif dan responsif, mendukung pemrosesan data real-time untuk menampilkan informasi yang dinamis, serta kompatibilitasnya yang baik dengan berbagai browser modern yang memungkinkan akses sistem dari berbagai perangkat di lapangan.

2.2.8 Framework Vue.js

Vue.js merupakan salah satu *framework javascript* untuk sisi *front end* yang diciptakan pada tahun 2013 oleh Evan You, yang disebut sebagai pembuat *framework angular*. *Framework* ini secara spesifik dimanfaatkan dalam pengembangan aplikasi web berjenis *Single Page Application* (Mufti Prasetiyo et al., 2022). Pemilihan Vue.js dalam pengembangan sistem monitoring lahan reklamasi bekas tambang dikarenakan *framework* ini menyediakan *reactive data binding* untuk pembaruan data *real-time*, serta mendukung *reusable components* yang efektif untuk membangun *interface monitoring* yang konsisten seperti dashboard dan grafik data reklamasi.

2.2.9 Framework Laravel

Laravel adalah sebuah *framework* berbasis PHP yang bertujuan untuk menyederhanakan proses pembuatan aplikasi *web*. Hal ini dicapai dengan menawarkan serangkaian perangkat dan aturan standar yang berfungsi untuk meringankan beban tugas-tugas rutin dalam pengembangan *web* (Ardhana et al., 2023). Penggunaan Laravel sebagai backend framework dipilih karena menyediakan *Eloquent ORM* (*Object-Relational Mapping*) yang memudahkan pengelolaan database monitoring lahan reklamasi, fitur *built-in authentication* dan *authorization* untuk keamanan sistem, serta struktur MVC (*Model View Controller)* yang terorganisir dengan baik untuk pengembangan *API* yang akan dikonsumsi oleh *frontend* Vue.js dalam menampilkan data reklamasi tambang.

2.2.10 Database

Database adalah suatu koleksi terstruktur dari tabel-tabel yang saling terhubung melalui penggunaan kunci relasional. Selain itu, sebuah *database* dapat mencakup keseluruhan populasi data yang relevan bagi satu unit kerja tertentu dalam sebuah lembaga, perusahaan, atau organisasi (Gede Endra Bratha, 2022). Database diperlukan dalam sistem monitoring lahan reklamasi bekas tambang untuk menyimpan data sensor IoT secara terstruktur, menyediakan sistem relasi antar tabel yang memungkinkan pengelolaan data lokasi, sensor, dan hasil monitoring secara efisien, serta memfasilitasi penyimpanan historis data reklamasi untuk analisis trend dan pelaporan jangka panjang.

2.2.11 Structured Query Language (SQL)

Bahasa yang kini dikenal sebagai *Structured Query Language* atau *SQL* pertama kali dikembangkan pada tahun 1970 dengan nama *SEQUEL*. Proses pengembangannya dilakukan di IBM oleh Donald D. Chamberlin dan Raymond F. Boyce, yang mendasarkan karyanya pada model manajemen basis data relasional temuan E. F. Codd. Pada masanya, bahasa ini secara spesifik digunakan untuk memanipulasi dan menarik data dalam *database management system* IBM yang bernama *System R* (Suliyanti, 2019). SQL digunakan dalam sistem monitoring lahan reklamasi untuk melakukan query data sensor secara efisien, memungkinkan agregasi dan analisis data monitoring dengan operasi JOIN antar tabel, serta menyediakan kemampuan filtering dan sorting data berdasarkan parameter waktu, alat, atau jenis sensor untuk mendukung dashboard monitoring yang responsif.

2.2.12 Web Server

Peran sebuah *web server* adalah sebagai *software* yang memproses permintaan dari klien untuk mengakses berkas-berkas pada sebuah situs *web*. Proses pengiriman data ini berlangsung menggunakan protokol *HTTP* ataupun *HTTPS* dan diterima oleh pengguna melalui aplikasi peramban *web*. Tulisan sumber juga menyamakan istilah ini dengan perangkat lunak yang berfungsi agar satu unit perangkat keras dapat mengoperasikan beberapa sistem operasi secara serentak (Cahyadi et al., 2023). Web server diperlukan dalam sistem monitoring lahan reklamasi bekas tambang untuk melayani request HTTP/HTTPS dari aplikasi Vue.js, menjalankan aplikasi Laravel backend yang memproses data sensor IoT, serta menyediakan akses yang aman dan stabil bagi pengguna untuk mengakses dashboard monitoring dari berbagai lokasi secara real-time.

2.2.13 Pengujian Black Box

Pengujian *Black Box* merupakan pendekatan dalam evaluasi perangkat lunak yang tidak menuntut penguji untuk memahami struktur kode internal. Penguji hanya berfokus pada kesesuaian keluaran (*output*) dengan hasil yang diharapkan berdasarkan spesifikasi fungsional (Rafiussani & Armin, 2024). Pada proyek ini, diterapkan metode *Equivalence Partitioning* untuk efisiensi pengujian. Teknik ini

bekerja dengan cara mengelompokkan data masukan ke dalam partisi valid dan tidak valid. Dengan asumsi bahwa semua data dalam satu partisi akan menghasilkan perilaku yang sama, pengujian cukup dilakukan pada satu sampel dari tiap partisi, sehingga tidak perlu menguji setiap kemungkinan masukan (Amalia et al., 2021). Selain itu, pengujian juga dilakukan untuk mengevaluasi kualitas aplikasi dengan meguji performa dan fungsionalitas aplikasi (Putra et al., 2025). Pengujian ini bertujuan untuk memastikan aplikasi dapat memberikan respons yang baik dengan waktu yang relatif cepat, serta stabil saat digunakan.

2.2.14 Pengujian White Box

Pengujian White Box merupakan sebuah metode pengujian perangkat lunak yang berfokus pada struktur internal kode untuk menemukan *error*. Metode yang sangat efektif untuk pengujian level unit ini memiliki tingkat deteksi *error* hingga 65% dan sering kali menggunakan teknik *basis path testing* untuk memastikan setiap alur kontrol telah teruji (Maspupah, 2024). Pada proyek ini, pengujian White Box diterapkan melalui pendekatan *unit testing*. Artinya, setiap komponen atau modul program diuji secara individual terlebih dahulu sebelum digabungkan menjadi satu sistem utuh. Fokus utama pengujian ini adalah untuk memastikan setiap modul berfungsi sesuai dengan spesifikasi fungsionalnya dan untuk mengidentifikasi adanya penyimpangan dari yang seharusnya (Selatan, 2020). Melalui pengujian ini, kesalahan logika dalam kode dapat ditemukan sejak awal, sebelum sistem diintegrasikan secara keseluruhan.

3. Metodologi Penelitian

3.1 Lokasi dan Waktu Penelitian

Penelitian ini berfokus pada pengembangan sistem untuk mendukung operasional reklamasi lahan pascatambang, sebuah isu krusial dalam industri pertambangan Indonesia. Lokasi penelitian ditetapkan di PT. Insani Baraperkasa, yang secara administratif berada di Purwajaya, Loa Janan, Kabupaten Kutai Kartanegara, Kalimantan Timur. Pemilihan lokasi ini didasarkan pada relevansi kegiatan reklamasi yang sedang aktif dilakukan oleh perusahaan, sehingga menyediakan studi kasus yang nyata dan aplikatif. Objek utama penelitian adalah sistem monitoring manual yang ada, dengan tujuan untuk mentransformasikannya menjadi sistem terpadu berbasis IoT. Seluruh rangkaian kegiatan penelitian, mulai dari analisis awal hingga penyebaran sistem, dilaksanakan dalam rentang waktu enam bulan, yaitu dari bulan Mei hingga Oktober 2024.

3.2 Objek Penelitian

Objek penelitian ini terbagi menjadi dua komponen utama: konteks permasalahan dan solusi teknologi yang dikembangkan. Objek utama yang menjadi konteks permasalahan adalah lahan reklamasi pascatambang di PT. Insani Baraperkasa, yang memerlukan sistem monitoring dan irigasi terpadu untuk mendukung keberhasilan proses revegetasi. Gambar 1 di bawah ini menampilkan kondisi aktual dari objek lahan tersebut. Area yang luas dan terbuka ini menunjukkan tantangan lingkungan yang dihadapi, seperti paparan langsung terhadap cuaca dan kebutuhan akan penyiraman yang merata. Kondisi ini menegaskan urgensi implementasi sistem monitoring yang andal untuk mengelola sumber daya air secara efisien dan memastikan pertumbuhan vegetasi yang optimal.

Gambar 1 Lahan Reklamasi Bekas Tambang

Sebagai solusi untuk mengelola lahan tersebut, dikembangkan sebuah sistem irigasi dan monitoring berbasis IoT yang menjadi objek pengembangan utama dalam penelitian ini. Perangkat ini berfungsi sebagai instrumen untuk menerapkan intervensi teknologi pada objek lahan. Gambar 2 menampilkan wujud fisik dari perangkat keras yang diimplementasikan di lapangan. Perangkat ini dirancang sebagai unit mandiri yang ditenagai oleh panel surya, dilengkapi dengan tandon air, sistem pompa, dan *sprinkler*. Seluruh operasionalnya dikendalikan oleh mikrokontroler yang ditempatkan di dalam kotak panel, yang berfungsi untuk mengeksekusi jadwal penyiraman otomatis dan berkomunikasi secara *real-time* dengan aplikasi web.

Gambar 2 Perangkat Irigasi dan Monitoring yang Digunakan

3.3 Alat dan Bahan

Proses perancangan, pengembangan, dan pengujian sistem monitoring lahan reklamasi dalam penelitian ini memerlukan dukungan perangkat keras (*hardware*) dan perangkat lunak (*software*). Perangkat keras digunakan untuk menjalankan aplikasi, sementara perangkat lunak digunakan untuk pengembangan aplikasi, pengelolaan basis data, dan pengujian sistem. Spesifikasi lengkap dari alat dan bahan yang digunakan disajikan pada Tabel 1.

