SISTEM PEMANTAUAN LIMBAH CAIR INDUSTRI

TERINTEGRASI INTERNET OF THINGS DAN

JAVASCRIPT OBJECT NOTATION WEB TOKEN

BERBASIS WEBSITE DAN ANDROID



Disusun oleh:

Imam Zaenal Abidin 4.31.21.1.15

Muhammad Adriano Khairur Rizky Setyawan 4.31.21.1.18

PROGRAM STUDI SARJANA TERAPAN TEKNIK TELEKOMUNIKASI

JURUSAN TEKNIK ELEKTRO

POLITEKNIK NEGERI SEMARANG

2025

SISTEM PEMANTAUAN LIMBAH CAIR INDUSTRI

TERINTEGRASI INTERNET OF THINGS DAN

JAVASCRIPT OBJECT NOTATION WEB TOKEN

BERBASIS WEBSITE DAN ANDROID



Tugas akhir ini disusun untuk melengkapi sebagian persyaratan menjadi Sarjana Terapan

Disusun oleh:

Imam Zaenal Abidin 4.31.21.1.15

Muhammad Adriano Khairur Rizky Setyawan 4.31.21.1.18

PROGRAM STUDI SARJANA TERAPAN TEKNIK TELEKOMUNIKASI

JURUSAN TEKNIK ELEKTRO

POLITEKNIK NEGERI SEMARANG

2025

PERNYATAAN KEASLIAN TUGAS AKHIR

Saya menyatakan dengan sesungguhnya bahwa tugas akhir dengan judul **“Sistem Pemantauan Limbah Cair Industri Terintegrasi *Internet of Things* dan *Javascript Object Notation Web Token* Berbasis *Website* dan Android”** yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Terapan pada Program Studi Teknik Telekomunikasi, Jurusan Teknik Elektro Politeknik Negeri Semarang, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari tugas akhir yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar Sarjana Terapan di lingkungan Politeknik Negeri Semarang maupun di perguruan tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

|  |  |
| --- | --- |
|  | Semarang, 18 Juli 2025 |
| Mahasiswa I | Mahasiswa II |
|  |  |
| Imam Zaenal Abidin | Muhammad Adriano K. R.S. |
| 4.31.21.1.15 | 4.31.21.1.18 |

HALAMAN PERSETUJUAN

Tugas akhir dengan judul **“Sistem Pemantauan Limbah Cair Industri Terintegrasi *Internet of Things* dan *Javascript Object Notation Web Token* Berbasis *Website* dan Android”**  dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Terapan pada Program Studi Teknik Telekomunikasi, Jurusan Elektro Politeknik Negeri Semarang dan disetujui untuk diajukan dalam sidang ujian tugas akhir.

|  |  |
| --- | --- |
|  | Semarang, 18 Juli 2025 |
| Pembimbing I, | Pembimbing II, |
|  |  |
| Dr. Amin Suharjono, S.T., M.T. | Roni Apriantoro, S.Tr.T., M.Tr.T. |
| NIP. 197210271999031002 | NIP. 199604092022031007 |

|  |
| --- |
| Mengetahui, |
| Ketua Program Studi Sarjana Terapan Teknik Telekomunikasi |
|  |
| Ari Sriyanto Nugroho, S.T, M.T., M.Sc |
| NIP. 197409042005011001 |

HALAMAN PENGESAHAN

Tugas akhir dengan judul **“Sistem Pemantauan Limbah Cair Industri Terintegrasi *Internet of Things* dan *Javascript Object Notation Web Token* Berbasis *Website* dan Android”** Telah dipertahankan dalam ujian wawancara dan diterima sebagai syarat untuk menjadi Sarjana Terapan pada Program Studi Teknik Telekomunikasi, Jurusan Teknik Elektro Politeknik Negeri Semarang pada tanggal.............

|  |  |  |
| --- | --- | --- |
| Tim Penguji | | |
| Penguji I, | Penguji II, | Penguji II, |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| Ketua Penguji, | Sekretaris Penguji, |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| Mengesahkan, |
| Ketua Jurusan Teknik Elektro |
|  |
| Yusnan Badruzzaman, S.T., M.Eng. |
| NIP. 197503132006041001 |

PRAKATA

Puji syukur Penulis panjatkan kepada Tuhan Yang Maha Esa. Atas berkat dan rahmat-Nya, Penulis dapat menyelesaikan tugas akhir ini yang berjudul **“Sistem Pemantauan Limbah Cair Industri Terintegrasi *Internet of Things* dan *Javascript Object Notation Web Token* Berbasis *Website* dan Android”** dengan baik. Tugas akhir ini disusun untuk memenuhi salah satu persyaratan kelulusan Program Studi Sarjana Terapan Teknik Telekomunikasi Jurusan Teknik Elektro Politeknik Negeri Semarang.

Selama proses penyusunan tugas akhir ini, penulis menghadapi berbagai hambatan, baik dari segi teknis maupun non-teknis. Namun, dengan bantuan dari berbagai pihak dan tekad yang kuat, hambatan-hambatan tersebut dapat diatasi dengan baik. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih kepada:

* + 1. Ibu Dr. Eni Dwi Wardihani, S.T., M.T., Selaku Direktur Politeknik Negeri Semarang,
    2. Bapak Yusnan Badruzzaman, S.T., M.Eng., selaku Ketua Jurusan Teknik Elektro Politeknik Negeri Semarang,
    3. Bapak Ari Sriyanto Nugroho, S.T, M.T.,M.Sc, selaku Ketua Program Studi Sarjana Terapan Teknik Telekomunikasi Politeknik Negeri Semarang,
    4. Bapak Dr. Amin Suharjono, S.T., M.T., selaku Dosen Pembimbing I,
    5. Bapak Roni Apriantoro, S.Tr.T., M.Tr.T., selaku Dosen Pembimbing II,
    6. Dosen - dosen Program Studi Sarjana Terapan Teknik Telekomunikasi yang telah memberikan bimbingan, arahan, dan ilmu selama menempuh studi.
    7. Orang tua serta keluarga penulis yang telah ikut memberikan doa, semangat, serta dukungan lain yang tak terhingga jumlahnya.
    8. Teman-teman kelas TE-4B yang senantiasa memberikan dukungan dan motivasi kepada penulis, baik berupa materi maupun spiritual.

Penulis menyadari bahwa tugas akhir ini masih memiliki banyak kekurangan. Oleh karena itu, penulis sangat mengharapkan kritik dan saran dari para pembaca guna penyempurnaan di masa mendatang. Penulis berharap tugas akhir ini dapat memberikan manfaat baik bagi penulis sendiri, maupun bagi para pembaca.

Semarang, 18 Juli 2025

Penulis

ABSTRAK

*Imam Zaenal Abidin dan Muhammad Adriano Khairur Risky Setyawan.* ***“Sistem Pemantauan Limbah Cair Industri Terintegrasi Internet of Things dan Javascript Object Notation Web Token Berbasis Website dan Android”****, Tugas Akhir Sarjana Terapan Jurusan Teknik Elektro Politeknik Negeri Semarang, di bawah bimbingan Bapak Dr. Amin Suharjono, S.T., M.T. dan Bapak Roni Apriantoro, S.Tr.T., M.Tr.T., 2025, …. halaman.*

**Isi abstrak**

*ABSTACT*

*Imam Zaenal Abidin dan Muhammad Adriano Khairur Risky Setyawan.* ***“Integrated Industrial Liquid Waste Monitoring System Using the Internet of Things and JavaScript Object Notation Web Token Based on Website and Android”****, Final Project for Applied Bachelor's Degree in Electrical Engineering, Semarang State Polytechnic, under the supervision of Dr. Amin Suharjono, S.T., M.T. and Roni Apriantoro, S.Tr.T., M.Tr.T., 2025, …. pages*

DAFTAR ISI

[PERNYATAAN KEASLIAN TUGAS AKHIR ii](#_Toc203773471)

[HALAMAN PERSETUJUAN iii](#_Toc203773472)

[HALAMAN PENGESAHAN iv](#_Toc203773473)

[PRAKATA v](#_Toc203773474)

[ABSTRAK vii](#_Toc203773475)

[DAFTAR ISI viii](#_Toc203773476)

[DAFTAR GAMBAR xi](#_Toc203773477)

[DAFTAR TABEL xii](#_Toc203773478)

[DAFTAR LAMPIRAN xiii](#_Toc203773479)

[BAB I PENDAHULUAN 1](#_Toc203773480)

[1.1 Latar Belakang 1](#_Toc203773482)

[1.2 Rumusan Masalah 3](#_Toc203773483)

[1.3 Tujuan 3](#_Toc203773484)

[1.4 Manfaat 3](#_Toc203773485)

[1.5 Batasan Masalah 4](#_Toc203773486)

[1.6 Sistematika Penulisan 4](#_Toc203773487)

[BAB II TINJAUAN PUSTAKA DAN DASAR TEORI 6](#_Toc203773488)

[2.1 Tinjauan Pustaka 6](#_Toc203773490)

[2.2 Dasar Teori 9](#_Toc203773491)

[2.2.1 *Internet of Things* (IoT) 9](#_Toc203773492)

[2.2.2 *Application Programming Interface* (API) 11](#_Toc203773493)

[2.2.3 Protokol Komunikasi 12](#_Toc203773494)

[2.2.4 JSON *Web Token* (JWT) 13](#_Toc203773495)

[2.2.5 *Hypertext Transfer Protocol Secure* (HTTPS) 13](#_Toc203773496)

[2.2.6 Git 14](#_Toc203773497)

[*2.2.7* *Javascript* 14](#_Toc203773498)

[2.2.8 *JavaScript* *Object Notation* (JSON) 15](#_Toc203773499)

[2.2.9 Next.js 16](#_Toc203773500)

[2.2.10 *Database* 17](#_Toc203773501)

[2.2.11 MySQL 18](#_Toc203773502)

[2.2.12 *Message Broker* 19](#_Toc203773503)

[2.2.13 ChirpStack 19](#_Toc203773504)

[2.2.14 *Balck Box Testing* 20](#_Toc203773505)

[2.2.15 *Quality of Service* (QoS) 20](#_Toc203773506)

[BAB III METODE PERANCANGAN SISTEM 23](#_Toc203773507)

[3.1 Pengumpulan Data 23](#_Toc203773509)

[3.2 Perancangan 25](#_Toc203773510)

[3.2.1 Arsitektur Sistem 25](#_Toc203773511)

[3.2.2 Diagram *Use Case* 27](#_Toc203773512)

[3.2.3 *Entity Relationship Diagram* (ERD) 28](#_Toc203773513)

[3.2.4 Tampilan UI pada Aplikasi 29](#_Toc203773514)

[3.3 Pembuatan 32](#_Toc203773515)

[3.4 Pengujian 32](#_Toc203773516)

[3.4.1 *Black Box Testing* (Pengujian fungsionalitas) 32](#_Toc203773517)

[3.4.2 Pengujian Sistem Keamanan 35](#_Toc203773518)

[3.4.3 Pengujian Notifikasi 36](#_Toc203773519)

[3.4.4 Pengujian Performa 37](#_Toc203773520)

[3.4.5 Pengujian QoS Protokol Komunikasi 38](#_Toc203773521)

[3.5 Evaluasi 39](#_Toc203773522)

[BAB IV HASIL PENGUJIAN DAN ANALISIS 40](#_Toc203773523)

[BAB V PENUTUP 41](#_Toc203773525)

[DAFTAR PUSTAKA 42](#_Toc203773527)

DAFTAR GAMBAR

DAFTAR TABEL

[Tabel 2.1 Ringkasan Tinjauan Pustaka 8](#_Toc204960493)

[Tabel 2.2 Kategori *Delay* (ETSI,1999) 21](#_Toc204960494)

[Tabel 2.3 Kategori *Jitter* (ETSI, 1999) 21](#_Toc204960495)

[Tabel 2.4 Kategori *Packet Loss* (ETSI, 1999) 22](#_Toc204960496)

[Tabel 3.1 Rancangan tampilan antarmuka aplikasi *website* 29](#_Toc204960497)

[Tabel 3.2 Rancangan tampilan antarmuka aplikasi Android 31](#_Toc204960498)

[Tabel 3.3 Rancangan pengujian fungsionalitas aplikasi *website* dan Android 33](#_Toc204960499)

[Tabel 3.4 Rancangan pengujian sistem keamanan 36](#_Toc204960500)

[Tabel 3.5 Rancangan pengujian notifikasi 37](#_Toc204960501)

[Tabel 3.6 Rancangan pengujian performa aplikasi *website* 39](#_Toc204960502)

[Tabel 4.1 Hasil pengujian fungsionalitas aplikasi *web* 40](#_Toc204960503)

[Tabel 4.2 Hasil pengujian fungsionalitas aplikasi android 44](#_Toc204960504)

[Tabel 4.3 Hasil pengujian sistem keamanan 67](#_Toc204960505)

[Tabel 4.4 Hasil pengujian notifikasi 68](#_Toc204960506)

[Tabel 4.5 Hasil pengujian performa aplikasi *website* 70](#_Toc204960507)

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1. Latar Belakang

Pemantauan adalah kegiatan pengumpulan data secara terstruktur dan terus-menerus untuk memantau suatu kondisi, parameter, atau proses tertentu (Gonzales dkk., 2021). Dalam konteks pengelolaan lingkungan, sistem pemantauan dirancang untuk memastikan kepatuhan terhadap regulasi, mendeteksi anomali, serta memberikan data yang akurat untuk pengambilan keputusan (Nurbaya, 2018). Sistem pemantauan terdiri dari perangkat keras, seperti sensor dan data logger, serta perangkat lunak yang mengolah data menjadi informasi yang dapat dianalisis. Salah satu contoh penerapan dari sistem pemantauan adalah pada pemantauan limbah cair industri.