Tabel 1. Kebutuhan Alat dan Bahan Penelitian

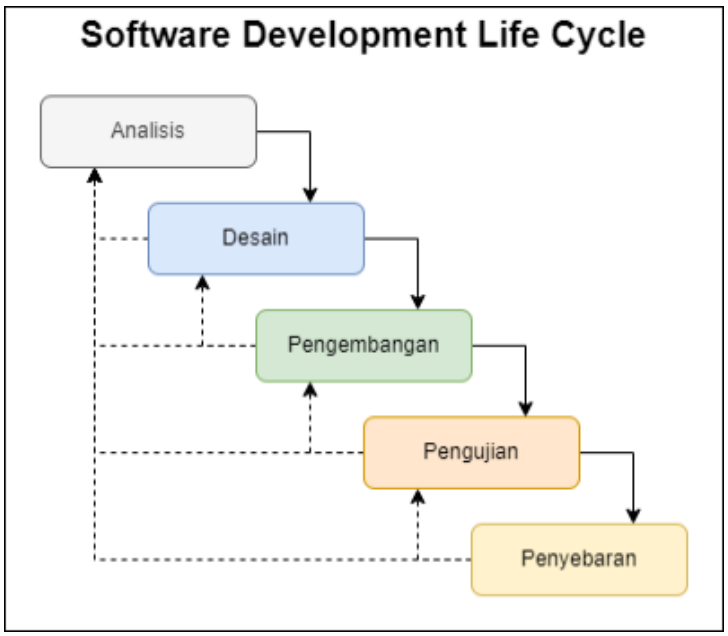
No	Nama	Kategori	Keterangan
1	Sistem Operasi	Perangkat Lunak	Sistem operasi Windows untuk tahap pengembangan, dan Linux untuk tahap produksi, lebih tepatnya Ubuntu
2	Alat	Perangkat Lunak	Visual Studio Code untuk menuliskan kode, Draw.io untuk merancang sistem dan database, Github untuk penyimpanan cloud kode, Ngixx yaitu web server pada tahap produksi, Postman untuk melakukan pengujian pada aplikasi.
3	Source Code	Perangkat Lunak	PHP sebagai bahasa yang akan jalan pada sisi server, dan Javascript sebagai bahasa yang akan jalan pada sisi klien
4	Framework	Perangkat Lunak	Laravel sebagai kerangka backend yang akan diimplementasikan kepada perangkat IoT, dan Vue.js sebagai kerangka frontend yang akan ditampilkan pada web browser.
5	Database	Perangkat Lunak	MySQL untuk bisa menjalankan transaksi, agar tidak terjadinya tabrakan data.
6	MQTT Broker	Perangkat Lunak	Mosquitto untuk menghubungkan antara perangkat IoT dengan website karena dapat mengirimkan pesan MQTT melalui Websocket.
7	Processor	Perangkat Keras	Processor dengan vCore 4 sudah cukup untuk menjalankan seluruh perangkat lunak yang akan digunakan.
8	RAM	Perangkat Keras	RAM sebesar 16 GB sudah cukup untuk menyimpan data sementara.
9	Storage	Perangkat Keras	SSD sebesar 200 GB sudah cukup untuk menyimpan aplikasi dan data pada aplikasi.
10	Microcontroller	Perangkat Keras	ESP32 untuk tahap pengembangan.

3.4 Tahap Penelitian

Proses pengembangan perangkat lunak dalam penelitian ini dirancang secara sistematis dengan mengadopsi metodologi *Software Development Life Cycle* (SDLC) model *Waterfall*. SDLC merupakan sebuah proses yang menjabarkan metode dan strategi untuk mengembangkan, merancang, dan memelihara perangkat lunak dengan tujuan utama memastikan semua sasaran serta kebutuhan fungsional pengguna terpenuhi, sehingga menghasilkan perangkat lunak berkualitas tinggi yang selesai tepat waktu sesuai estimasi biaya (Arora & Arora, 2016). Model *Waterfall* ini dipilih karena pendekatannya yang linear dan sekuensial sangat sesuai untuk proyek dengan lingkup dan kebutuhan fungsional yang telah terdefinisi secara jelas sejak awal. Keunggulan utama dari metode *Waterfall* adalah penekanannya pada dokumentasi yang detail dan penyelesaian setiap fase secara tuntas sebelum melanjutkan ke fase berikutnya, sehingga memungkinkan adanya kontrol kualitas yang terkelola dengan baik pada setiap tahapan (Ardhana et al., 2023; D. B. Pratama & Armin, 2024; Pricillia & Zulfachmi, 2021). Pendekatan yang terstruktur ini menjadi pilihan strategis, mengingat sistem yang dikembangkan memerlukan alokasi sumber daya dan waktu yang signifikan sehingga menuntut perencanaan awal yang sangat matang. Selain itu, karena pengembangan dilakukan secara mandiri, alur kerja Waterfall yang jelas dan terdefinisi memberikan kerangka kerja yang lebih terkontrol dan mudah untuk dikelola.

Gambar 3 berikut mengilustrasikan alur kerja dari model SDLC *Waterfall* yang diterapkan dalam penelitian ini. Diagram ini menunjukkan aliran proses yang bergerak secara linear dari atas ke bawah,

dimulai dari Tahap Analisis, dilanjutkan dengan Desain, Pengembangan, Pengujian, hingga berakhir pada Tahap Penyebaran. Setiap tahapan menghasilkan *deliverable* atau output spesifik yang menjadi input wajib bagi tahapan selanjutnya, memastikan proses pengembangan berjalan secara terstruktur dan sistematis.



Gambar 3 Tahapan Penelitian dengan SDLC Model Waterfall

1. Tahap Analisis: Pada tahap analisis, dilakukan identifikasi kebutuhan sistem secara mendalam dari sisi pengguna maupun teknis, yang mencakup kebutuhan fungsional dan non-fungsional (Haniva et al., 2023). Secara fungsional, sistem didefinisikan harus mampu menampilkan data sensor dari perangkat IoT secara *real-time* serta mengirimkan perintah kendali dari antarmuka pengguna ke perangkat di lapangan. Analisis ini juga mencakup pemahaman terhadap alur kerja operasional yang ada, lingkungan teknis, serta batasan-batasan yang mungkin dihadapi. Hasil dari tahap ini berupa dokumen spesifikasi kebutuhan yang menjadi landasan fundamental dalam penyusunan desain sistem pada tahap berikutnya.
2. Tahap Design: Tahap desain merupakan kelanjutan dari proses analisis, di mana rancangan sistem yang detail dan komprehensif dibuat berdasarkan kebutuhan yang telah diidentifikasi (Deni Murdiani & Muhamad Sobirin, 2022). Pada tahap ini, dirancang beberapa komponen krusial, antara lain desain arsitektur aplikasi secara keseluruhan serta desain struktur basis data menggunakan MySQL yang divisualisasikan dengan *Entity Relationship Diagram* (ERD). Perancangan yang matang pada tahap ini memastikan bahwa pengembangan sistem dilakukan dengan landasan teknis yang kuat, terstruktur, dan selaras dengan tujuan penelitian.
3. Tahap Pengembangan: Implementasi kode program dilakukan berdasarkan rancangan sistem yang telah disusun pada tahap sebelumnya. Tahapan ini mencakup tiga alur kerja utama (Badrul et al., 2021). Pertama, pengembangan *backend* menggunakan *framework* Laravel (PHP) yang bertugas untuk mengelola seluruh logika bisnis, interaksi dengan basis data, dan menyediakan REST API. Kedua, pengembangan *frontend* menggunakan VueJS (JavaScript) sebagai antarmuka pengguna yang bertanggung jawab untuk menyajikan data dan menerima input. Ketiga, integrasi protokol MQTT sebagai jalur komunikasi ringan yang memungkinkan pengiriman data dan perintah secara *real-time* antara perangkat IoT dan server. Pengembangan pada sisi *frontend* dan *backend* dilakukan secara paralel dengan titik integrasi utama melalui REST API dan *broker* MQTT.
4. Tahap Pengujian: Pengujian dilakukan secara menyeluruh untuk memastikan setiap komponen sistem berjalan sesuai dengan fungsinya dan bebas dari kesalahan kritis. Proses pengujian dibagi menjadi tiga area utama. Pertama, pengujian *backend* menggunakan PHPUnit dan Postman untuk memvalidasi setiap *endpoint* API dan proses logika bisnis. Kedua, pengujian komunikasi MQTT dilakukan untuk mengevaluasi latensi dan keandalan transmisi data *real-time* antara perangkat dan

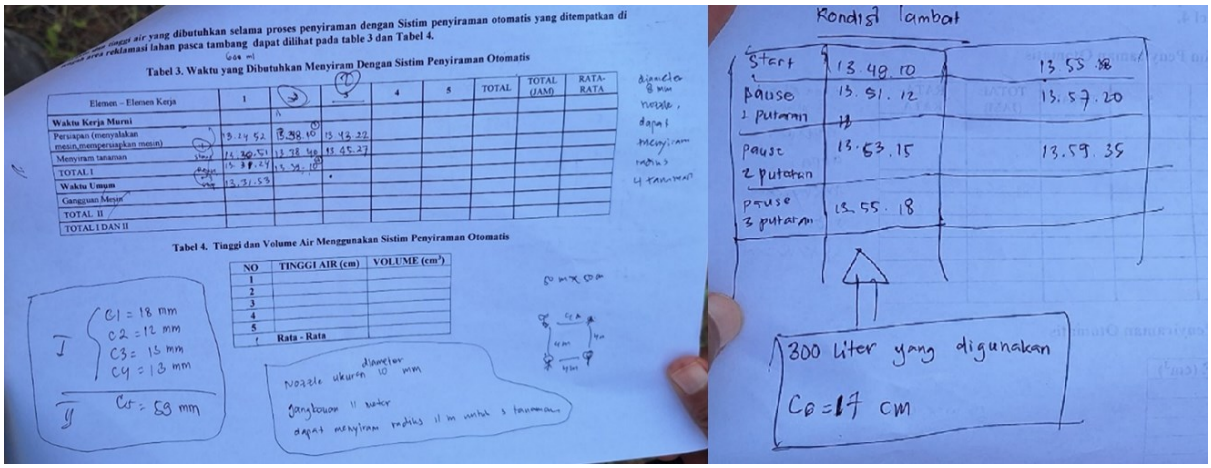
sistem. Ketiga, pengujian *frontend* menggunakan Chrome DevTools untuk mengevaluasi performa visualisasi data, konsumsi memori, dan kecepatan *rendering* antarmuka pengguna. Rangkaian pengujian ini memberikan jaminan bahwa sistem yang dihasilkan dapat berjalan secara stabil, responsif, dan sesuai dengan kebutuhan operasional pengguna.