Peraturan regulasi terkait pemantauan limbah cair industri telah ditetapkan oleh Kementrian Lingkungan Hidup dan Kehutanan (KLHK) dalam Peraturan Nomor P.80/MENLHK/SETJEN/KUM.1/10/2019 terkait pemantauan kualitas air limbah. Regulasi ini mewajibkan tindakan pemantauan kualitas air limbah secara berkala pada industri dengan menggunakan Sistem Pemantauan Kualitas Air Limbah secara Terus Menerus dan Dalam Jaringan (SPARING). SPARING merupakan instrumen pemantauan pencemaran lingkungan hidup dan pemenuhan baku mutu air limbah. SPARING terdiri dari alat pemantauan, data logger, dan pusat data yang mengolah informasi hasil pemantauan. Adapun parameter air limbah yang dipantau yaitu Potential of Hydrogen (pH), Chemical Oxygen Demand (COD), Total Suspended Solid (TSS), amonia nitrogen (NH3-N), dan debit (Nurbaya, 2019).

Untuk memastikan kepatuhan terhadap regulasi SPARING yang ditetapkan oleh KLHK, teknologi pemantauan terus berkembang seiring dengan kemajuan digital. Di era Revolusi Industri 4.0, Internet of Things (IoT) kerap digunakan agar pemantauan menjadi lebih efektif. IoT memungkinkan perangkat pemantauan seperti sensor untuk berkomunikasi satu sama lain melalui jaringan internet, sehingga dapat mempermudah pengumpulan data secara otomatis (Sawitri, 2023). Dengan meningkatnya jumlah perangkat IoT yang tersebar, tantangan baru muncul dalam pembuatan sistem pemantauan IoT yang efektif dan efisien (Sari, 2024).

Sebagian besar sistem pemantauan IoT yang sudah ada hanya menyediakan pemantauan pasif, tanpa adanya peringatan atau notifikasi otomatis saat parameter lingkungan melebihi ambang batas aman (Gonzales dkk., 2021). Selain itu, pemantauan hanya dapat dilakukan melalui website, sehingga menyulitkan pemantauan oleh pengguna dengan perangkat mobile (Akasiadis dkk., 2019). Kemudian dalam hal interoperabilitas, protokol komunikasi menjadi aspek yang krusial pada sebuah perangkat dalam sebuah sistem IoT. Hal ini mengakibatkan perangkat yang menggunakan protokol yang berbeda sulit untuk dikelola dalam satu ekosistem (Pramukantoro, 2020). Tantangan lain yang berhubungan erat dengan interoperabilitas adalah masalah keamanan. Dengan semakin banyaknya perangkat yang terhubung ke internet, keamanan data dalam sebuah sistem IoT akan semakin rentan terhadap resiko serangan siber, pencurian data, dan eksploitasi perangkat (Shakdher dkk., 2019).

Masalah keamanan pada sistem IoT merupakan masalah serius yang harus ditangani. Untuk mengatasi masalah tersebut, diperlukan sebuah mekanisme keamanan agar dapat melindungi integritas dan autentikasi data yang dikirim dan diterima oleh perangkat IoT. Salah satu mekanisme keamanan yang digunakan adalah JavaScript Object Notation Web Token (JWT) (Ahmed dkk., 2019). JWT merupakan standar token berbasis JSON yang aman dan banyak digunakan dalam aplikasi web. Penerapan JWT memastikan verifikasi identitas pengguna atau perangkat lebih efisien dan aman, serta dapat mengurangi beban server sehingga meningkatkan skalabilitas sistem. Selain itu, JWT juga dapat memastikan otorisasi kepada pengguna atau perangkat untuk mengakses dan mengendalikan perangkat IoT (Darmawan dkk., 2023).

Hingga saat ini telah banyak penelitian yang merancang dan membuat sebuah alat IoT untuk memantau limbah cair, namun belum ada sistem pemantauan yang memadai secara khusus berbasis web atau aplikasi mobile yang dapat memantau dan meningkatkan respons terhadap kondisi kritis dari parameter sensor secara real-time, serta dapat mengatasi masalah terkait interoperabilitas dan keamanan. Oleh karena itu, tugas akhir ini bertujuan untuk merancang dan mengimplementasikan sistem pemantauan limbah cair industri yang mudah dan aman. Maka penulis merancang penelitian dengan judul “Sistem Pemantauan Limbah Cair Industri Terintegrasi Internet of Things dan JavaScript Object Notation Web Token Berbasis Website dan Android”, yang dapat memberikan solusi efektif dalam mengatasi permasalahan pengelolaan IoT tersebut.

1. Rumusan Masalah

Berdasarkan latar belakang tersebut, dapat dirumuskan permasalahannya yaitu:

1. Keterbatasan sistem pemantauan limbah cair industri dalam hal aksesibilitas dan kemudahan penggunaan secara mandiri oleh pengguna.
2. Belum diterapkannya sistem pemantauan limbah cair industri yang dapat mendukung beberapa protokol komunikasi.
3. Belum diterapkannya sistem keamanan pada sistem pemantauan limbah cair industri, yang berpotensi menyebabkan perubahan pada data secara ilegal.
4. Perlu adanya fitur notifikasi untuk memberikan peringatan secara cepat dan mudah kepada pengguna.
5. Tujuan

Tujuan dari pembuatan tugas akhir ini adalah:

1. Membuat antarmuka pemantauan berbasis *website* dan Android.
2. Menerapkan integrasi beberapa protokol komunikasi.
3. Menerapkan sistem keamanan menggunakan JWT dan HTTPS.
4. Menerapkan fitur notifikasi perangkat IoT untuk pengguna.
5. Manfaat

Manfaat setelah pembuatan tugas akhir adalah:

1. Akan mempermudah penyampaian informasi terhadap situasi darurat kepada tim teknis atau operator jika parameter limbah di luar batas aman, sehingga kondisi yang memerlukan tindakan cepat dapat segera terdeteksi, seperti halnya penyesuaian pada aliran limbah.
2. Akan mempermudah pengembangan sistem IoT secara berkelanjutan, seperti penambahan perangkat IoT baru tanpa merubah infrastruktur yang sudah ada.
3. Akan meningkatkan keamanan data, sehingga kerahasiaan data terkait parameter limbah industri maupun data pribadi pengguna akan terjaga.
4. Akan mendukung kepatuhan terhadap regulasi lingkungan terkait standar emisi yang sudah ditetapkan, sehingga dapat mengurangi risiko pelanggaran dan turut serta menjaga kelestarian lingkungan.
5. Batasan Masalah

Berdasarkan latar belakang, maka batasan masalahnya adalah:

1. Pembahasan akan berfokus pada pembuatan *platform* pemantauan IoT.
2. Sistem IoT akan menggunakan mikrokontroler ESP32.
3. Parameter air yang akan diukur adalah pH, polutan organik, kekeruhan, amonia dan debit.
4. Protokol komunikasi yang akan digunakan adalah HTTP, MQTT, dan LoRaWAN.
5. Fitur yang akan dibuat adalah autentikasi dan otorisasi pengguna, grafik visualisasi data, manajemen perangkat IoT, serta alarm notifikasi.
6. Sistematika Penulisan

Dalam penyusunan tugas akhir ini dilakukan pengelompokkan isi dalam beberapa bab. Bagian yang dapat berdiri sendiri dipisahkan dengan bagian yang lain dan ditempatkan dalam bab tersendiri dengan maksud mempermudah pemahamam. Adapun sistematika penulisannya adalah sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini membahas mengenai latar belakang dari tugas akhir ini, perumusan masalah pada tugas akhir, tujuan dari tugas akhir, manfaat yang bisa dipetik dari tugas akhir, pembatasan masalah dari tugas akhir, dan sistematika penulisan pada tugas akhir.

**BAB II TINJAUAN PUSTAKA DAN DASAR TEORI**

Bab ini membahas mengenai tinjauan pustaka yang merupakan referensi dalam pembuatan tugas akhir ini dan dasar teori yang mendukung pembuatan tugas akhir.

**BAB III METODE PERANCANGAN SISTEM**

Bab ini membahas mengenai proses perancangan dan pengujian dari rancang bangun perangkat lunak (Web dan Android) untuk sistem pemantauan limbah cair industri.

**BAB IV HASIL PENGUJIAN DAN ANALISIS**

Bab ini membahas mengenai hasil pengujian sistem beserta analisi data hasil pengujian dari rancang bangun perangkat lunak (Web dan Android) dari sistem pemantauan limbah cair industri.

**BAB V PENUTUP**

Bab ini berisi tentang kesimpulan dan saran yang diperoleh dari analisis yang telah dilakukan pada bab sebelumnya.

**DAFTAR PUSTAKA**

Daftar pustaka berisi sumber-sumber, jurnal, studi Pustaka yang penulis cantum dalam proses penyelesaian tugas akhir ini.

**LAMPIRAN**

Lampiran berisi data atau pelengkap atau hasil olahan yang menunjang penulisan laporan tugas akhir tetapi tidak dicantumkan di dalam isi tugas akhir, karena akan mengganggu kesinambungan penulisan.

BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

1. Tinjauan Pustaka

Dalam penyusunan tugas akhir ini, penulis mengambil beberapa penelitian terkait dengan sistem pemantauan IoT dan permasalahannya. Salah satu tantangan utama yang diangkat adalah keamanan protokol komunikasi IoT, seperti yang dijelaskan oleh Shakdher dkk. (2019) dalam penelitiannya yang berjudul ”Security Vulnerabilities in Consumer IoT Applications”. Penelitian ini mengungkap kelemahan pada perangkat IoT yang mentransmisikan data dalam format teks biasa, sehingga rentan terhadap serangan Man In The Middle (MITM). Peneliti menggunakan aplikasi Wireshark untuk menganalisa trafik data dan menyoroti perlunya fitur keamanan tambahan seperti hashing untuk melindungi data dari ancaman serangan siber. Menanggapi hal tersebut, Shodiq dkk. (2021) dalam penelitiannya yang berjudul ”Secure MQTT Authentication and Message Exchange Methods for IoT Constrained Device” mengusulkan mekanisme autentikasi berbasis JWT untuk protokol MQTT. Penambahan ini bertujuan melindungi jaringan dari akses tidak sah serta meningkatkan keamanan data dengan enkripsi XXTEA. Hasil validasi menunjukkan mekanisme ini efektif terhadap serangan brute force dan MITM, sekaligus memastikan kompatibilitas dengan perangkat berdaya rendah. Solusi ini memberikan kontribusi penting dalam mengatasi kerentanan keamanan tanpa mengorbankan performa sistem.

Penelitian terkait pemantauan kualitas air telah dilakukan oleh Bogdan dkk. (2023), dalam penelitiannya yang berjudul ”Low-Cost Internet-of-Things Water-Quality Monitoring System for Rural Areas”. Mereka mengembangkan sistem berbasis Arduino UNO dengan beberapa sensor untuk mengukur parameter seperti suhu air, pH, Total Dissolved Solid (TDS), dan Turbidity (kekeruhan). Sistem ini memanfaatkan komunikasi Bluetooth untuk mengirim data ke aplikasi mobile berbasis Android yang memungkinkan pemantauan secara real-time. Meskipun memberikan hasil yang relevan, seperti identifikasi sumber air yang tidak layak konsumsi, sistem ini masih memiliki keterbatasan dalam jangkauan transmisi data. Untuk mengatasi kendala jangkauan tersebut, diperlukan peningkatan teknologi komunikasi jarak jauh yang lebih andal dan efisien, seperti yang dilakukan oleh Jáquez dkk. (2023) dalam penelitiannya yang berjudul ”Extension of LoRa coverage and integration of an unsupervised anomaly detection algorithm in an IoT water quality monitoring system”. Mereka mengintegrasikan repeater LoRa untuk memperluas cakupan komunikasi menggunakan protokol LoRaWAN agar dapat memantau pH, TDS, dan suhu air melalui website. Purwarupa ini terbukti mampu meningkatkan performa komunikasi jarak jauh dengan tingkat kehilangan paket yang rendah dan sensitivitas tinggi. Sementara itu, Raman & Martin (2024) dalam penelitiannya yang berjudul ”IoT-Enabled Water Pollution Detection for Real-Time Monitoring and Pollution Source Identification with MQTT Protocol” mengembangkan platform IoT yang memanfaatkan protokol MQTT untuk pengiriman data sensor secara real-time. Sensor digunakan untuk mengukur parameter penting seperti komposisi kimia, pH, Dissolved Oxygen (DO), dan amonia (NH3-N). Dengan menggunakan protokol ini, informasi yang didapat dari perangkat IoT dapat diproses lebih cepat untuk mengidentifikasi sumber pencemaran, sehingga memungkinkan tindakan pencegahan yang lebih efektif.

Di sisi lain, penelitian terkait pemantauan limbah cair juga berkembang pesat. Salem dkk. (2022) menggunakan ESP8266 untuk memantau pH dan suhu limbah cair melalui platform ThingSpeak dalam penelitiannya yang berjudul ”An Industrial Cloud-Based IoT System for Real-Time Monitoring and Controlling of Wastewater”. Sistem ini juga dilengkapi layanan Short Message Service (SMS) untuk notifikasi, meskipun masih terdapat kelemahan pada sisi keamanan data karena belum menggunakan metode hashing. Begitupun sistem yang diusulkan oleh Febriana & Farros (2024) dalam penelitiannya yang berjudul ”Rancang bangun Industrial Internet of Things Board Multiguna untuk Monitoring Limbah Cair”, mereka merancang sebuah board IIoT multiguna dengan konektivitas LoRa untuk memantau parameter limbah seperti pH, Chemical Oxygen Demand (COD), Total Suspended Solid (TSS), NH3-N, dan Water flow meter (Debit air). Data sensor disimpan di kartu SD dan ditampilkan pada Nextion Display, serta dapat dipantau melalui platform ThingSpeak. Namun, sistem IoT yang telah dibuat belum memiliki platform pemantauan yang efektif berbasis mobile (Android atau iOS). Sistem ini juga belum memiliki layanan notifikasi yang bisa memberitahu pengguna tentang perubahan nilai sensor yang signifikan.

Berdasarkan penelitian-penelitian sebelumnya, telah banyak penelitian yang mirip dengan tugas akhir yang dilakukan. Namun masih banyak kelemahan dari penelitian diatas, seperti kurang lengkapnya protokol komunikasi yang didukung oleh platform, serta belum diterapkannya sistem keamanan data dan fitur notifikasi pada platform. Kelebihan penelitian yang saat ini dilakukan adalah penerapan message broker dan LoRa network server untuk mengintegrasikan protokol komunikasi MQTT dan LoraWAN pada sistem yang dibuat. Kemudian terdapat tambahan enkripsi HTTPS dan hashing JWT untuk menjaga keamanan dan integritas data. Fitur notifikasi melalui email dan bawaan aplikasi juga akan diterapkan agar pengguna dapat mengetahui anomali pada nilai sensor dengan lebih mudah dan cepat. Tabel 2.1 merupakan tabel yang berisi ringkasan tinjauan pustaka yang telah dijelaskan diatas.

1. Ringkasan Tinjauan Pustaka

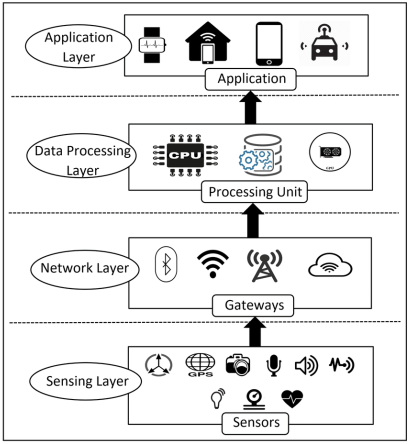
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Judul Penelitian | **Sensor** | **Antarmuka** | **Protokol** | **Notifikasi** | **Keamanan** |
| (Febriana & Farros, 2024) | COD, pH,  TSS, NH3-N, Debit air | *Website* | HTTP | - | - |
| (Raman & Martin, 2024) | NH3-N, DO, pH, COD | *Website* | MQTT | - | - |
| (Salem dkk., 2022) | Suhu air, pH | *Website* | HTTP | SMS | - |
| (Jáquez dkk., 2023) | Suhu air, pH, TDS | *Website* | LoraWAN | Email | - |
| (Bogdan dkk., 2023) | Suhu air, pH, TDS, Kekeruhan | *Android* | HTTP | *Built-In* (Bawaan) | - |
| (Shakdher dkk., 2019) | - | *Website* | HTTP | - | - |
| (Shodiq dkk., 2021) | - | *Website* | MQTT | - | JWT, Enkripsi XXTEA |
| Usulan Tugas Akhir | COD, pH, Debit air, NH3-N, Kekeruhan | *Website,* Android | HTTP, MQTT, LoraWAN | *Built-In* (Bawaan) | JWT, HTTPS |

1. Dasar Teori

Pada bagian ini menjelaskan mengenai teori-teori yang digunakan dalam pembuatan tugas akhir yang berjudul Sistem Pemantauan Limbah Cair Industri Terintegrasi *Internet of Things* dan *Javascript Object Notation Web Token* Berbasis Website dan Android.

1. *Internet of Things* (IoT)

IoT adalah paradigma teknologi yang menghubungkan berbagai perangkat fisik, seperti sensor, aktuator, dan mikrokontroler ke dalam jaringan internet untuk memungkinkan komunikasi data secara otomatis dan *real-time* (Shakdher dkk., 2019). IoT memanfaatkan kombinasi teknologi perangkat keras (seperti sensor dan mikrokontroler), protokol komunikasi, serta sistem *cloud* atau *edge computing* guna menciptakan sistem yang cerdas, responsif, dan adaptif terhadap lingkungan sekitarnya. Dalam arsitektur sistem pemantauan limbah cair industri, IoT berperan sebagai fondasi utama yang memungkinkan pengawasan kualitas air secara berkelanjutan dengan tingkat akurasi tinggi. Sistem ini terdiri dari beberapa lapisan yang saling terintegrasi. Gambar 2.1 merupakan ilustrasi dari beberapa lapisan tersebut.



1. Arsitektur IoT

(Sumber: Dokumentasi Pribadi)

Setiap lapisan memiliki tanggung jawab spesifik dalam proses akuisisi data, transmisi informasi, hingga penyajian layanan kepada penguna akhir. Untuk memperjelas fungsi dan peran masing-masing komponen dalam sistem, penjelasan berikut disusun berdasarkan pembagian umum lapisan IoT.

*Sensing Layer*

Lapisan ini merupakan fondasi dari sistem IoT yang terdiri dari berbagai sensor dan aktuator yang mengumpulkan data dari lingkungan fisik. Lapisan ini terdiri atas sensor-sensor fisis seperti sensor pH, suhu, cahaya, tekanan, suara, dan sebagainya. Sensor ini dikendalikan oleh mikrokontroler dan bertugas mengakuisisi data secara periodik. Data yang diperoleh bersifat mentah dan perlu dikirimkan ke lapisan berikutnya untuk diproses lebih lanjut.

*Network Layer*

Lapisan ini bertanggung jawab atas transmisi data dari *sensing layer* ke *data processing layer*. Lapisan ini terdiri dari berbagai macam perangkat jaringan yang saling terhubung agar dapat menunjang komunikasi data antar jaringan lokal maupun internet.

*Data Processing Layer*

Pada lapisan ini, data yang diterima dari perangkat IoT (*node*) akan diproses, disimpan, dan dianalisis. Komponen utama dalam layer ini adalah *server*, yang dapat berupa *cloud-based server* atau *on-premise server*. Teknologi seperti *edge computing*, *machine learning*, atau sistem basis data digunakan untuk menghasilkan informasi yang lebih bermakna dari data mentah yang dikumpulkan sensor.

*Application Layer*

Lapisan ini merupakan antarmuka pengguna berupa aplikasi berbasis *website*, *mobile,* atau perangkat pintar yang menampilkan data hasil pemrosesan secara *real-time* dan historis. Melalui lapisan ini, pengguna dapat berinteraksi secara langsung dengan sistem.

Karakteristik kunci dari implementasi IoT dalam sistem pemantauan industri meliputi skalabilitas, keandalan, interoperabilitas, dan keamanan (Mirani dkk, 2022). Skalabilitas memungkinkan sistem untuk dikembangkan dengan menambahkan sensor atau *node* baru tanpa perlu merombak keseluruhan infrastruktur yang telah ada, menjadikannya fleksibel untuk kebutuhan jangka panjang. Keandalan sangat penting agar data dari sensor dapat dikirim dan diterima secara konsisten tanpa kehilangan informasi. Interoperabilitas memungkinkan integrasi berbagai perangkat dan protokol komunikasi dari vendor yang berbeda dalam satu sistem yang terkoordinasi. Aspek keamanan menjadi sangat krusial untuk mencegah akses tidak sah dan manipulasi data, yang dicapai melalui mekanisme enkripsi dan autentikasi yang kuat, seperti penerapan JWT dalam proses komunikasi data antara *node* dan *server*. JWT memastikan bahwa hanya perangkat dan pengguna yang sah yang dapat mengakses atau mengirimkan data ke server. Token ini dihasilkan saat autentikasi dan disertakan dalam setiap permintaan API atau publikasi data dari perangkat.

1. *Application Programming Interface* (API)

Application Programming Interface (API) merupakan antarmuka yang menyediakan seperangkat fungsi dan prosedur yang memungkinkan suatu perangkat lunak berinteraksi dengan perangkat lunak lainnya secara terstruktur. API berfungsi sebagai perantara komunikasi antara program aplikasi dengan sistem lain, seperti sistem operasi, sistem manajemen basis data (DBMS), atau layanan berbasis protokol komunikasi tertentu. Implementasi API dilakukan melalui pemanggilan fungsi-fungsi tertentu dalam kode program untuk menjalankan tugas-tugas spesifik. Dalam konteks pengembangan sistem terdistribusi, API memegang peranan penting karena memungkinkan pertukaran data yang efisien dan terstandar melalui protokol komunikasi seperti HTTP, dengan metode umum seperti GET, POST, PUT, dan DELETE.

Dalam tugas akhir ini, API digunakan untuk menjembatani komunikasi antara perangkat IoT dengan platform pemantauan berbasis web dan Android. Melalui API, data dari sensor dapat dikirim ke server secara real-time dan disajikan kepada pengguna dalam bentuk yang informatif dan mudah diakses. API modern terbukti mendukung fleksibilitas serta interoperabilitas sistem berbasis IoT (Kim, Park, & Choi, 2021).