5. Tahap Penyebaran: Tahap terakhir dari metodologi ini adalah *deployment* yaitu hasil dari tahapan-tahapan sebelumnya yang akan digunakan user (Normah et al., 2022). Aplikasi di-*deploy* pada sebuah server yang menjalankan sistem operasi Ubuntu 22.04 dan menggunakan Nginx sebagai *web server*. Seluruh konfigurasi harus dilakukan secara cermat untuk memastikan sistem berjalan dengan lancar. Penyebaran ini memungkinkan aplikasi untuk digunakan secara penuh oleh PT Insani Baraperkasa sebagai sistem pendukung dalam pengelolaan dan monitoring lahan reklamasi tambang batubara secara digital dan *real-time*.

4. Hasil dan Pembahasan

4.1 Hasil Pengmpulan Data

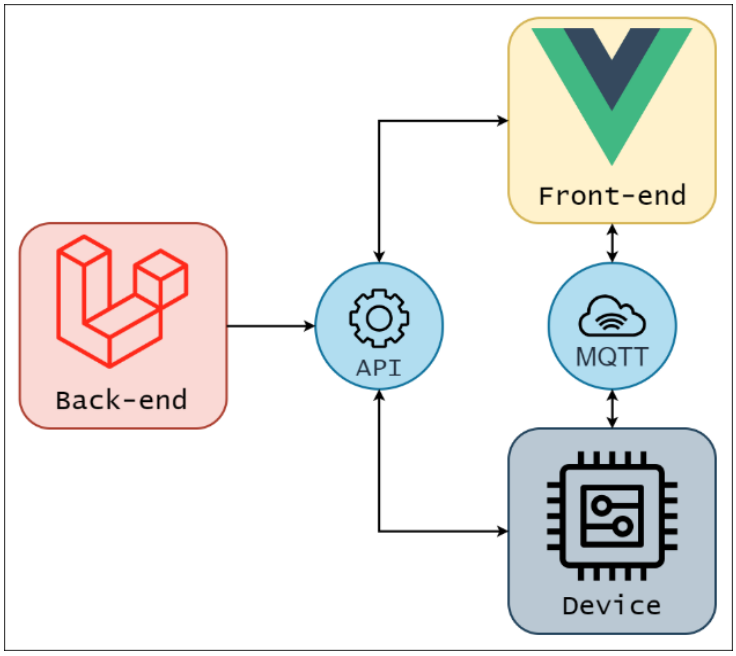
Tahap perancangan sistem diawali dengan pengumpulan data operasional melalui observasi lapangan, di mana proses pengujian empiris dan hasilnya didokumentasikan pada Gambar 4. Melalui pengujian ini, diidentifikasi parameter-parameter kinerja krusial, di antaranya adalah durasi satu putaran penuh (*rotation*) *sprinkler* yang konsisten sekitar 2 menit (tercatat dari pukul 13:49:10 hingga 13:51:12), serta jangkauan semprotan (*radius*) efektif yang mencapai 11 meter dengan sebaran air yang diukur pada beberapa titik (misalnya, 18 mm dan 12 mm). Kombinasi data primer ini, yaitu waktu rotasi 2 menit untuk cakupan radius 11 meter, menjadi acuan dasar (*baseline*) fundamental dalam perancangan algoritma kontrol presisi yang mampu menghitung durasi penyiraman optimal demi mencapai tingkat kebasahan yang merata di seluruh area jangkauannya.



Gambar 4 Hasil Pengujian Alat secara Manual

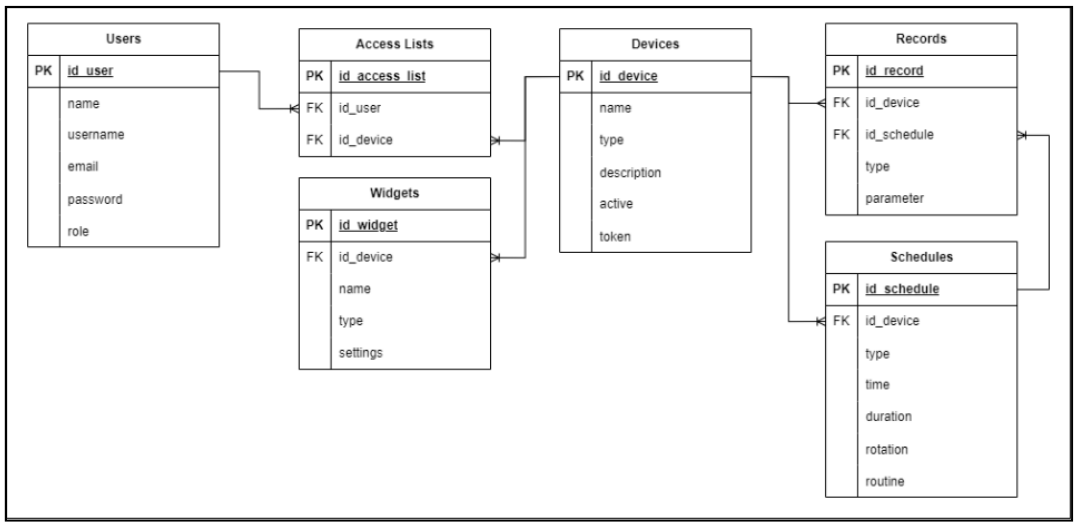
4.2 Hasil Analisis dan Desain Sistem

Berdasarkan analisis kebutuhan, sistem dirancang menggunakan arsitektur *headless* yang memisahkan secara tegas antara lapisan *backend* dan *frontend* untuk mencapai fleksibilitas pengembangan, skalabilitas, dan reusabilitas. Sebagaimana divisualisasikan pada Gambar 5, arsitektur ini terdiri dari lima komponen utama. *Backend*, yang dibangun menggunakan *framework* Laravel, berfungsi sebagai otak aplikasi yang menangani seluruh logika bisnis dan manajemen data. *Frontend*, yang dikembangkan dengan Vue.js, bertanggung jawab penuh menyajikan antarmuka pengguna (UI) yang interaktif. Komunikasi antara keduanya dijembatani oleh *API* (*Application Programming Interface*), sementara untuk komunikasi data *real-time* dengan Perangkat *Internet of Things* (*Device*), sistem memanfaatkan protokol *MQTT* (*Message Queuing Telemetry Transport*) yang ringan dan efisien. Alur kerja sistem dimulai dari interaksi pengguna di *frontend* yang mengirimkan permintaan ke API, yang kemudian diproses oleh *backend*. Secara paralel, *Device* terus-menerus mengirimkan data telemetri ke *broker* MQTT, yang kemudian diterima dan ditampilkan secara *real-time* oleh *frontend*, memastikan penyajian data yang sinkron dan responsif.



Gambar 5 Desain Arsitektur Aplikasi

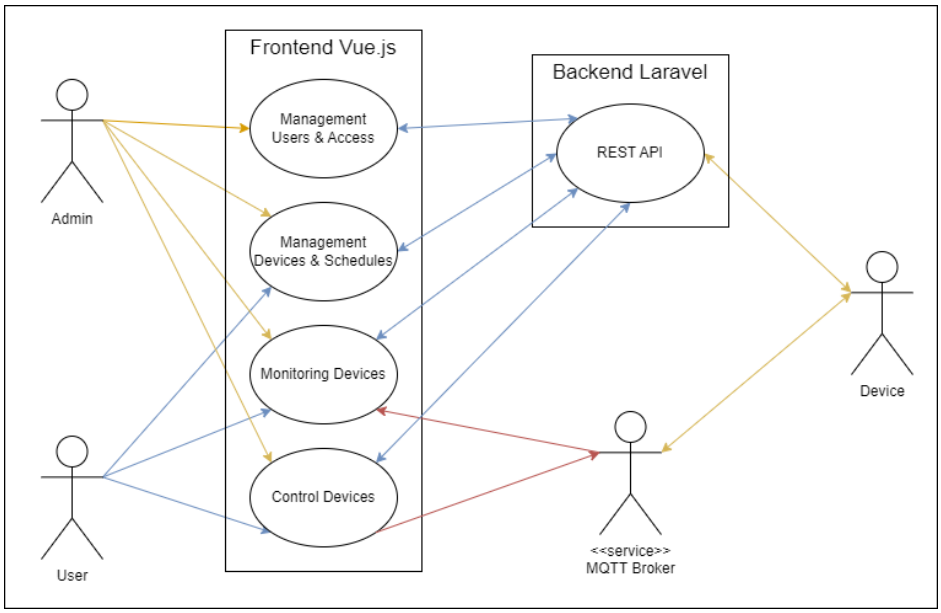
Struktur basis data, yang divisualisasikan dalam *Entity Relationship Diagram* (ERD) pada Gambar 6, dirancang secara spesifik untuk mengelola entitas, relasi, dan data operasional sistem, dengan entitas inti *Users* sebagai pusat otentikasi dan *Devices* yang merepresentasikan setiap perangkat fisik. Untuk mengatur hak akses secara fleksibel, diimplementasikan relasi *many-to-many* antara *Users* dan *Devices* melalui tabel perantara *Access Lists*, yang memungkinkan seorang pengguna memiliki akses ke banyak perangkat dan sebaliknya. Selanjutnya, entitas *Devices* menjadi pusat dari beberapa relasi *one-to-many* yang krusial: dengan *Widgets* untuk kustomisasi antarmuka, *Schedules* untuk penjadwalan tugas otomatis, dan *Records* untuk pencatatan data historis. Entitas *Records* sendiri dirancang untuk mencatat setiap aktivitas atau data telemetri dengan relasi tidak hanya ke *Devices* tetapi juga ke *Schedules*, sehingga memungkinkan penelusuran data berdasarkan pemicu jadwal tertentu. Secara keseluruhan, desain basis data ini menyediakan fondasi yang normal dan terstruktur, memastikan integritas data serta mendukung fitur-fitur utama aplikasi seperti kontrol akses multi-pengguna, penjadwalan, dan pencatatan riwayat perangkat secara efisien.