1. Protokol Komunikasi

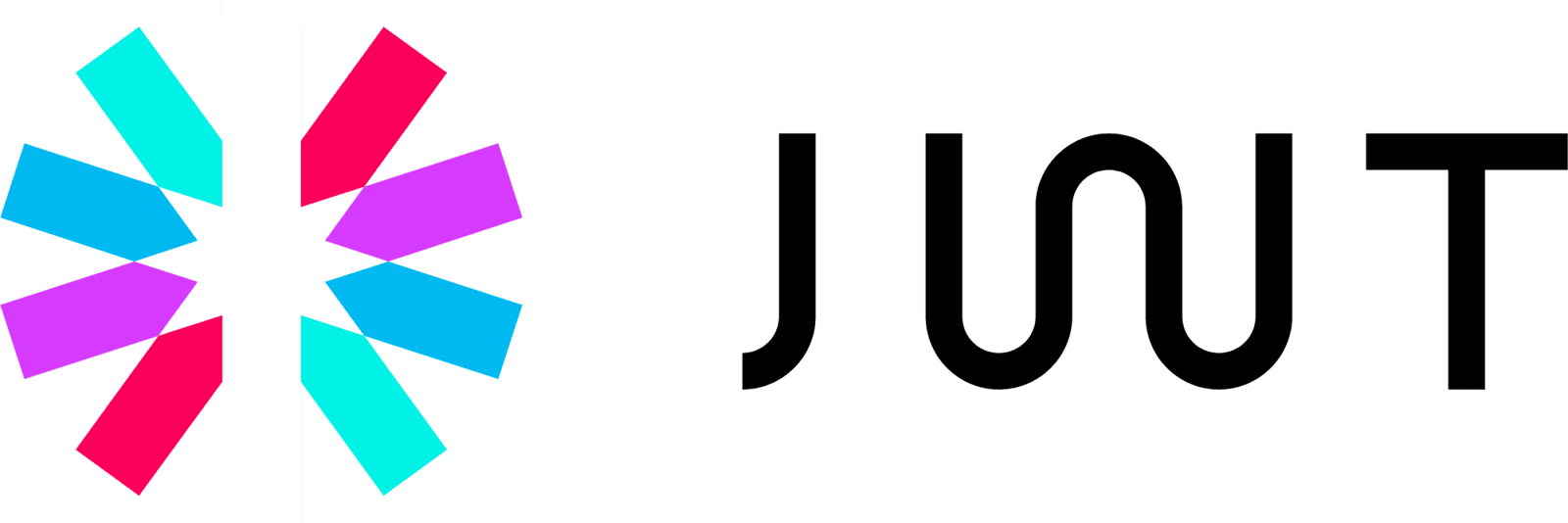
Protokol komunikasi adalah aturan dan standar yang digunakan oleh berbagai macam perangkat pada jaringan lokal untuk menukar informasi lewat internet dengan efisien. Dalam tugas akhir ini, terdapat beberapa protokol yang akan digunakan diantaranya:

*Hypertext Transfer Protocol* (HTTP): protokol komunikasi berbasis teks yang dirancang untuk mentransfer data antara perangkat IoT dan server. HTTP umumnya digunakan untuk mengakses dan mengirimkan data ke server melalui *Representational State Transfer* (REST) API  (Fielding & Taylor, 2002). HTTP bekerja dengan pendekatan request-response, di mana *client* mengirimkan permintaan atau *request* dan *server* merespons dengan informasi yang diminta (Shakdher dkk., 2019).

*Message Queuing Telemetry Transport* (MQTT): Protokol ini digunakan untuk mengirimkan pesan berbasis *publisher*/*subscriber* (Pub/Sub) antara perangkat IoT dan server. Protokol ini sangat cocok untuk digunakan dalam sistem IoT yang memiliki bandwidth yang terbatas dan koneksi yang tidak stabil (Raman & Martin, 2024).

*Long Range Wide Area Network* (LoRaWAN): Protokol ini merupakan protokol khusus yang digunakan untuk komunikasi nirkabel menggunakan modul LoRa. Modul ini mendukung konektivitas jarak jauh dan konsumsi daya rendah pada perangkat IoT (Jáquez dkk., 2023). LoRaWAN menggunakan teknologi modulasi LoRa, yang memungkinkan perangkat untuk mentransmisikan data hingga jarak 10 hingga 15 kilometer di area pedesaan dan 2 hingga 5 kilometer di area perkotaan (Centenaro, dkk., 2016).

1. JSON *Web Token* (JWT)



1. Logo JWT

(Sumber: <https://seeklogo.com/vector-logo/428033/jwt>)

JSON *Web Token* (JWT) merupakan standar terbuka yang digunakan untuk pertukaran informasi secara aman dalam format objek JSON, terutama dalam konteks autentikasi dan otorisasi pada aplikasi berbasis *web*. JWT bersifat ringan, terstruktur, dan mendukung komunikasi tanpa penyimpanan status sesi (*stateless*) di sisi server, sehingga efisien dalam sistem terdistribusi. Token JWT terdiri atas tiga komponen utama, yaitu *header*, *payload*, dan *signature*, yang masing-masing di-*encode* menggunakan format Base64 dan berfungsi memastikan keabsahan serta integritas data yang dikirimkan (Jwt.io, 2024; Darmawan dkk., 2023). Dalam tugas akhir ini, JWT diterapkan untuk mengamankan akses antara pengguna maupun perangkat *Internet of Things* (IoT) terhadap sistem, sehingga hanya entitas yang telah terverifikasi yang dapat mengakses data atau layanan tertentu, serta mengurangi beban penyimpanan di sisi server (Dahiya & Malik, 2020).

1. *Hypertext Transfer Protocol Secure* (HTTPS)

HTTPS adalah versi aman dari protokol HTTP yang digunakan untuk mengirimkan data antara *client* dan *server* melalui internet. HTTPS mengintegrasikan lapisan keamanan *Transport Layer Security* (TLS) atau *Secure Sockets Layer* (SSL) untuk memberikan tiga elemen utama keamanan, yaitu enkripsi data, integritas, dan autentikasi. Enkripsi memastikan bahwa data yang dikirimkan tidak dapat diakses oleh pihak yang tidak berwenang selama dalam perjalanan. Integritas menjamin bahwa data tidak diubah selama transmisi, sementara autentikasi memungkinkan verifikasi identitas *server* sehingga *client* dapat memastikan bahwa mereka terhubung ke sumber yang sah (Hanif, 2024).

1. Git



1. Logo Git

(Sumber: <https://www.pngegg.com/en/png-etzdt>)

Git adalah sebuah perangkat lunak yang berfungsi untuk melacak perubahan dari suatu proyek. Git dirancang untuk mendukung kolaborasi tim dalam mengembangkan proyek secara bersamaan tanpa konflik menggunakan GitHub, yang merupakan layanan Git berbasis *cloud* (Loeliger & McCullough, 2012).

1. *Javascript*

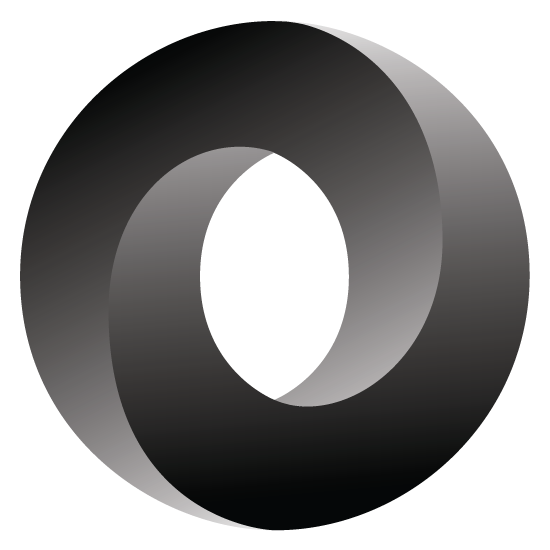


1. Logo Javascript

(Sumber: <https://www.pngegg.com/en/png-hjxmr>)

JavaScript adalah bahasa pemrograman yang awalnya dirancang untuk membuat halaman web menjadi lebih interaktif. Namun, dengan perkembangan teknologi, JavaScript kini digunakan untuk berbagai aplikasi, termasuk pengembangan backend, aplikasi *desktop*, hingga aplikasi *mobile*. JavaScript memiliki keunggulan berupa sintaks yang sederhana, fleksibilitas tinggi, dan dukungan komunitas yang luas. Ekosistem JavaScript mencakup pustaka dan framework seperti React JS, Angular, dan Vue JS untuk pengembangan *front-end*, Node JS untuk pengembangan *backend,* serta React Native, serta Next.js untuk pengembangan web modern dengan fitur server-side rendering dan static site generation(Vercel, 2023). Fitur seperti *event-driven programming* dan asinkronisasi membuat JavaScript sangat efisien untuk menangani operasi *real-time*, seperti *streaming* data atau aplikasi berbasis chat. Bahasa ini juga kompatibel dengan berbagai *browser* dan perangkat, menjadikannya salah satu teknologi paling relevan dalam pengembangan aplikasi modern (Flanagan, 2011).

1. *JavaScript* *Object Notation* (JSON)



1. Logo JSON

(Sumber: <https://www.pngegg.com/en/png-wtyje>)

JSON adalah format data ringan yang digunakan untuk pertukaran data antara server dan aplikasi berbasis web. JSON menggunakan struktur berbasis objek dengan pasangan *key-value*, membuatnya mudah dibaca oleh manusia dan diproses oleh mesin. JSON banyak digunakan dalam aplikasi web karena kompatibilitasnya yang luas dengan berbagai bahasa pemrograman dan efisiensinya dalam serialisasi data (json.org, 2006).

Dalam konteks pengembangan aplikasi berbasis Internet of Things (IoT), JSON menjadi format utama dalam proses serialisasi dan deserialisasi data sensor yang dikirimkan dari perangkat ke server. Format ini digunakan baik dalam komunikasi API maupun dalam penyusunan payload pada JSON Web Token (JWT). JSON mendukung efisiensi dan kecepatan pertukaran data, yang sangat penting untuk sistem pemantauan real-time seperti sistem pemantauan limbah cair.

Selain itu, JSON mendukung interoperabilitas sistem karena mampu berfungsi lintas platform dan lintas bahasa pemrograman tanpa membutuhkan protokol tambahan yang kompleks. Dalam tugas akhir ini, JSON memfasilitasi integrasi antara mikrokontroler ESP32 sebagai pengirim data dan sistem backend berbasis web yang dibangun menggunakan framework Next.js. Dengan demikian, JSON berperan penting dalam menjaga kesederhanaan, fleksibilitas, dan skalabilitas arsitektur sistem secara keseluruhan.

JSON memiliki kinerja parsing yang cepat dan menghasilkan ukuran data yang relatif kecil dibandingkan dengan format lain seperti XML. Hal ini menjadikannya pilihan yang tepat untuk sistem dengan keterbatasan bandwidth dan kebutuhan pemrosesan data secara real-time, seperti sistem pemantauan limbah cair berbasis IoT (Nurseitov et al., 2021).

1. Next.js



1. Logo Next.js

(Sumber: <https://marketplace.visualstudio.com/items?itemName=marcbruederlin.next-icons>)

Next.js adalah framework open-source berbasis React yang dirancang untuk membangun aplikasi web modern dengan dukungan *server-side rendering* dan *static site generation* (Vercel, 2023). Next.js menyediakan serangkaian fitur seperti sistem routing otomatis, optimasi performa, pemisahan kode, serta dukungan API bawaan yang memudahkan pengembangan front-end dan back-end dalam satu proyek. Fitur unggulannya mencakup rendering hybrid, optimasi SEO, serta integrasi dengan TypeScript dan pengelolaan gambar otomatis. Selain itu, proses pengembangan dapat dijalankan secara lokal maupun melalui platform Vercel untuk *deployment* instan dengan konfigurasi minimal (nextjs.org, 2024).

Dalam tugas akhir ini, Next.js digunakan sebagai fondasi pengembangan antarmuka web untuk sistem pemantauan. Framework ini dipilih karena kemampuannya dalam membangun aplikasi yang ringan, responsif, dan optimal untuk kebutuhan pemantauan data secara real-time. Dengan dukungan rendering sisi server dan manajemen komponen yang efisien, Next.js dapat meningkatkan kecepatan akses serta pengalaman pengguna dalam mengakses dashboard sistem pemantauan (Brahmbhatt & Vora, 2022).

1. *Database*

Database atau basis data adalah kumpulan data yang disimpan secara sistematis dan dapat diakses serta dikelola secara elektronik melalui sistem komputer. Tujuan utama dari basis data adalah untuk menyimpan informasi secara terorganisir agar dapat dengan mudah dicari, ditambahkan, diubah, atau dihapus. Terdapat berbagai jenis model database, namun yang paling umum digunakan adalah model relasional. Dalam model ini, data disimpan dalam bentuk tabel (relasi) yang terdiri dari baris dan kolom. Masing-masing baris mewakili suatu entitas data, sementara kolom mewakili atribut dari entitas tersebut (AWS, 2023).