Gambar 6 *Entity Relationship Diagram* Aplikasi

Gambar 7 menyajikan *Use Case Diagram* yang memodelkan fungsionalitas sistem secara keseluruhan, dengan mendefinisikan interaksi tiga aktor utama: *Admin*, *User*, dan *Device*. Admin memiliki

kewenangan penuh atas fungsi manajemen sistem seperti pengelolaan pengguna dan perangkat (*Management Users & Access*, *Management Devices & Schedules*), yang diakses melalui REST API pada sisi *backend*. Sementara itu, User memiliki akses terbatas pada fungsi *Monitoring Devices dan Control Devices* sesuai dengan hak aksesnya. Arsitektur sistem ini mengadopsi pendekatan komunikasi *dual-pathway*, di mana *REST API* menangani operasi *stateful* berbasis *request-response* seperti pengambilan data awal untuk monitoring, sedangkan *MQTT Broker*, yang dimodelkan sebagai aktor <<service>>, bertanggung jawab atas komunikasi *real-time* dan asinkron untuk pengiriman perintah kontrol dan penerimaan pembaruan data. Aktor *Device*, sebagai perangkat fisik, berinteraksi dengan kedua jalur tersebut dengan mengirimkan data telemetri secara *real-time* ke *MQTT Broker* serta berkomunikasi dengan *REST API* untuk sinkronisasi konfigurasi. Pemisahan alur ini memastikan bahwa operasi yang menuntut integritas data ditangani oleh *REST API*, sementara interaksi yang sensitif terhadap latensi dikelola secara efisien oleh *MQTT*.

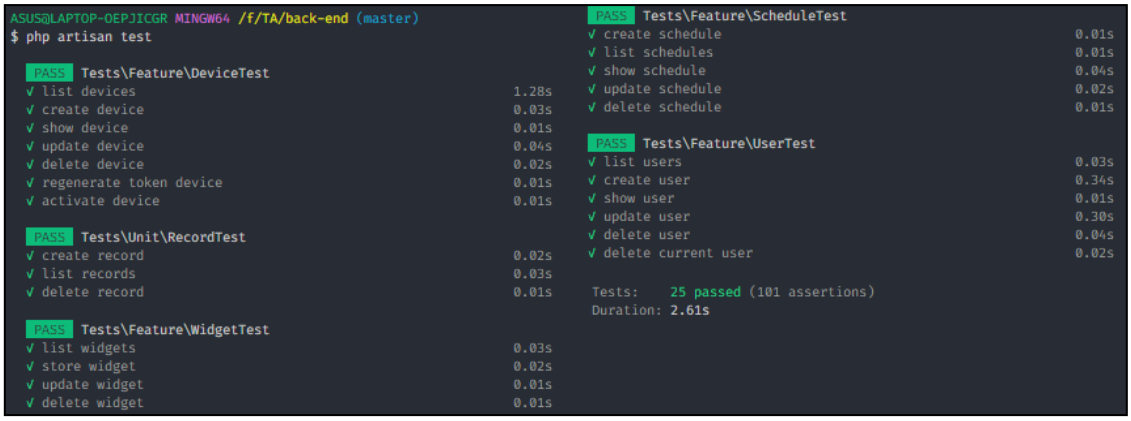


Gambar 7 *Use Case Diagram* Aplikasi

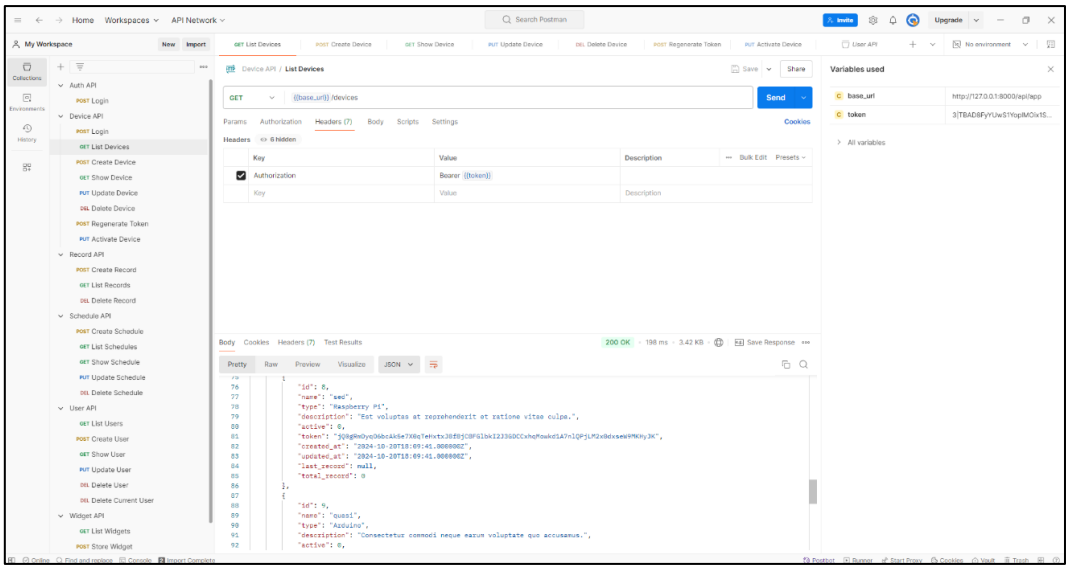
4.3 Hasil Pengujian Sistem

4.3.1 Hasil Pengujian Backend

Pengujian *backend* dilakukan secara komprehensif dengan menerapkan dua pendekatan yang saling melengkapi: *white-box testing* untuk validasi internal dan *black-box testing* untuk verifikasi eksternal. Sebagaimana ditunjukkan pada Gambar 8, pengujian *white-box* dieksekusi secara otomatis menggunakan *framework PHPUnit* melalui antarmuka baris perintah (*CLI*). Pengujian ini mencakup *unit testing*, yang bertujuan untuk mengisolasi dan memverifikasi kebenaran fungsional dari setiap metode atau kelas secara individual, serta *feature testing* yang menguji alur kerja lengkap dari sebuah fitur, seperti operasi *CRUD* (*Create, Read, Update, Delete*), untuk memastikan semua komponen internal berinteraksi dengan benar. Sementara itu, pengujian *black-box*, yang divisualisasikan pada Gambar 9, dilakukan menggunakan *Postman* untuk mensimulasikan interaksi nyata dari aplikasi klien. Pada tahap ini, setiap *endpoint API* diuji secara manual dengan mengirimkan permintaan *HTTP* spesifik untuk memastikan beberapa aspek krusial: validitas *header* otorisasi, ketepatan *status code* respons (*HTTP status code*) seperti 200 OK, serta keakuratan struktur dan isi data dari *response payload* dalam format *JSON*. Kombinasi kedua metode pengujian ini memastikan bahwa *backend* tidak hanya solid secara logika internal, tetapi juga andal dan sesuai dengan kontrak *API* ketika diakses dari luar.



Gambar 8 Pengujian *Backend* dengan *Unit* dan *Feature Test*



Gambar 9 Pengujian *Backend* dengan *Postman*

Untuk memvalidasi fungsionalitas dan mengukur kinerja *backend*, serangkaian pengujian komprehensif dijalankan pada setiap *endpoint* API esensial yang mencakup operasi *CRUD*. Data kinerja dari setiap *test case*, terutama waktu eksekusi, dicatat untuk skenario pengujian internal (*PHPUnit*) dan eksternal (*Postman*). Hasil dari pengujian tersebut, yang disajikan secara komparatif pada Tabel 2, menunjukkan bahwa pendekatan *white-box* dengan *PHPUnit* sangat efisien untuk validasi selama pengembangan dengan total waktu 2,61 detik. Di sisi lain, pengujian *black-box* dengan *Postman* memberikan gambaran performa yang lebih realistis dengan total waktu 4,242 detik, karena telah mencakup latensi jaringan dan *request-response cycle* secara penuh. Perbedaan waktu yang signifikan pada beberapa *test case*, seperti "List Devices" (1280 ms pada *PHPUnit* vs. 198 ms pada *Postman*), menyoroti bagaimana *PHPUnit* mungkin melibatkan *setup* data yang intensif, berbeda dengan *Postman* yang mengukur latensi permintaan tunggal. Secara keseluruhan, analisis kuantitatif ini mengonfirmasi bahwa *backend* tidak hanya valid secara fungsional, tetapi juga memiliki performa yang andal untuk penggunaan di dunia nyata.

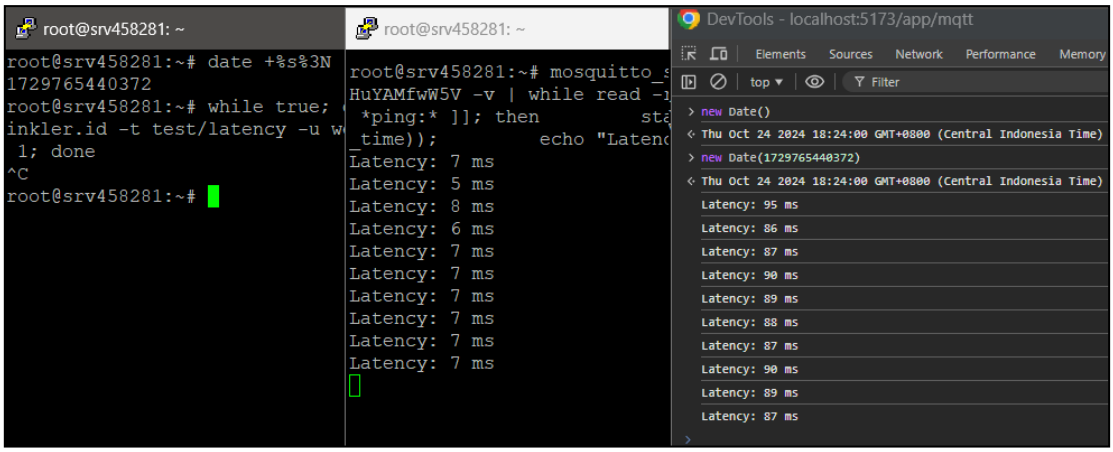
Tabel 2. Hasil Pengujian Backend

No	Nama Pengujian	Deskripsi	PHPUnit	Postman
1	List Devices	Menguji daftar perangkat yang tersedia.	1280 ms	198 ms
2	Create Device	Menguji pembuatan perangkat baru.	30 ms	193 ms
3	Show Device	Menguji tampilan detail perangkattertentu.	10 ms	161 ms
4	Update Device	Menguji pembaruan informasi perangkat.	40 ms	174 ms
5	Delete Device	Menguji penghapusan perangkat.	20 ms	162 ms