Structured Query Language (SQL) merupakan bahasa pemrograman standar yang digunakan untuk mengelola database relasional. SQL memungkinkan pengguna untuk melakukan berbagai operasi seperti pencarian, penyisipan, pembaruan, dan penghapusan data secara efisien (Gillis, 2024). Selain itu, SQL juga mendukung pengelompokan data, pengurutan, dan penggabungan antar tabel (IBM, 2024). Sejak pertama kali dikembangkan oleh IBM pada awal 1970-an, SQL telah menjadi standar resmi dari American National Standards Institute (ANSI) pada tahun 1986 dan International Organization for Standardization (ISO) pada tahun 1987 (W3Schools, 2024). Keberadaan standar ini menjadikan SQL sebagai bahasa yang seragam dan interoperable dalam berbagai sistem manajemen basis data (DBMS), seperti MySQL, PostgreSQL, Microsoft SQL Server, dan lainnya.

1. MySQL



1. Logo MySQL

(Sumber: <https://www.pngegg.com/en/png-wpzzg>)

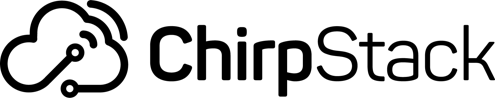
MySQL adalah salah satu contoh sistem pengelolaan basis data yang dirancang untuk menyimpan dan mengelola data dengan menggunakan *Structured Query Language* (SQL). MySQL banyak digunakan dalam pengembangan aplikasi web dan sistem informasi karena kinerjanya yang cepat, stabilitas tinggi, dan kemudahan integrasi dengan berbagai bahasa pemrograman. Dalam pengembangan aplikasi, MySQL sering digunakan untuk menyimpan data pengguna, *log* aktivitas, dan informasi transaksi secara efisien (Ariata, 2023).

Sebagai sistem manajemen basis data relasional yang bersifat open-source, MySQL menyediakan fitur yang mendukung kebutuhan sistem skala kecil hingga besar. Fitur-fitur tersebut meliputi dukungan transaksi dengan prinsip ACID (*Atomicity*, *Consistency*, *Isolation*, *Durability*), *indexing* untuk mempercepat pencarian data, serta replikasi dan clustering untuk menjaga ketersediaan dan keandalan data. MySQL juga kompatibel dengan berbagai sistem operasi dan memiliki komunitas pengguna serta dokumentasi yang sangat luas (Sarwar & Khan, 2021).

1. *Message Broker*

Message broker adalah komponen perangkat lunak yang berfungsi sebagai perantara komunikasi data antar sistem atau perangkat untuk bertukar pesan secara asinkron, tanpa harus terhubung secara langsung satu sama lain secara *real-time*. *Message broker* bekerja dengan cara menerima pesan dari pengirim (*publisher*), kemudian menyimpannya sementara, dan mendistribusikannya ke penerima (*subscriber*) yang terdaftar pada topik atau saluran tertentu. Proses ini dikenal sebagai model *publish-subscribe*. Dengan demikian, sistem menjadi lebih fleksibel karena setiap komponen dapat beroperasi secara independen. Contoh *message broker* yang umum digunakan dalam sistem IoT adalah RabbitMQ, MQTT broker (seperti Mosquitto), dan Apache Kafka.

1. ChirpStack



1. Logo ChirpStack

(Sumber: <https://www.chirpstack.io/>)

ChirpStack adalah *platform open-source* LoRaWAN Network Server yang menyediakan infrastruktur untuk mengelola dan menghubungkan perangkat LoRaWAN ke sistem *backend*. *Platform* ini mencakup komponen utama seperti *application serve*r, *network server*, dan *gateway bridge*, yang memungkinkan pemantauan perangkat, pengelolaan *payload*, dan pengaturan aplikasi IoT (chirpstack.io, 2024).

ChirpStack merupakan platform open-source yang menyediakan infrastruktur lengkap untuk jaringan LoRaWAN. Platform ini terdiri atas komponen-komponen seperti Network Server, Application Server, dan Gateway Bridge, yang memungkinkan integrasi perangkat IoT berbasis LoRa dengan sistem backend. Komponen-komponen ini saling berkoordinasi untuk menerima data dari perangkat, meneruskannya ke sistem aplikasi, serta memungkinkan autentikasi dan manajemen perangkat secara terpusat.

Dalam tugas akhir ini, ChirpStack digunakan untuk mengelola komunikasi perangkat IoT melalui protokol LoRaWAN, khususnya pada area yang memiliki keterbatasan jaringan internet. Penggunaan ChirpStack memungkinkan pengiriman data dari perangkat ke server secara efisien, terenskripsi, dan dapat dimonitor melalui antarmuka web yang telah disediakan.

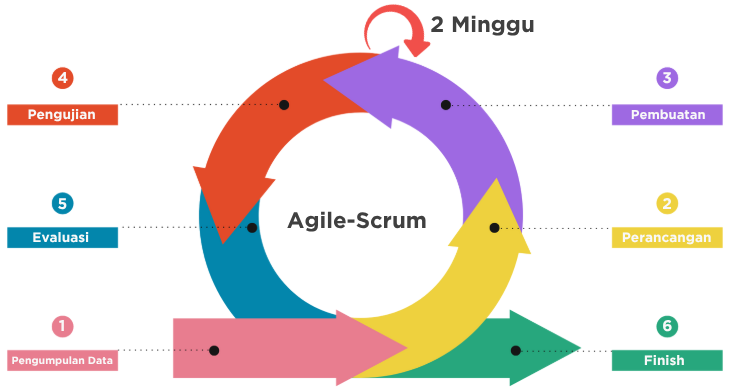
1. *Balck Box Testing*

*Black Box Testing* adalah metode pengujian perangkat lunak yang berfokus pada evaluasi fungsionalitas aplikasi tanpa melihat struktur internal atau implementasi kode. Pengujian ini dilakukan dengan memberikan input tertentu dan memeriksa apakah output yang dihasilkan sesuai dengan spesifikasi yang diharapkan. Pengujian ini dilakukan di akhir pembuatan perangkat lunak untuk mengetahui apakah perangkat lunak dapat berfungsi dengan baik. Pendekatan ini sangat efektif untuk mengidentifikasi bug terkait fungsionalitas, kompatibilitas, atau kinerja dari perspektif pengguna, memastikan bahwa aplikasi memenuhi kebutuhan bisnis dan berfungsi sebagaimana mestinya (Myers dkk., 2011).

BAB III

METODE PERANCANGAN SISTEM

Metode penelitian yang digunakan dalam penyusunan tugas akhir “Sistem Pemantauan Limbah Cair Industri Terintegrasi IoT dan JWT Berbasis *Website* dan Android” yaitu menggunakan metode Agile-Scrum. Agile adalah sebuah metode pengembangan perangkat lunak yang berbasis padapengembangan iteratif, di mana persyaratan dan solusi berkembang melalui kolaborasi antar tim yang terorganisir. Salah satu jenis metode Agile yang paling populer adalah Scrum. Scrum mengorganisir pengembangan software ke dalam siklus waktu terbatas yang disebut sprint. Dalam setiap sprint, tim melakukan perencanaan, analisis, desain, implementasi, pengujian, dan peninjauan (biznetgio.com, 2024). Oleh karena itu, tahap-tahapan metode penelitian dalam tugas akhir ini adalah sebagai berikut:

**

1. Metode Agile-Scrum

(Sumber: Dokumentasi Pribadi)

1. Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data sebelum tugas akhir dilaksanakan, yaitu dengan mencari literasi studi pustaka, serta mengidentifikasi kebutuhan. Studi Pustaka dilaksanakan untuk memahami teori penunjang sistem yang akan digunakan pada tahap perancangan dan pembuatan (biznetgio.com, 2024). Literasi dalam penelitian ini adalah yang berkaitan dengan tugas akhir. Studi dilakukan secara sistematis dengan menelusuri berbagai sumber seperti tugas akhir tahun-tahun sebelumnya, dan referensi digital dari jurnal ilmiah nasional maupun internasional yang relevan pada website Google Scholar, IEEE Xplore, serta MDPI. Proses pencarian difokuskan pada artikel yang dipublikasikan dalam rentang waktu 2020 hingga 2024 dengan menggunakan beberapa kata kunci seperti “IoT Platform”, “Wastewater Monitoring”, “REST API”, “MQTT”, “LORAWAN”, dan “JSON Web Token”. Dari hasil pencarian, terkumpul sebanyak 22 artikel maupun jurnal ilmiah yang relevan. Selanjutnya penulis melakukan proses seleksi sesuai dengan topik dan ruang lingkup tugas akhir melalui abstrak karya ilmiah, didapatkanlah sebanyak 7 karya ilmiah yang dijadikan referensi utama. Masing-masing karya ilmiah yang terpilih memiliki metode yang berbeda dalam membuat sistem pemantauan limbah cair industri yang efektif. Namun, masih terdapat kekurangan dalam hal skalabilitas, keandalan, interoperabilitas, atau keamanan.

Selanjutnya dilaksanakan identifikasi kebutuhan untuk memastikan tahapan pembuatan dapat berjalan dengan lancar. Hasil identifikasi yaitu berupa kebutuhan perangkat keras dan perangkat lunak yang akan digunakan. Perangkat keras dapat mendukung dalam pembuatan sistem, sedangkan perangkat lunak dapat membantu dalam membuat program atau aplikasi. Berikut adalah hasil identifikasi kebutuhan:

1. Perangkat Keras yang Digunakan (*Hardware*)

*Hardware* yang diperlukan ialah seperangkat alat atau elemen elektronik yang dapat membantu atau mendukung dalam kinerja aplikasi ini, sehingga aplikasi yang diusulkan dapat bekerja dengan baik. *Hardware* minimal yang digunakan oleh *server* adalah sebagai berikut:

* 1. Komputer dengan *processor 1 Core*.
  2. Memori internal berukuran 1 GB.
  3. Penyimpanan internal berukuran 60 GB.
  4. Koneksi internet.

Adapun *Hardware* minimal yang digunakan oleh *client* (pengguna) adalah sebagai berikut:

* 1. Komputer dengan sistem operasi Windows, Linux, atau Mac.
  2. *Smartphone* dengan sistem operasi Android, atau IoS*.*
  3. Mempunyai koneksi jaringan seluler, atau Wi-Fi.

1. Perangkat Lunak yang Digunakan (*Software*)

Software yang diperlukan ialah yang dapat mendukung dari sistem operasi dan aplikasi database, adapun software pengembang yang digunakan adalah sebagai berikut:

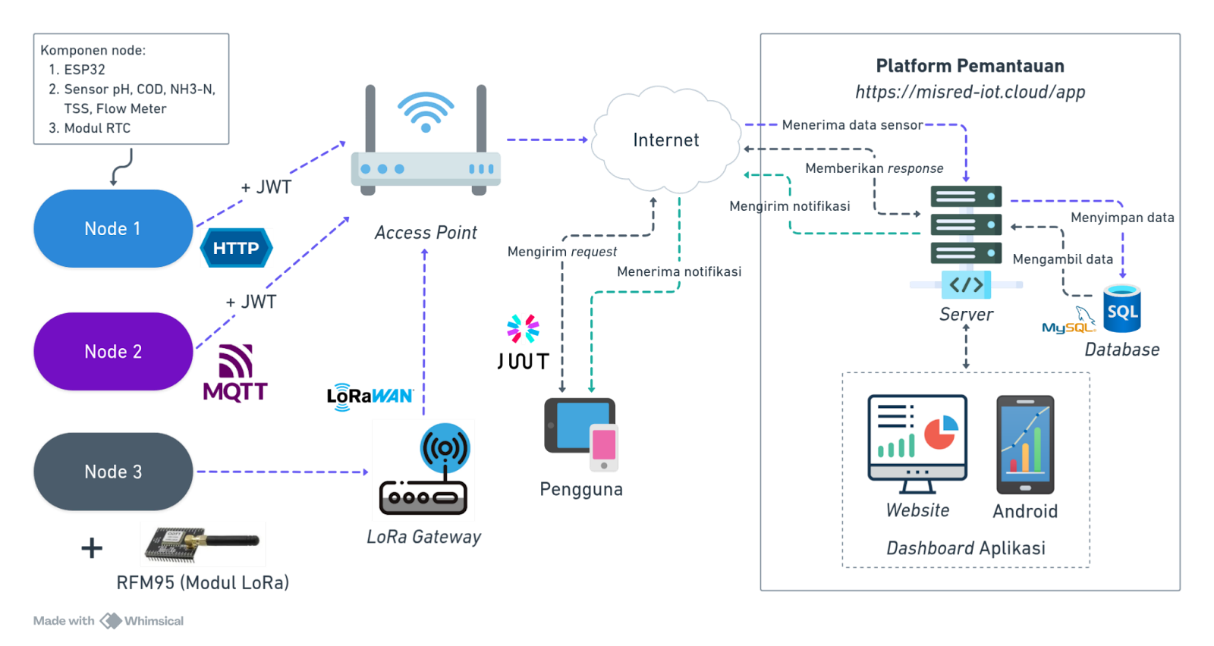
* 1. Visual Studio Code.
  2. Android Studio.
  3. Browser.
  4. Figma.

1. Perancangan

Pada tahap ini dilakukan pembuatan rancangan kerja untuk sistem pemantauan. Tahap ini meliputi pembuatan rancangan arsitektur dari sistem, rancangan diagram *Use Case*, rancangan ERD, dan rancangan tampilan *User Interface* (UI) dari aplikasi.

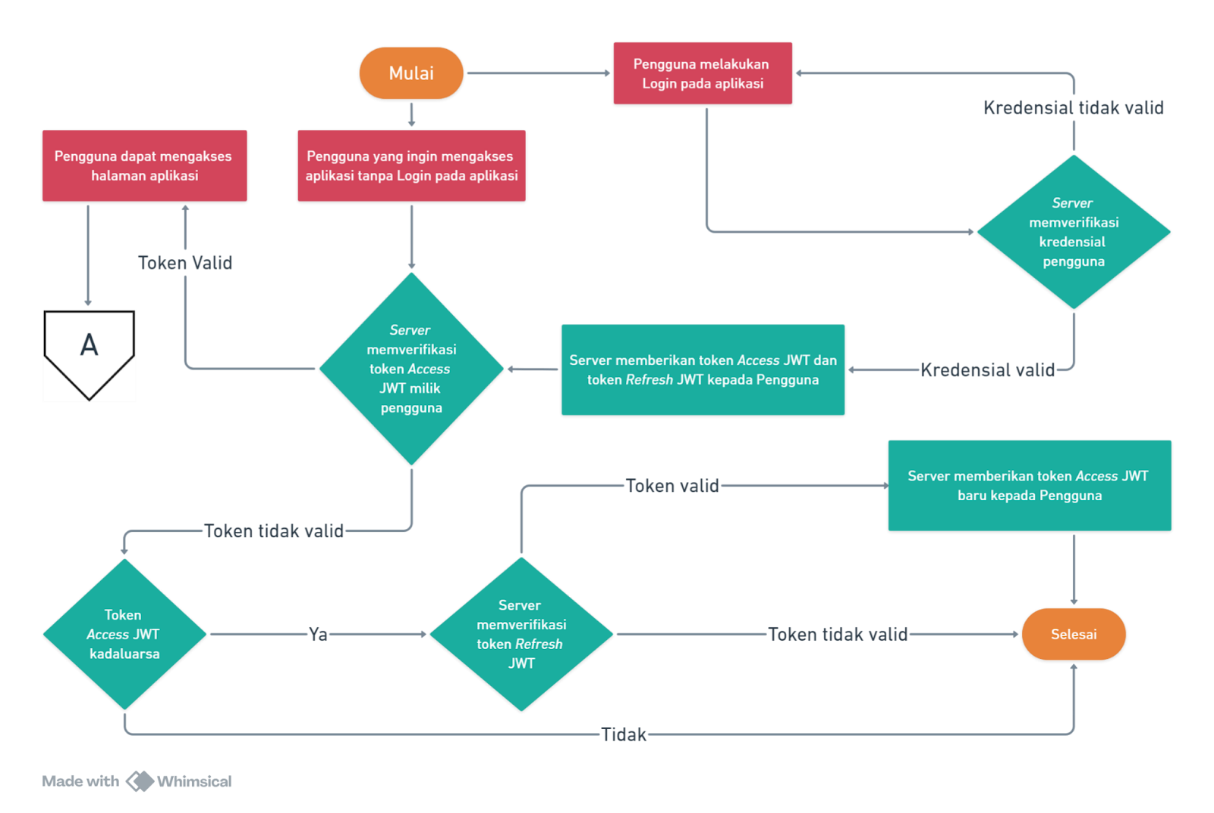
1. Arsitektur Sistem

Rancangan dari arsitektur sistem terdiri dari diagram blok sistem secara keseluruhan, dan diagram blok sistem pada tugas akhir. Pada tugas akhir ini, terdapat tiga buah *node* dengan tiga jenis protokol yang akan digunakan untuk berkomunikasi dengan *server. Node* merupakan perangkat IoT yang bertugas mengumpulkan data. *Node* terdiri dari mikrokontroler ESP32 dan beberapa sensor, yaitu pH, COD, kekeruhan, NH3-N, debit air, serta modul *Real-Time Clock* (RTC). *Node* pertama akan menggunakan protokol HTTP dan *node* kedua akan menggunakan protokol MQTT. *Node* ketiga akan menggunakan protokol LoRaWAN, dengan konektivitas menggunakan modul LoRa (RFM95) yang terhubung ke sebuah LoRa *Gateway*. Pengiriman data yang dilakukan oleh pengguna dan node akan di-*hashing* dengan JWT. *Platform* pemantauan yang berfungsi sebagai *server*,akan menerima data sensor yang dikirim oleh *node*, memvalidasi token JWT milik *node*, dan menyimpannya pada basis data MySQL. *Platform* pemantauan memiliki dua antarmuka yang dapat diakses oleh pengguna, yaitu *website* dan Android. *Platform* berfungsi untuk menampilkan visualisasi data dari sensor untuk dipantau secara *real-time*. Gambaran umum dari arsitektur sistem secara keseluruhan dan *flowchart* dapat dilihat pada Gambar 3.2 dan 3.3.



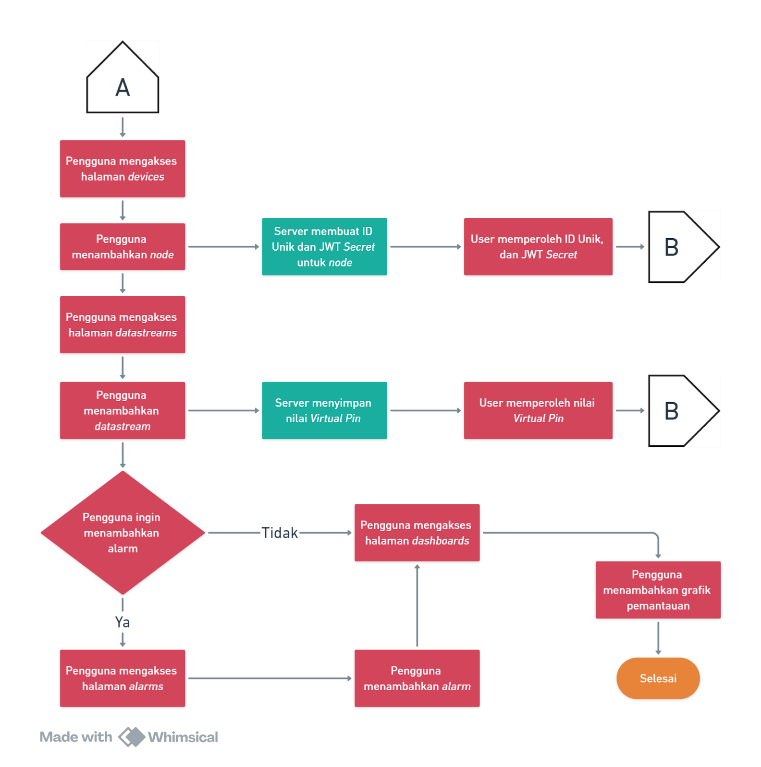
1. Arsitektur Sistem Secara Keseluruhan

(Sumber: Dokumentasi Pribadi)



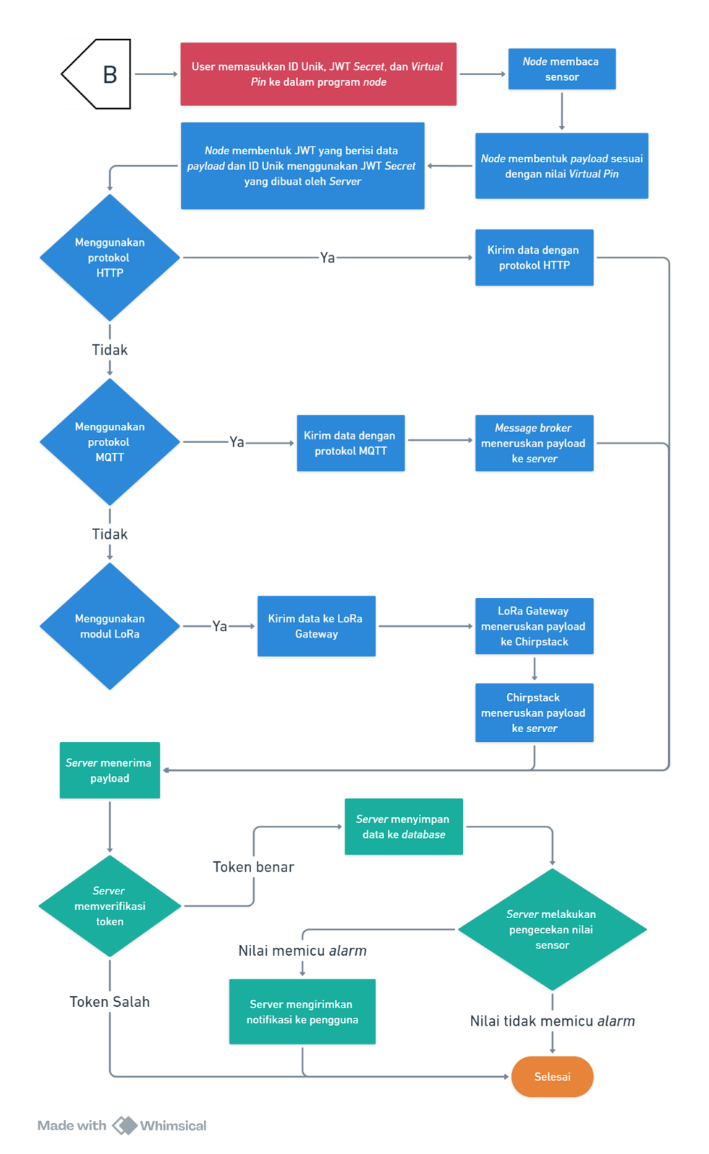
1. Flowchart Sistem Bagian Otorisasi Dan Autentikasi

(Sumber: Dokumentasi Pribadi)



1. Flowchart Sistem Bagian User

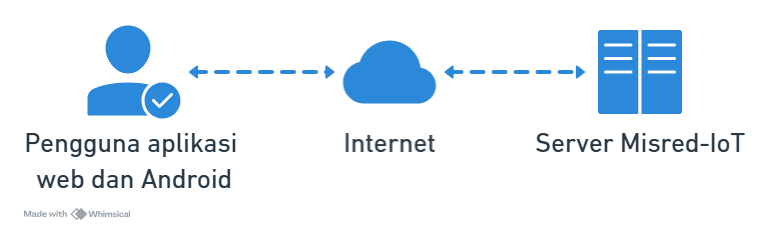
(Sumber: Dokumentasi pribadi)



1. Flowchart Sistem Bagian Node

(Sumber: Dokumentasi pribadi)

Tugas akhir ini akan menghasilkan sebuah aplikasi web dan Android. Aplikasi akan berinteraksi dengan sistem melalui *server* ketika terhubung dengan internet. Gambar 9.3 menunjukkan arsitektur dari sistem yang akan dihasilkan pada tugas akhir ini. *Server* yang digunakan berada pada *domain https://misred-iot.com*. Aplikasi web dan Android akan berkomunikasi dengan *server* menggunakan protokol HTTP melalui REST API. REST API akan digunakan untuk proses pemantauan parameter limbah cair, dan proses autentikasi serta otorisasi pengguna aplikasi.