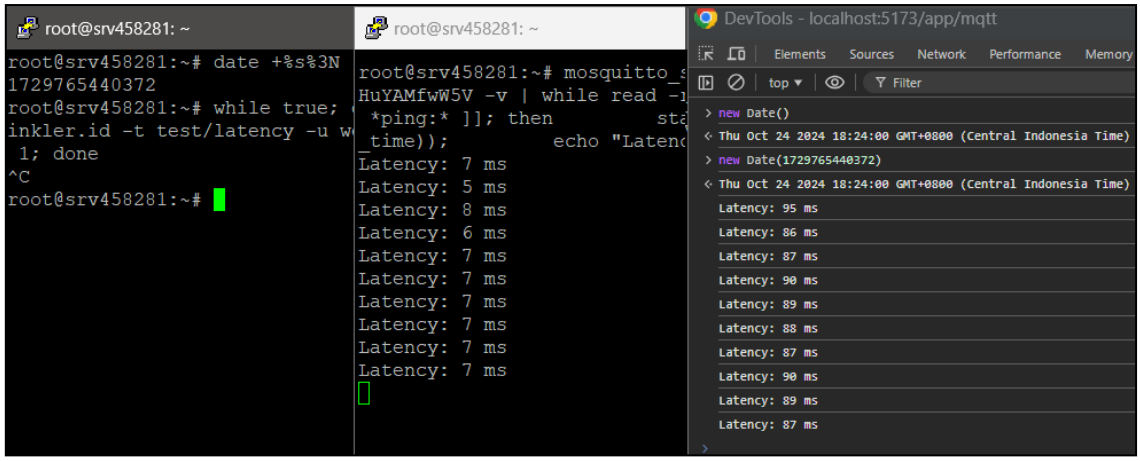
6	Regenerate TokenDevice	Menguji regenerasi token perangkat.	10 ms	164 ms
7	Active Device	Menguji aktivasi perangkat.	10 ms	171 ms
8	Create Record	Menguji pembuatan record baru.	20 ms	184 ms
9	List Records	Menguji daftar record yang ada.	30 ms	163 ms
10	Delete Record	Menguji penghapusan record.	10 ms	164 ms
11	Create Schedule	Menguji pembuatan jadwal baru.	10 ms	156 ms
12	List Schedules	Menguji daftar jadwal yang tersedia.	20 ms	114 ms
13	Show Schedule	Menguji tampilan detail jadwal tertentu.	40 ms	191 ms
14	Update Schedule	Menguji pembaruan informasi jadwal.	20 ms	189 ms
15	Delete Schedule	Menguji penghapusan jadwal.	20 ms	166 ms
16	List Users	Menguji daftar pengguna yang tersedia.	30 ms	152 ms
17	Create User	Menguji pembuatan pengguna baru.	40 ms	358 ms
18	Show User	Menguji tampilan detail pengguna tertentu.	40 ms	119 ms
19	Update User	Menguji pembaruan informasi pengguna.	400 ms	342 ms
20	Delete User	Menguji penghapusan pengguna.	300 ms	114 ms
21	Delete CurrentWidget	Menguji penghapusan akun pengguna saat ini.	30 ms	106 ms
22	List Widgets	Menguji daftar widget yang tersedia.	30 ms	156 ms
23	Store Widget	Menguji pembuatan widget baru.	20 ms	122 ms
24	Update Widget	Menguji pembaruan informasi widget.	20 ms	115 ms
25	Delete Widget	Menguji penghapusan widget.	10 ms	108 ms

4.3.2 Hasil Pengujian MQTT

Pengujian protokol *MQTT* dilakukan secara spesifik untuk mengukur latensi komunikasi *real-time* antara *backend* dan *frontend* melalui *MQTT broker*. Pengujian ini melibatkan dua skenario utama yang divisualisasikan pada Gambar 10 dan Gambar 11. Skenario pertama (Gambar 10) menguji latensi pengiriman pesan dari klien *MQTT native* (menggunakan perintah mosquitto_pub pada *command-line interface*) ke klien *frontend* yang terhubung melalui *WebSocket over MQTT*. Skenario kedua (Gambar 11) menguji arah sebaliknya, yaitu latensi pengiriman pesan dari klien *frontend* (menggunakan *library MQTT.js* dalam lingkungan *Node.js*) ke *subscriber MQTT native* (menggunakan perintah mosquitto_sub pada *CLI*). Kedua gambar menampilkan log konsol dari sisi *publisher* dan *subscriber*. Pada sisi *publisher*, terlihat stempel waktu (*timestamp*) saat pesan "ping" dikirimkan. Pada sisi *subscriber* (*DevTools* pada *browser frontend*), terlihat stempel waktu saat pesan diterima dan perhitungan latensi (*latency*) yang merupakan selisih antara waktu penerimaan dan waktu pengiriman. Pengujian ini dilakukan berulang kali untuk mendapatkan sampel latensi yang beragam, sehingga dapat dievaluasi kinerja komunikasi *real-time* sistem dalam berbagai kondisi.



Gambar 10 Hasil Pengujian *Request* dari *MQTT* ke *Websocket* dan *MQTT*



Gambar 11 Hasil Pengujian *Request* dari *Websocket* ke *Webscoket* dan *MQTT*

Hasil pengujian kuantitatif latensi komunikasi *real-time*, yang dirangkum pada Tabel 3, menyajikan analisis perbandingan antara dua skenario utama. Skenario pertama, "Request dari *MQTT*", menyimulasikan pengiriman data dari perangkat (*device*) ke *frontend*. Data menunjukkan bahwa latensi dari *publisher* ke *subscriber MQTT native* sangat rendah, dengan rata-rata 7 ms, yang merefleksikan kecepatan murni komunikasi *machine-to-machine* melalui *MQTT broker*. Namun, ketika pesan yang sama diterima oleh *frontend* melalui *WebSocket*, latensi meningkat signifikan menjadi rata-rata 89 ms. Peningkatan ini disebabkan oleh *overhead* yang diperlukan untuk menerjemahkan protokol *MQTT* ke dalam *frame WebSocket* agar dapat diterima oleh *browser*. Skenario kedua, "*Request* dari *Websocket*", menguji arah sebaliknya, di mana *frontend* mengirim perintah ke perangkat. Latensi satu arah dari *frontend* ke *subscriber MQTT native* tercatat sekitar 114 ms, menunjukkan adanya waktu tambahan untuk pemrosesan di sisi klien (*browser*) sebelum pesan dikirim. Data juga menunjukkan adanya beberapa lonjakan latensi (*latency spike*), seperti nilai 646 ms, yang merupakan fenomena wajar dalam komunikasi berbasis *web* akibat faktor-faktor seperti *network jitter* atau proses internal *browser*. Secara keseluruhan, pengujian ini mengonfirmasi bahwa meskipun terdapat *overhead* latensi yang terukur, penggunaan *WebSocket* sebagai jembatan merupakan solusi yang efektif dan esensial untuk mengintegrasikan kapabilitas *real-time* dari *MQTT* ke antarmuka pengguna berbasis *web* dengan performa yang masih sangat responsif untuk kebutuhan visualisasi data.

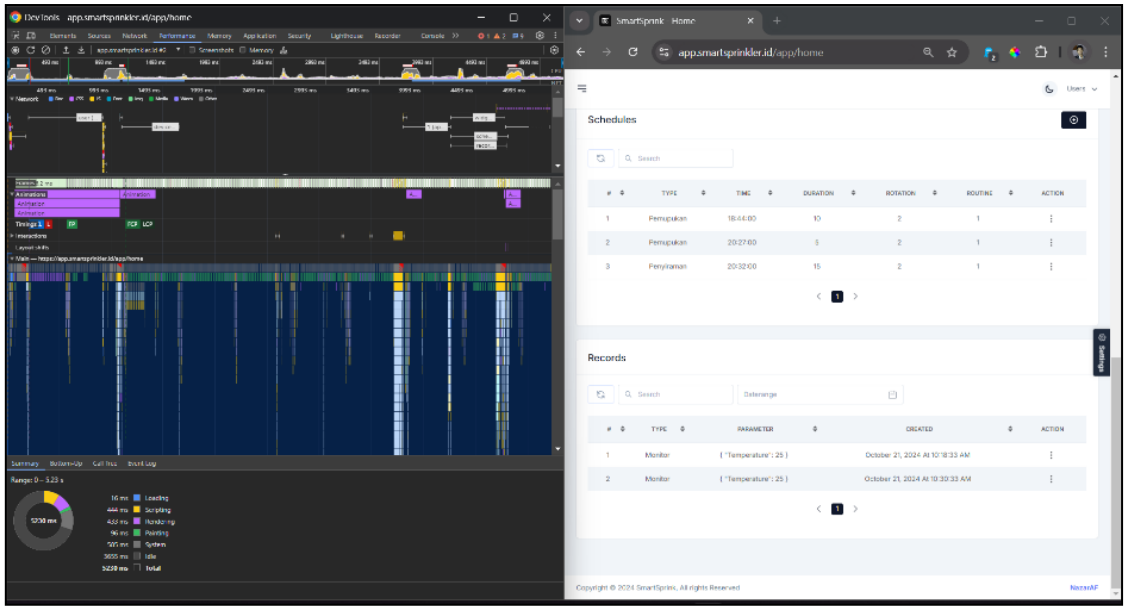
Tabel 3. Hasil Pengujian *MQTT*

No	***Request* dari *MQTT***		***Request* dari *Websocket***	
	Websocket	***MQTT***	***Websocket***	***MQTT***
1	95 ms	7 ms	193 ms	116 ms
2	86 ms	5 ms	192 ms	114 ms
3	87 ms	8 ms	191 ms	115 ms
4	90 ms	6 ms	191 ms	114 ms
5	89 ms	7 ms	192 ms	113 ms
6	88 ms	7 ms	646 ms	113 ms
7	87 ms	7 ms	193 ms	114 ms
8	90 ms	7 ms	191 ms	113 ms
9	89 ms	7 ms	191 ms	113 ms
10	87 ms	7 ms	194 ms	113 ms

4.3.3 Hasil Pengujian Frontend

Pengujian *frontend* difokuskan pada analisis kinerja sisi klien (*client-side performance*) untuk memastikan pengalaman pengguna yang responsif. Menggunakan panel *Performance* pada *DevTools Google Chrome*, sebagaimana ditunjukkan pada Gambar 12, perekaman dan analisis mendalam dilakukan terhadap seluruh aktivitas *browser* saat memuat dan merender halaman utama aplikasi. Hasil analisis yang ditampilkan pada diagram *flame chart* dan ringkasan (*summary*) memberikan data

kuantitatif yang detail mengenai alokasi sumber daya. Dari total waktu perekaman selama 3.12 detik, skrip (*scripting*) hanya memakan waktu 432 ms, proses *rendering* 16 ms, dan *painting* (melukis piksel ke layar) 6.4 ms. Waktu yang dominan digunakan untuk *idle* (1230 ms) dan *system* (1420 ms), yang menunjukkan bahwa beban kerja utama aplikasi pada *CPU* sangat efisien. Data ini mengonfirmasi bahwa arsitektur *frontend* yang digunakan mampu merender antarmuka dengan cepat, tidak menyebabkan *bottleneck* pada *main thread browser*, dan secara keseluruhan memberikan performa yang ringan dan cepat saat diakses oleh pengguna.