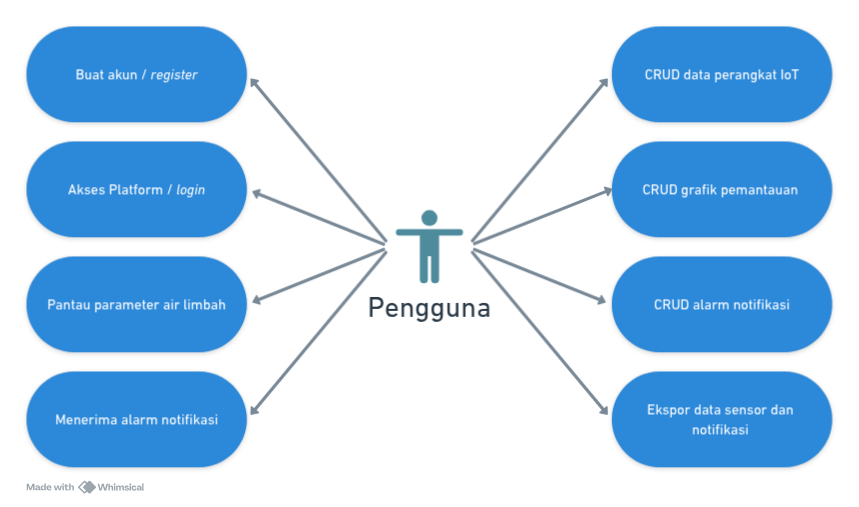


1. Arsitektur Sistem pada Tugas Akhir

(Sumber: Dokumentasi Pribadi)

1. Diagram *Use Case*

Berdasarkan arsitektur sistem dari tugas akhir, dibuatlah diagram *use case* untuk desain skenario awal dari pengguna. Pengguna dapat melakukan *register* atau *login* pada platform pemantauan, melakukan proses *Create, Read, Update, Delete* (CRUD) pada perangkat IoT dan grafik pemantauan parameter air limbah, serta mengatur alarm notifikasi untuk parameter tertentu. Gambar 9.4 menunjukkan tentang diagram ini.

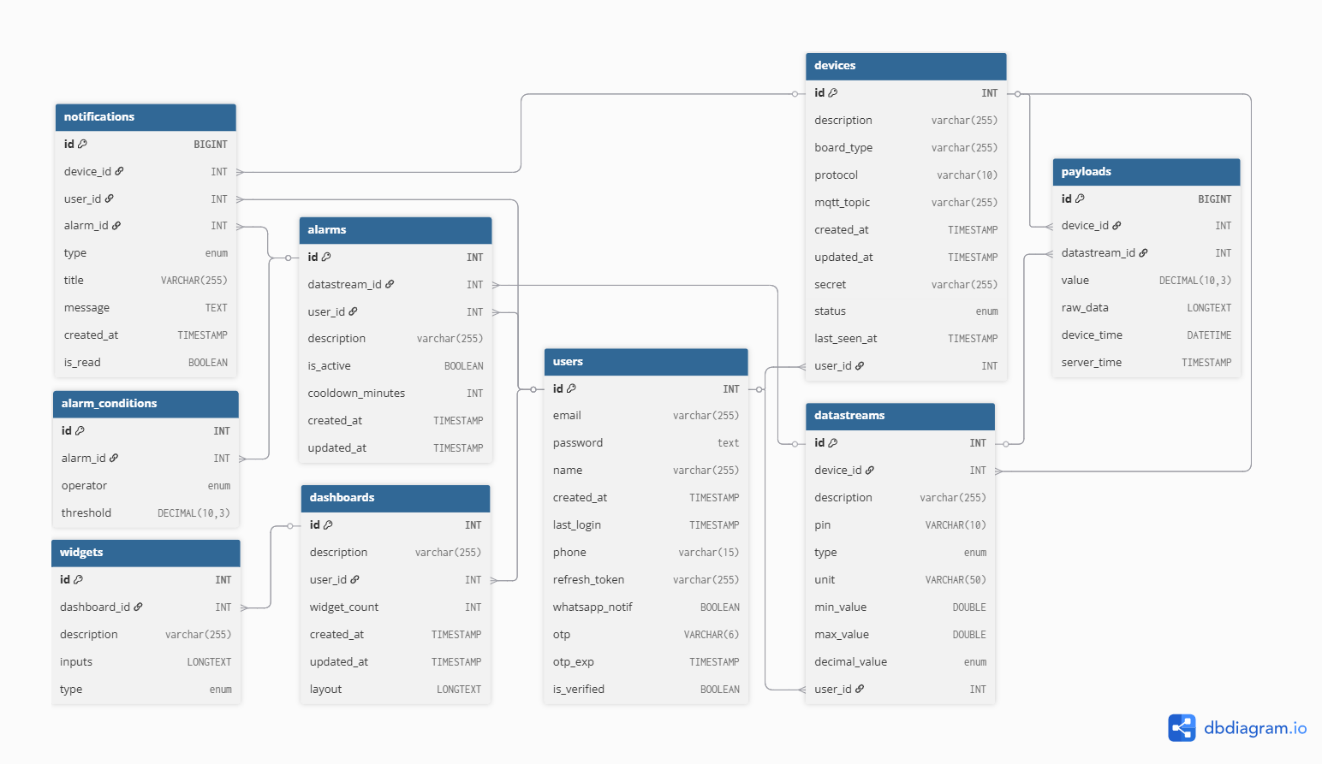


1. *Use Case Diagram*

(Sumber: Dokumentasi Pribadi)

1. *Entity Relationship Diagram* (ERD)

Pada tugas akhir ini, digunakan sembilan tabel basis data. Tabel *users* digunakan untuk menyimpan kredensial pengguna aplikasi, OTP, dan token JWT. Tabel *devices* digunakan untuk menyimpan data perangkat (*node*) yang terdaftar pada platform, termasuk jenis board, dan protokol komunikasi. Tabel *widgets* digunakan untuk menyimpan informasi visualisasi data dari perangkat dalam bentuk grafik. Tabel *payloads* digunakan untuk menyimpan data sensor yang dikirimkan oleh perangkat beserta waktu pengirimannya. Tabel *alarms* digunakan untuk menyimpan data alarm yang dibuat oleh pengguna. Relasi antar tabel pada Gambar 9.5 menunjukkan bahwa setiap *node* dapat memiliki satu atau banyak data sensor, grafik, dan alarm notifikasi.



1. *Entity Relationship Diagram*

(Sumber: Dokumentasi Pribadi)

1. Tampilan UI pada Aplikasi

Rancangan *User Interface* atau tampilan antarmuka dari aplikasi *website* dan Androiddibuat menggunakan *software* Figma. Tabel 3.1 menunjukkan tampilan antarmuka pada aplikasi *website*, sedangkan Tabel 3.2 menunjukkan tampilan antarmuka pada aplikasi Android.

1. Rancangan tampilan antarmuka aplikasi *website*

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Halaman** | **Gambar** |
| 1. | Login |  |
| 2. | Dashboard |  |
| 3. | Devices |  |
| 4. | Datastreams |  |
| 5. | Alarms |  |

1. Rancangan tampilan antarmuka aplikasi Android

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Halaman** | **Gambar** |
| 1. | Login |  |
| 2. | Dashboard |  |
| 3. | Devices |  |
| 4. | Datastreams |  |
| 5. | Alarms |  |

1. Pembuatan

Pada tahap ini dilakukan pengerjaan aplikasi web dan Android dengan *software* Visual Studio Code. Tahap ini meliputi berbagai langkah untuk membangun sistem pemantauan limbah cair industri berbasis web dan Android. Langkah-langkah tersebut adalah:

1. Membuat tabel data dalam basis data pada *server* sesuai rancangan basis data pada Gambar 3.5.
2. Membuat kode untuk membangun API antara aplikasi web dan Android dengan basis data *server* dan *web server.* Pembuatan kode dilakukan menggunakan *framework* Node JS.
3. Membuat kode untuk menjalankan tampilan aplikasi sesuai dengan rancangan UI yang sudah dibuat. Pembuatan kode dilakukan menggunakan *framework* Next JS dan bahasa pemrograman CSS.
4. Pengujian

Pada tahap ini dilakukan pengujian sistem yang telah dibuat. Dalam tugas akhir ini terdapat lima pengujian, yaitu:

1. *Black Box Testing* (Pengujian fungsionalitas)

Pengujian yang dilakukan pertama kali adalah *Black Box Testing*. Pengujian ini dilakukan untuk menguji apakah fungsionalitas aplikasi sudah benar dan sesuai dengan rancangan. Tabel 3.3 menunjukkan rancangan tabel pengujian fungsionalitas untuk aplikasi *website* dan Android.

1. Rancangan pengujian fungsionalitas aplikasi *website* dan Android

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Deskripsi Pengujian** | **Parameter yang Diharapkan** | **Tampilan yang dihasilkan** |
| 1. | Melakukan *login* dengan data *username* dan *password* benar. | Pengguna berhasil *login*. |  |
| 2. | Melakukan *login* dengan data *username* dan *password* salah. | Pengguna gagal *login.* |  |
| 3. | Mengakses halaman dashboard. | Pengguna akan diarahkan menuju halaman *dashboard* yang berisi widget dari tiap parameter lengkap dengan waktu terakhirnya*.* |  |
| 4. | Mengakses halaman devices | Pengguna dapat menampilkan daftar perangkat IoT. |  |
| 5. | Menambahkan perangkat IoT | Pengguna dapat menambahkan perangkat IoT pada halaman *devices* |  |
| 6. | Mengubah data perangkat IoT | Pengguna dapat mengubah data perangkat IoT pada halaman *devices* |  |
| 7. | Menghapus perangkat IoT | Pengguna dapat menghapus perangkat IoT pada halaman *devices* |  |
| 8. | Menambahkan *datastream* | Pengguna dapat menambahkan data *datastream* dari perangkat IoT pada halaman *datasteams* |  |
| 9. | Mengubah *datastream* | Pengguna dapat mengubah data dari *datastream* halaman *datastreams* |  |
| 10. | Menghapus *datastream* | Pengguna dapat menghapus *datastream* pada halaman *datastreams* |  |
| 11. | Menambahkan grafik pemantauan | Pengguna dapat menambahkan grafik pemantauan pada halaman *dashboard.* |  |
| 12. | Mengubah grafik pemantauan | Pengguna dapat mengubah grafik pemantauan pada halaman *dashboard.* |  |
| 13. | Menghapus grafik pemantauan | Pengguna dapat menghapus grafik pemantauan pada halaman *dashboard.* |  |
| 14. | Menambahkan alarm notifikasi | Pengguna dapat menambahkan alarm pada menu halaman *dashboard.* |  |
| 15. | Mengubah alarm notifikasi | Pengguna dapat mengubah alarm pada menu halaman *dashboard.* |  |
| 16. | Menghapus alarm notifikasi | Pengguna dapat menghapus alarm pada menu halaman *dashboard.* |  |

1. Pengujian Sistem Keamanan

Pengujian selanjutnya adalah pengujian sistem keamanan. Pengujian ini dilakukan untuk dapat mengetahui isi dari *payload* yang dikirimkan dan menguji integritas *payload* tersebut. Tabel 3.4 menunjukkan rancangan pengujian ini.

1. Rancangan pengujian sistem keamanan

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Deskripsi Pengujian** | **Gambar** | **Status** |
| 1. | Melakukan pengecekan enkripsi pada *payload* |  |  |
| 2. | Mengakses halaman aplikasi menggunakan token JWT yang valid. |  |  |
| 3. | Mengakses halaman aplikasi menggunakan token JWT yang dimodifikasi. |  |  |
| 4. | Mengakses halaman aplikasi tanpa disertai *refresh token* JWT |  |  |

1. Pengujian Notifikasi

Pengujian selanjutnya adalah pengujian notifikasi. Pengujian ini dilakukan untuk menguji keberhasilan aplikasi dalam memunculkan notifikasi *pop-up* pada browser dan sistem operasi Android. Tabel 9.5 menunjukkan rancangan pengujian ini.

1. Rancangan pengujian notifikasi

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Deskripsi Pengujian** | **Gambar** | **Status** |
| 1. | Nilai datastream memicu alarm *browser* |  |  |
| 2. | Nilai datastream memicu alarm WhatsApp |  |  |

1. Pengujian Performa

Pengujian selanjutnya adalah pengujian performa. Pengujian ini dilakukan untuk dapat mengetahui penggunaan memori, CPU, dan jaringan. Dalam pengujian performa aplikasi *website*, digunakan fitur ‘*Network Monitor*’ pada *browser* Google Chrome. Fitur ini digunakan untuk memperoleh waktu yang dibutuhkan untuk memuat halaman. Waktu yang diambil adalah waktu dimuatnya struktur HTML halaman saja (waktu *DOMContentLoaded*), waktu dimuatnya halaman tanpa ikon (waktu *load*), dan waktu dimuatnya keseluruhan halaman (waktu *finish*). Tabel 9.6 menunjukkan rancangan pengujian ini.