Gambar 12 Pengujian *Frontend* dengan *DevTools*

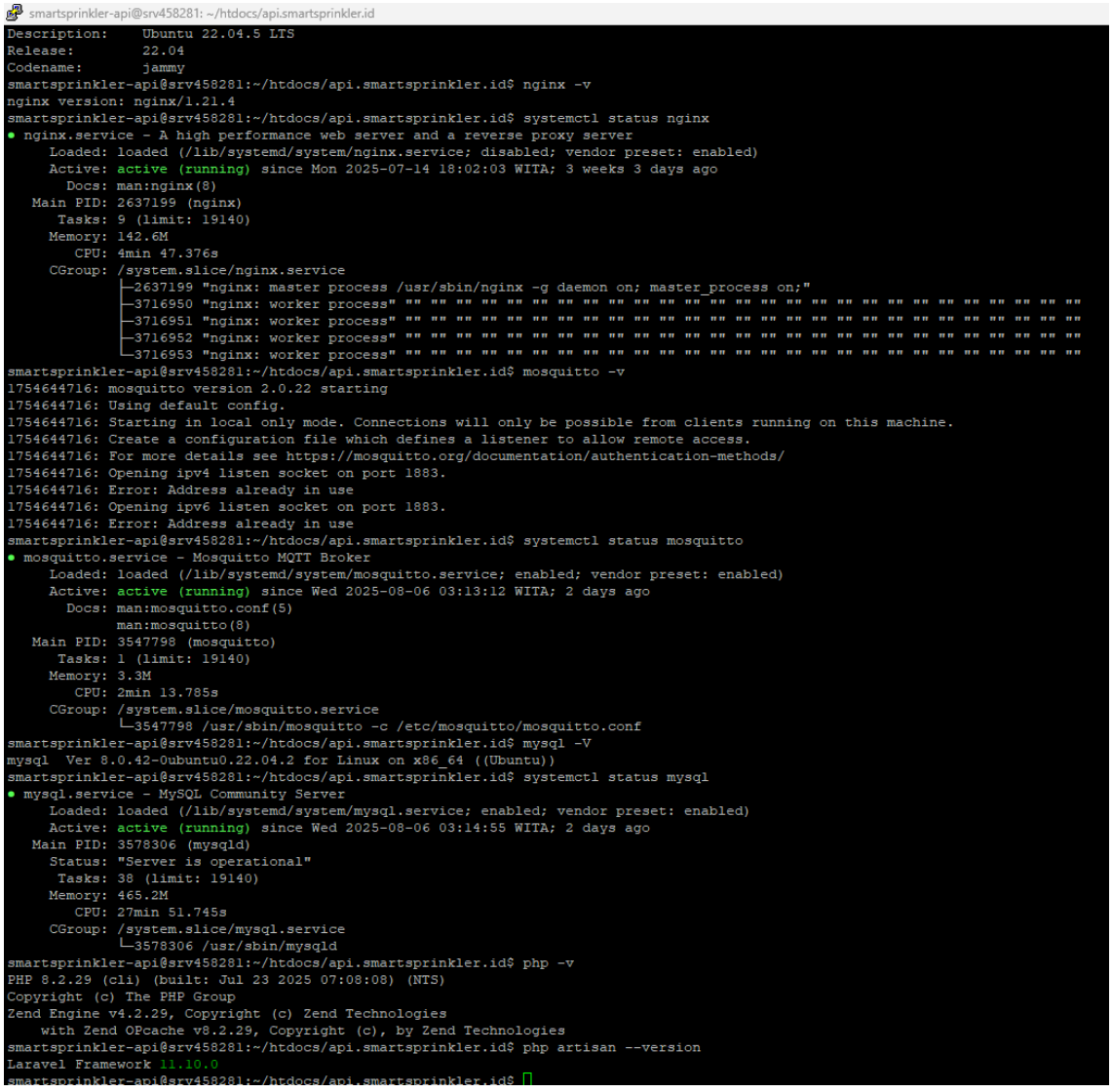
Tabel 4 menyajikan hasil pengukuran kuantitatif dari pengujian performa pada enam halaman utama aplikasi *frontend*. Data menunjukkan bahwa waktu *render* untuk sebagian besar halaman berada dalam rentang 254-425 ms, yang mengindikasikan kinerja pemuatan yang sangat baik. Namun, halaman "Detail *Device*" mencatatkan waktu *render* terlama (562 ms) dan penggunaan memori (*memory footprint*) tertinggi secara signifikan, yaitu 35.901 kB (sekitar 35.9 MB). Lonjakan penggunaan sumber daya ini dapat diatribusikan pada kompleksitas halaman tersebut, yang tidak hanya memuat komponen antarmuka yang lebih banyak, tetapi juga melakukan visualisasi data sensor historis dalam bentuk grafik dan menangani aliran data *real-time* melalui koneksi *WebSocket*. Sementara halaman lain seperti "*Login*" sangat ringan (8.3 MB), tingginya kebutuhan memori pada halaman detail perangkat memberikan implikasi praktis bahwa untuk memastikan interaksi yang lancar dan tanpa jeda, terutama saat memantau data dalam jangka waktu lama, perangkat pengguna disarankan memiliki kapasitas memori yang memadai.

Tabel 4. Hasil Pengujian *Frontend*

No	Halaman	Waktu Render	Penggunaan Memori
1	*Login*	43 ms	8319 KB
2	*Dashboard*	334 ms	13777 KB
3	*Users*	254 ms	13966 KB
4	*Devices*	404 ms	11047 KB
5	*Detail Device*	562 ms	359001 KB
6	*MQTT*	425 ms	10342 KB

4.4 Hasil Penyebaran

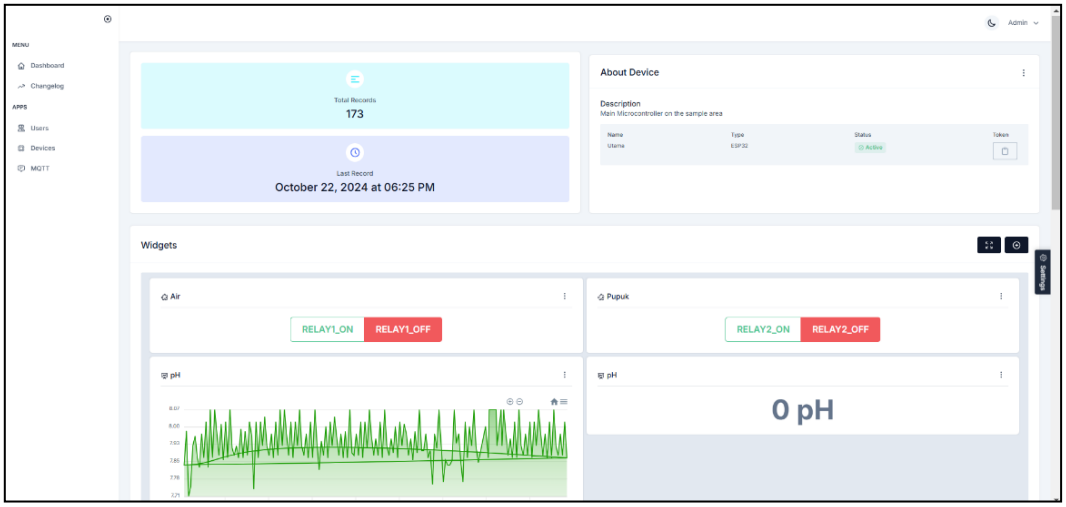
Setelah melalui seluruh tahapan pengembangan dan pengujian, aplikasi berhasil diimplementasikan (*deploy*) pada *cloud server*. Gambar 13 menampilkan verifikasi tumpukan perangkat lunak (*software stack*) yang berjalan di lingkungan produksi. Sistem operasi yang digunakan adalah Ubuntu 22.04.5 LTS, yang menjadi fondasi stabil untuk seluruh layanan. Nginx 1.21.4 berfungsi sebagai *web server* dan *reverse proxy* berkinerja tinggi. Untuk komunikasi *real-time*, Mosquitto 2.0.22 diimplementasikan sebagai *MQTT broker*. Persistensi data ditangani oleh *MySQL Server* 8.0.42, sementara inti dari aplikasi *backend* berjalan di atas *PHP* 8.2.29 dengan *framework* Laravel 11.13.0. Kombinasi perangkat lunak yang terverifikasi aktif dan berjalan ini menciptakan lingkungan server produksi yang solid, aman, dan mampu menangani komunikasi data konvensional melalui *API* serta aliran data *real-time* dari perangkat IoT secara efisien.



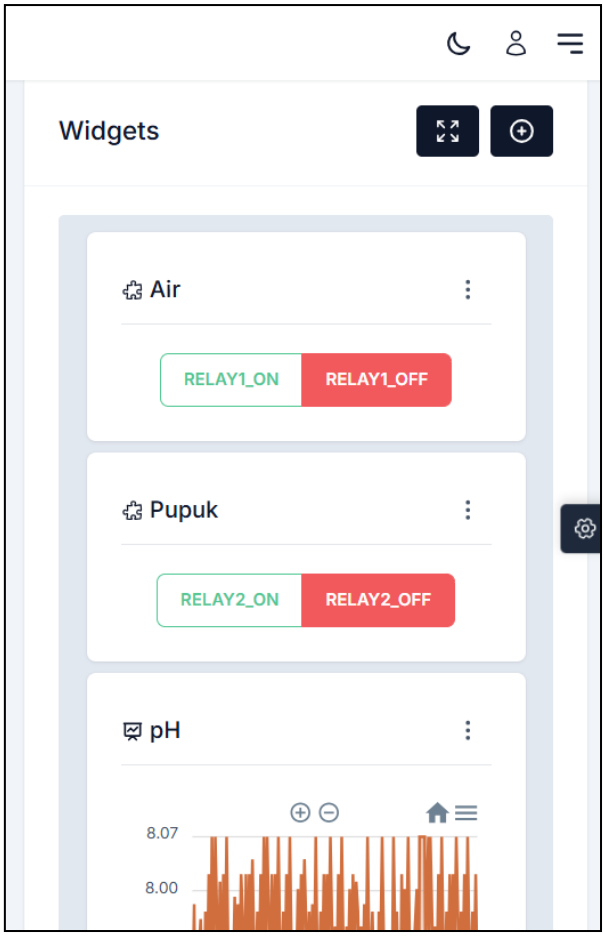
Gambar 13 *Environtment* pada *Cloud*

Hasil akhir dari penelitian ini adalah sebuah aplikasi *web* fungsional dengan desain responsif yang siap mendukung operasional reklamasi lahan. Gambar 14 dan Gambar 15 menampilkan antarmuka halaman *Detail Device*, yang merupakan fitur utama aplikasi, disajikan dalam tampilan *desktop* (Gambar 14) dan *mobile* (Gambar 15). Halaman ini berfungsi sebagai pusat kontrol dan pemantauan untuk satu perangkat spesifik, yang menyediakan ringkasan data operasional seperti jumlah total *record* dan waktu data terakhir diterima. Selain itu, halaman ini juga menyajikan analisis data historis melalui grafik interaktif serta fitur kontrol aktuator secara langsung melalui berbagai *widget* kustom. Implementasi desain responsif terlihat jelas dari perbedaan tata letak: tampilan *desktop* menggunakan format multi-

kolom untuk menyajikan informasi secara komprehensif, sementara tampilan *mobile* menyusun ulang semua elemen dalam satu kolom vertikal untuk memastikan aksesibilitas dan keterbacaan pada layar yang lebih kecil. Pendekatan ini menjamin pengalaman pengguna yang optimal dan konsisten di berbagai perangkat, baik di ruang kontrol maupun saat di lapangan.



Gambar 14 *Detail Device* pada *Desktop*

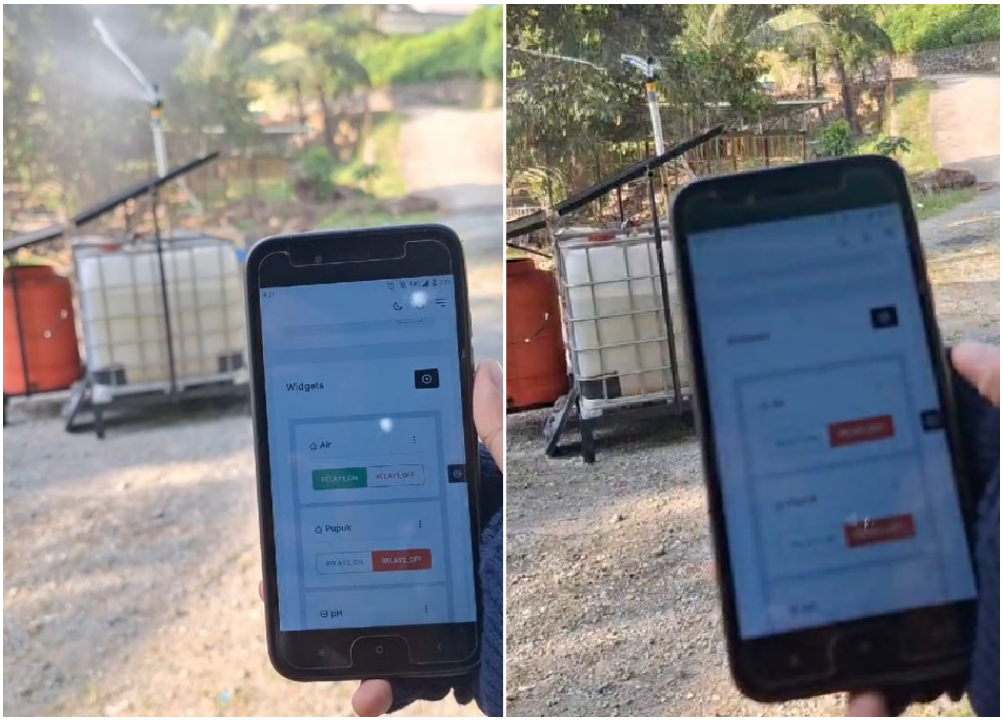


Gambar 15 Tampilan *Detail Device* pada *Mobile*

Kemampuan operasional sistem di lapangan menjadi bukti nyata dari integrasi fungsionalitas aplikasi yang telah dirancang. Gambar 16 mengilustrasikan penerapan praktis ini, di mana *monitoring* kondisi lahan dapat dilakukan secara *live* dan *on-site* melalui antarmuka *mobile*. Seluruh data dan perintah

kontrol yang diakses melalui aplikasi terhubung secara *real-time* ke unit irigasi dan *monitoring* fisik, yang diperlihatkan pada Gambar 17. Perangkat ini dirancang untuk beroperasi secara mandiri, ditenagai oleh panel surya serta dilengkapi tandon dan *sprinkler*. Dengan demikian, integrasi antara antarmuka digital dan perangkat fisik ini secara efektif menerjemahkan perintah virtual menjadi aksi nyata di lapangan untuk mendukung manajemen lahan yang responsif.

Gambar 16 Tampilan *Detail Device* pada *Mobile* saat Perangkat Aktif



Gambar 17 Tampilan *Detail Device* dan Perangkat ketika diaktifkan

5. Kesimpulan

Berdasarkan hasil pengembangan dan pengujian menyeluruh, penelitian ini telah berhasil mencapai tujuannya dengan mengembangkan aplikasi *web* sistem irigasi dan monitoring yang fungsional untuk mendukung operasional reklamasi lahan bekas tambang. Aplikasi yang dibangun terbukti mampu menyediakan tiga fitur utama yaitu pemantauan kondisi lahan secara *real-time*, penjadwalan penyiraman otomatis, dan pengendalian perangkat *IoT* terpusat. Kestabilan *backend* terkonfirmasi melalui 25 *test case* yang berhasil dieksekusi menggunakan *PHPUnit* dalam 2,61 detik dan *Postman* dalam 4,242 detik, yang memvalidasi logika internal sekaligus kinerja pada skenario nyata. Fitur komunikasi *real-time* terbukti andal, di mana pengujian *MQTT* menunjukkan latensi sangat rendah pada komunikasi *native* (rata-rata 7 ms) dan tetap responsif saat dijembatani ke *frontend* melalui *WebSocket* (rata-rata 89 ms). Pada sisi *frontend*, kinerja aplikasi secara umum efisien, meskipun teridentifikasi adanya kebutuhan optimasi pada halaman "Detail Device" yang mencatatkan waktu *render* terlama (562 ms) dan penggunaan memori tertinggi (sekitar 35.9 MB). Dengan demikian, aplikasi ini secara fungsional layak dan andal untuk diimplementasikan lebih lanjut. Rencana penelitian lanjutan dapat difokuskan pada optimasi penggunaan memori di sisi *frontend*, integrasi algoritma *machine learning*, serta pengembangan aplikasi *mobile native* untuk meningkatkan fleksibilitas operasional.

Limitasi dan Studi Lanjutan

Penelitian ini memiliki beberapa keterbatasan, seperti fokus pada platform web tanpa aplikasi *mobile native*, penggunaan protokol yang terbatas pada HTTP dan MQTT, serta adanya temuan isu kinerja pada *frontend* yang membutuhkan perbaikan. Oleh karena itu, studi lanjutan direkomendasikan untuk fokus pada optimalisasi kinerja *frontend*, pengembangan aplikasi *mobile native* untuk fleksibilitas, serta integrasi algoritma *machine learning* untuk menciptakan sistem penjadwalan irigasi yang prediktif dan lebih efisien.

Ucapan Terima Kasih

Penulis berterima kasih kepada Politeknik Negeri Samarinda atas segala dukungan dan fasilitas. Ucapan terima kasih juga disampaikan kepada dosen pembimbing atas bimbingan dan arahannya. Terima kasih kepada PT. Insani Baraperkasa atas izin dan kerja sama selama penelitian, serta seluruh rekan Program Studi Teknologi Rekayasa Komputer atas diskusi dan dukungannya.

Referensi

Amalia, A., Wanda, S., Hamidah, P., & Kristanto, T. (2021). *Pengujian Black Box Menggunakan Teknik Equivalence Partitions Pada Aplikasi E-Learning Berbasis Web*. *3*(3), 269–274. https://doi.org/10.47065/bits.v3i3.1062

Ardhana, V. Y. P., Hidayat, M. T., Jannah, M., Sumiati, S., Rini, P., & Sari, N. (2023). Implementasi RESTful API Pada Laravel dan Simulator IoT Wokwi Untuk Pengukuran Suhu dan Kelembaban Menggunakan Metode Waterfall. *Arcitech: Journal of Computer Science and Artificial Intelligence*, *3*(2), 93. https://doi.org/10.29240/arcitech.v3i2.9334

Arora, R., & Arora, N. (2016). Analysis of SDLC Models. *International Journal of Current Engineering and Technology*, *6*(1), 2277–4106. http://inpressco.com/category/ijcet

Badrul, M., Ardy, R., Nusa Mandiri Jl Jatiwaringin Raya No, S., & Cipinang Melayu Jakarta Timur, K. (2021). Penerapan Metode Waterfall pada Perancangan Sistem Informasi Pendaftaran Siswa Baru. *Jurnal Sains Komputer & Informatika (J-SAKTI*, *5*(1), 52–61. https://doi.org/http://dx.doi.org/10.30645/j-sakti.v5i1.297

Barus, A. C., Harungguan, J., & Manulu, E. (2021). Pengujian api website untuk perbaikan performansi aplikasi ditenun. *Journal of Applied Technology and Informatics Indonesia*, *1*(2), 14–21.

bps.go.id. (2024). *Angka Deforestasi (Netto) Indonesia di Dalam dan di Luar Kawasan Hutan Tahun 2013-2022 (Ha/Th)*. Bps.Go.Id. https://www.bps.go.id/id/statistics-table/1/MjA4MSMx/angka-deforestasi--netto--indonesia-di-dalam-dan-di-luar-kawasan-hutan-tahun-2013-2022--ha-th-.html

Cahyadi, M. L., Herri Setiawan, H., & Mair, Z. R. (2023). *Analisis Perbandingan Kinerja Web Server Nginx dan Litespeed Menggunakan Httperf Dengan Sistem Operasi Debian*. https://doi.org/https://doi.org/10.47747/jpsii.v4i2.1739

Christian, Y., & Hengky, H. (2023). Analysis Of Software Developer Perceptions Towards The

Selection Of Javascript Framework In Batam City. *Jurnal Teknologi Informasi Dan Komunikasi*, *14*(1), 190–200. https://doi.org/10.51903/jtikp.v14i1.538

Deni Murdiani, & Muhamad Sobirin. (2022). Perbandingan Metodologi Waterfall Dan Rad (Rapid Application Development) Dalam Pengembangan Sistem Informasi. *Jurnal Informatika Teknologi Dan Sains*, *4*(4), 302–306. https://doi.org/10.51401/jinteks.v4i4.2008

Eka Putra, F. P., Muslim, F., Hasanah, N., Holipah, Paradina, R., & Alim, R. (2024). Analisis Komparasi Protokol Websocket dan MQTT Dalam Proses Push Notification. *Jurnal Sistim Informasi Dan Teknologi*, *5*, 63–72. https://doi.org/10.60083/jsisfotek.v5i4.325

Gede Endra Bratha, W. (2022). Literature Review Komponen Sistem Informasi Manajemen: Software, Database Dan Brainware. *Jurnal Ekonomi Manajemen Sistem Informasi*, *3*(3), 344–360. https://doi.org/10.31933/jemsi.v3i3.824

Handayani, A., Apriliani, E., Padrisi, Z., Albin Sugiartha, R., Arsyil Adzhim, M., Muhammad, F., Wicaksono, T., Anggi Saputro, F., Al Faridzi, F., & Linux Samsoni, F. (2023). Copyright @ Implementasi Sistem Keamanan Komputer Host Menggunakan Sistem Operasi. *Journal Of Social Science Research*. https://j-innovative.org/index.php/Innovative/article/view/377

Haniva, D. T., Ramadhan, J. A., & Suharso, A. (2023). Systematic Literature Review Penggunaan Metodologi Pengembangan Sistem Informasi Waterfall, Agile, dan Hybrid. *Journal of Information Engineering and Educational Technology*, *7*(1), 36–42. https://doi.org/10.26740/jieet.v7n1.p36-42

Herman, D. A., & Frederick, F. (2022). Pengembangan Dan Implementasi Frontend Dan Backend Website Perpustakaan Di SMA Maitreyawira Batam Menggunakan Model Addie. *National Conference for Community Service Project (NaCosPro)*, *4*(1), 1058–1064. https://doi.org/https://doi.org/10.37253/nacospro.v4i1.7076

Hidayat, A., Arief, V., Atina, W., Aldika, L., Niki, Y., Yudha, A., & Nugroho, A. S. (2020). Monitoring Suhu Dan Kelembaban Tanah Tanaman Buah Naga Berbasis IoT. *Seminar Nasional Terapan Riset Inovatif (SENTRINOV) Ke-6*, *6*(1), 1040–1047.

Maspupah, A. (2024). Literature Review: Advantages And Disadvantages Of Black Box And White Box Testing Methods. *Jurnal Techno Nusa Mandiri*, *21*(2), 151–162. https://doi.org/10.33480/techno.v21i2.5776

Maulana Syahaddan, M., Hakiki, R., Sulaeman, H., & Sahria, Y. (2024). Perancangan Sistem Monitoring dan Controlling Tanaman Menggunakan IoT. *JRIIN: Jurnal Riset Informatika Dan Inovasi*, *1*(7), 1015–1021.

Mufti Prasetiyo, S., Ivan Prayogi Nugroho, M., Lima Putri, R., & Fauzi, O. (2022). Pembahasan Mengenai Front-End Web Developer dalam Ruang Lingkup Web Development. *Jurnal Multidisiplin Ilmu*, *1*(6), 1015–1020. https://journal.mediapublikasi.id/index.php/bullet

Noor Saputra, M. R., Alzami, F., Biyantama, K., Abdillah, M. R., Steven, A., Umam, C., Nurhindarto, A., & Wahyudi, F. (2023). Blueprint for Recording Data on Red Onion Farming Activities Using the Bpr Method. *Jurnal Teknik Informatika (Jutif)*, *4*(2), 311–319. https://doi.org/10.52436/1.jutif.2023.4.2.748

Normah, Rifai, B., Vambudi, S., & Maulana, R. (2022). Analisa Sentimen Perkembangan Vtuber Dengan Metode Support Vector Machine Berbasis SMOTE. *Jurnal Teknik Komputer AMIK BSI*, *8*(2), 174–180. https://doi.org/10.31294/jtk.v4i2

Nugraha, I. R., Putra, W. H. N., & Setiawan, E. (2024). A Comparative Study of HTTP and MQTT for IoT Applications in Hydroponics. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *8*(1), 119–126. https://doi.org/10.29207/resti.v8i1.5561

Nur, A., Zumaroh, A., Maulida, T., Akbar, H., Rozaq, A., Ananda, R. S., Yahya Syafa'at, A., & Tahyudin, I. (2022). DEVELOPMENT OF APPLICATION PROGRAMMING INTERFACE (API) FOR AMIKOM PURWOKERTO HANDSANITIZER (AMPUH) DATA LOGGER VISUALIZATION. *Jurnal Teknik Informatika (JUTIF)*, *3*(3), 791–796. https://doi.org/10.20884/1.jutif.2022.3.3.222

Pahlevi, O., Mulyani, A., & Khoir, M. (2018). SISTEM INFORMASI INVENTORI BARANG MENGGUNAKAN METODE OBJECT ORIENTED DI PT. LIVAZA TEKNOLOGI INDONESIA JAKARTA. *Jurnal PROSISKO*, *5*(1). https://livaza.com/.

Pahlevi, R. R., Sukarno, P., & Erfianto, B. (2019). Implementation of Event-Based Dynamic Authentication on MQTT Protocol. *Jurnal Rekayasa Elektrika*, *15*(2).

https://doi.org/10.17529/jre.v15i2.13963
Pratama, D. B., & Armin, A. P. (2024). Pengembangan Sistem Informasi Aplikasi Mobile Pelayanan Elektronik Dispendukcapil Kota Malang. *Jurnal Ilmu Siber Dan Teknologi Digital*, *3*(1), 11–41. https://doi.org/https://doi.org/10.35912/jisted.v3i1.5098
Pratama, R. F., Wicaksono, R. S. R., & Pramudhita, A. N. (2023). Perancangan Dan Implementasi Protokol Mqtt Pada Sistem Parkir Cerdas Berbasis Iot. *Jurnal Informatika Dan Teknik Elektro Terapan*, *11*(3), 475–483. https://doi.org/10.23960/jitet.v11i3.3191
Pricillia, T., & Zulfachmi. (2021). Perbandingan Metode Pengembangan Perangkat Lunak (Waterfall, Prototype, RAD). *Jurnal Bangkit Indonesia*, *10*(1), 6–12. https://doi.org/10.52771/bangkitindonesia.v10i1.153
Putra, G. D. A., Rahmadi, A. A., & Armin, A. P. (2025). Rancang Bangun Sistem Informasi Manajemen Klinik Chania Care Center Berbasis Web Responsif. *Jurnal Ilmu Siber Dan Teknologi Digital*, *3*(2), 153–179. https://doi.org/10.35912/jisted.v3i2.5095
Rafiussani, M. I. A., & Armin, A. P. (2024). Rancang Bangun Aplikasi Pencatatan Keuangan dan Manajemen Pesanan untuk Bisnis Usaha Lontong. *Jurnal Ilmu Siber Dan Teknologi Digital*, *3*(1), 85–110. https://doi.org/https://doi.org/10.35912/jisted.v3i1.5152
Sayekti, I., Supriyo, B., Kusumastuti, S., Krishna, B., Kartika, V. S., Utomo, K., Dadi, D., Beta, S., Pramuji, T., & Aji, A. F. (2022). Pendampingan penerapan teknologi sistem monitoring dan penyiraman berbasis IoT pada budidaya tanaman obat keluarga. *ABSYARA: Jurnal Pengabdian Pada Masyarakat*, *3*(1), 150–158. https://doi.org/10.29408/ab.v3i1.5616
Selatan, T. (2020). *Implementasi Metode White Box Testing Pada Proses Quality Assurance Perangkat Lunak Berbasis Web Dan Mobile Collection System*. *XV*(10), 57–63.
Suliyanti, W. N. (2019). *STUDI LITERATUR BASIS DATA SQL DAN NOSQL*. *8*(1).
Syahril, M., & Ramdhani, M. F. (2024). Sistem Administrator Debt Recording Berbasis Website pada Toko Serunai Sekayu. *Jurnal Ilmu Siber Dan Teknologi Digital*, *2*(2), 75–107. https://doi.org/https://doi.org/10.35912/jisted.v3i1.5152
Utamy, A. R., Siswanto, & Sutarti. (2023). Prototype Wireless Sensor Network Sistem Pengukuran Debu Dan Suhu Udara Berbasis Mqtt Server. *PROSISKO: Jurnal Pengembangan Riset Dan Observasi Sistem Komputer*, *10*(2), 152–164. https://doi.org/10.30656/prosisko.v10i2.7158
Utomo, K. B., Azizah, A., & Pangestu, M. A. (2022). Peran Computer Assited Test dalam Implementasi Penilaian di SD Negeri 005 Palaran. *Jurnal Ilmu Siber Dan Teknologi Digital*, *1*(1), 29–39. https://doi.org/10.35912/jisted.v1i1.1529