1. Rancangan pengujian performa aplikasi *website*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Halaman** | **Waktu memuat (ms)** | | |
| ***DOMContentLoaded*** | ***Load*** | ***Finish*** |
| 1. | Login |  |  |  |
| 2. | Dashboard |  |  |  |
| 3. | Devices |  |  |  |
| 4. | Datastreams |  |  |  |
| 5. | Alarms |  |  |  |

Sementara pada pengujian performa aplikasi Android, digunakan fitur ‘Android Profiler’ pada *software* Android Studio. Fitur ini dapat menyediakan informasi mengenai bagaimana kondisi CPU, memori, dan jaringan saat aplikasi dijalankan.

1. Evaluasi

Pada tahap ini dilakukan proses *debugging* dan perilisan*. Debugging* dilakukan untuk mengatasi ketidakselarasan kode. Ketidakselarasan kode ini biasanya sebuah galat yang tidak terdeteksi saat pembuatan hingga proses *compiling* dan hanya akan terlihat setelah program dijalankan. Selain itu, debugging juga digunakan sebagai peninjau apakah aplikasi sudah sesuai dan dapat dengan mudah dipahami oleh pengguna. Untuk aplikasi web, akan dirilis pada *server* Misred-IoT dengan alamat https://misred-iot.cloud/app. Adapun aplikasi Android akan dirilis secara lokal dan dapat dipasang pada perangkat Android melalui implementasi Android WebView.

BAB IV

HASIL PENGUJIAN DAN ANALISIS

1. Hasil Pengujian Fungsionalitas Aplikasi (*Black Box Testing*)

Hasil dari pengujian fungsionalitas aplikasi atau pengujian *balck box* adalah berupa sukses atau gagalnya aplikasi berjalan sesuai scenario. Pada pengujian ini hanya berfokus pada fungsionalitas dan keluaran aplikasi. Hasil pengujian fungsionalitas dapat dilihat pada Tabel 4.1 dan Tabel 4.2 berikut.

1. Hasil pengujian fungsionalitas aplikasi *web*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Deskripsi Pengujian** | **Parameter yang Diharapkan** | **Tampilan yang dihasilkan** |
| 1. | Melakukan *login* dengan data *username* dan *password* benar | Pengguna berhasil *login* | Login berhasil (sesuai) |
| 2. | Melakukan *login* dengan data *username* dan *password* salah | Pengguna gagal *login* | Gagal login (sesuai) |
| 3. | Mengakses halaman *dashboard* | Pengguna akan diarahkan menuju halaman *dashboard* yang berisi widget dari tiap parameter lengkap dengan waktu terakhirnya | Grafik dengan filter 1 hari terakhir (sesuai) |
| 4. | Mengakses halaman *devices* | Pengguna dapat menampilkan daftar perangkat IoT | Menampilkan daftar perangkat (sesuai) |
| 5. | Menambahkan perangkat IoT | Pengguna dapat menambahkan perangkat IoT pada halaman *devices* | Menambahkan perangkat (sesuai) |
| 6. | Mengubah data perangkat IoT | Pengguna dapat mengubah data perangkat IoT pada halaman *devices* | Mengubah data perangkat (sesuai) |
| 7. | Menghapus perangkat IoT | Pengguna dapat menghapus perangkat IoT pada halaman *devices* | Menghapus perangkat (sesuai) |
| 8. | Mengkases halaman *datastreams* | Pengguna dapat menampilkan daftar *datastream* | Menampilkan daftar *datastream* (sesuai) |
| 9. | Menambahkan *datastream* | Pengguna dapat menambahkan data *datastream* dari perangkat IoT pada halaman *datasteams* | Menambah *datastream* (sesuai) |
| 10. | Mengubah *datastream* | Pengguna dapat mengubah data dari *datastream* halaman *datastreams* | Mengubah *datastream* (sesuai) |
| 11. | Menghapus *datastream* | Pengguna dapat menghapus *datastream* pada halaman *datastreams* | Menghapus *datastream* (sesuai) |
| 12. | Menambahkan grafik pemantauan | Pengguna dapat menambahkan grafik pemantauan pada halaman *dashboard.* | Menambahkan grafik (sesuai) |
| 13. | Mengubah grafik pemantauan | Pengguna dapat mengubah grafik pemantauan pada halaman *dashboard.* | Mengubah grafik (sesuai) |
| 14. | Menghapus grafik pemantauan | Pengguna dapat menghapus grafik pemantauan pada halaman *dashboard.* | Menghapus grafik (sesuai) |
| 15. | Menambahkan alarm notifikasi | Pengguna dapat menambahkan alarm pada menu halaman alarm*.* | Menambahkan alarm (sesuai) |
| 16. | Mengubah alarm notifikasi | Pengguna dapat mengubah alarm pada menu halaman alarm*.* | Mengubah alarm (sesuai) |
| 17. | Menghapus alarm notifikasi | Pengguna dapat menghapus alarm pada menu halaman alarm*.* | Menghapus alarm (sesuai) |

(Sumber: Dokumentasi pribadi)

1. Hasil pengujian fungsionalitas aplikasi android

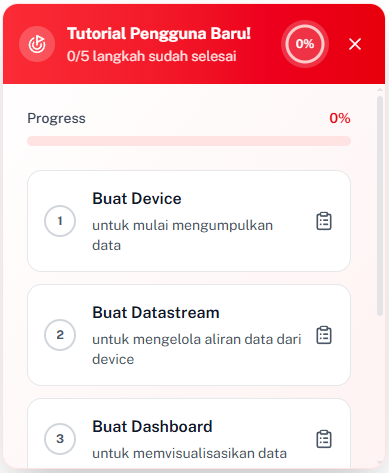
|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Deskripsi Pengujian** | **Parameter yang Diharapkan** | **Tampilan yang dihasilkan** |
| 1. | Melakukan *login* dengan data *username* dan *password* benar. | Pengguna berhasil *login*. | Login berhasil (sesuai) |
| 2. | Melakukan *login* dengan data *username* dan *password* salah. | Pengguna gagal *login.* | Gagal login (sesuai) |
| 3. | Mengakses halaman dashboard. | Pengguna akan diarahkan menuju halaman *dashboard* yang berisi widget dari tiap parameter lengkap dengan waktu terakhirnya*.* | Grafik dengan filter 1 jam terakhir (sesuai) |
| 4. | Mengakses halaman devices | Pengguna dapat menampilkan daftar perangkat IoT. | Menampilkan daftar perangkat (sesuai) |
| 5. | Menambahkan perangkat IoT | Pengguna dapat menambahkan perangkat IoT pada halaman *devices* | Menambahkan perangkat (sesuai) |
| 6. | Mengubah data perangkat IoT | Pengguna dapat mengubah data perangkat IoT pada halaman *devices* | Mengubah data perangkat (sesuai) |
| 7. | Menghapus perangkat IoT | Pengguna dapat menghapus perangkat IoT pada halaman *devices* | Menghapus perangkat (sesuai) |
| 8. | Mengakses halaman *datastreams* | Pengguna dapat menampilkan daftar *datastream* | Menampilkan daftar *datastream* (sesuai) |
| 9. | Menambahkan *datastream* | Pengguna dapat menambahkan data *datastream* dari perangkat IoT pada halaman *datasteams* | Menambah *datastream* (sesuai) |
| 10. | Mengubah *datastream* | Pengguna dapat mengubah data dari *datastream* halaman *datastreams* | Mengubah *datastream* (sesuai) |
| 11. | Menghapus *datastream* | Pengguna dapat menghapus *datastream* pada halaman *datastreams* | Menghapus *datastream* (sesuai) |
| 12. | Menambahkan grafik pemantauan | Pengguna dapat menambahkan grafik pemantauan pada halaman *dashboard.* | Menambahkan grafik (sesuai) |
| 13. | Mengubah grafik pemantauan | Pengguna dapat mengubah grafik pemantauan pada halaman *dashboard.* | Mengubah grafik (sesuai) |
| 14. | Menghapus grafik pemantauan | Pengguna dapat menghapus grafik pemantauan pada halaman *dashboard.* | Menghapus grafik (sesuai) |
| 15. | Menambahkan alarm notifikasi | Pengguna dapat menambahkan alarm pada menu halaman alarm*.* | Menambahkan alarm (sesuai) |
| 16. | Mengubah alarm notifikasi | Pengguna dapat mengubah alarm pada menu halaman alarm*.* | Mengubah alarm (sesuai) |
| 17. | Menghapus alarm notifikasi | Pengguna dapat menghapus alarm pada menu halaman alarm*.* | Menghapus alarm (sesuai) |

(Sumber: Dokumentasi pribadi)

Data hasil pengujian fungsionalitas aplikasi terbagi ke dalam 16 bagian, mencakup platform *website* maupun Android, di antaranya:

* + 1. *Login* Berhasil

Proses *login* dilakukan dengan mengisi alamat email dan kata sandi pengguna, kemudian menekan tombol masuk. Alternatifnya, pengguna juga dapat melakukan login melalui akun Google dengan memilih opsi “Login dengan Gmail”. Apabila proses login berhasil, sistem akan langsung mengarahkan pengguna ke halaman *dashboard*. Pada halaman ini, ditampilkan daftar panduan penggunaan awal yang ditujukan bagi pengguna baru agar dapat memahami dan menggunakan aplikasi secara optimal. Daftar panduan terbaru dapat terlihat seperti Gambar berikut.



1. Daftar tutorial pengguna baru

(Sumber: Dokumentasi pribadi)

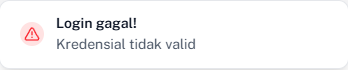
Pengguna baru dapat mengikuti panduan tersebut dengan menekan daftar langkah yang tersedia. Terdapat lima langkah utama dalam panduan ini, yaitu:

|  |  |  |
| --- | --- | --- |
| 1) | Buat *Device* : | Langkah pertama adalah membuat atau menambahkan *device* (perangkat IoT) yang akan digunakan. Langkah ini bertujuan untuk memulai proses pengumpulan data dari perangkat yang terhubung. |
| 2) | Buat *Datastream* : | Langkah kedua adalah membuat atau menambahkan *datastream*, yang digunakan untuk mengelola aliran data dari perangkat IoT yang telah ditambahkan. |
| 3) | Buat *Dashboard* : | Langkah ketiga adalah membuat *dashboard* baru yang memungkinkan pengguna untuk memvisualisasikan data serta menambahkan *widget* sesuai kebutuhan. |
| 4) | Buat *Widget* : | Langkah keempat adalah menambahkan *widget* dengan menekan tombol *edit* pada halaman *dashboard*. |
| 5) | Buat *Alarm* : | Langkah kelima adalah membuat *alarm* yang berfungsi untuk menetapkan ambang batas pada parameter tertentu, sehingga sistem dapat memberikan notifikasi peringatan apabila terjadi pelampauan nilai. |

Setiap langkah yang diselesaikan akan meningkatkan progres pengguna hingga mencapai 100%, yang menandakan bahwa seluruh panduan telah berhasil dilaksanakan.

* + 1. *Login* Gagal

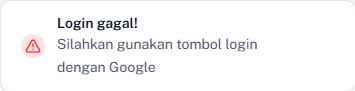
Proses *login* dapat dinyatakan gagal apabila pengguna memasukkan alamat email atau kata sandi yang tidak sesuai. Dalam kondisi tersebut, sistem akan menampilkan indikator berupa pesan peringatan “Login gagal!” disertai dengan keterangan “Kredensial tidak valid”, sebagaimana ditunjukkan pada Gambar berikut.



1. Login gagal

(Sumber: Dokumentasi pribadi)

Indikator ini muncul khususnya ketika pengguna telah mendaftarkan akun secara manual melalui halaman registrasi. Namun, apabila pengguna mencoba *login* menggunakan akun yang didaftarkan melalui integrasi akun Google tanpa menggunakan tombol “Login dengan Gmail”, sistem akan menampilkan indikator “Login gagal!” dengan keterangan “Silakan gunakan tombol login dengan Google”, seperti diperlihatkan pada Gambar berikut.



1. Login gagal menggunakan akun Google

(Sumber: Dokumentasi pribadi)

Selain itu, apabila pengguna menekan tombol *login* tanpa mengisi kolom email maupun kata sandi, sistem akan menampilkan indikator kesalahan pada masing-masing kolom yang belum diisi, sebagaimana ditunjukkan pada Gambar berikut.

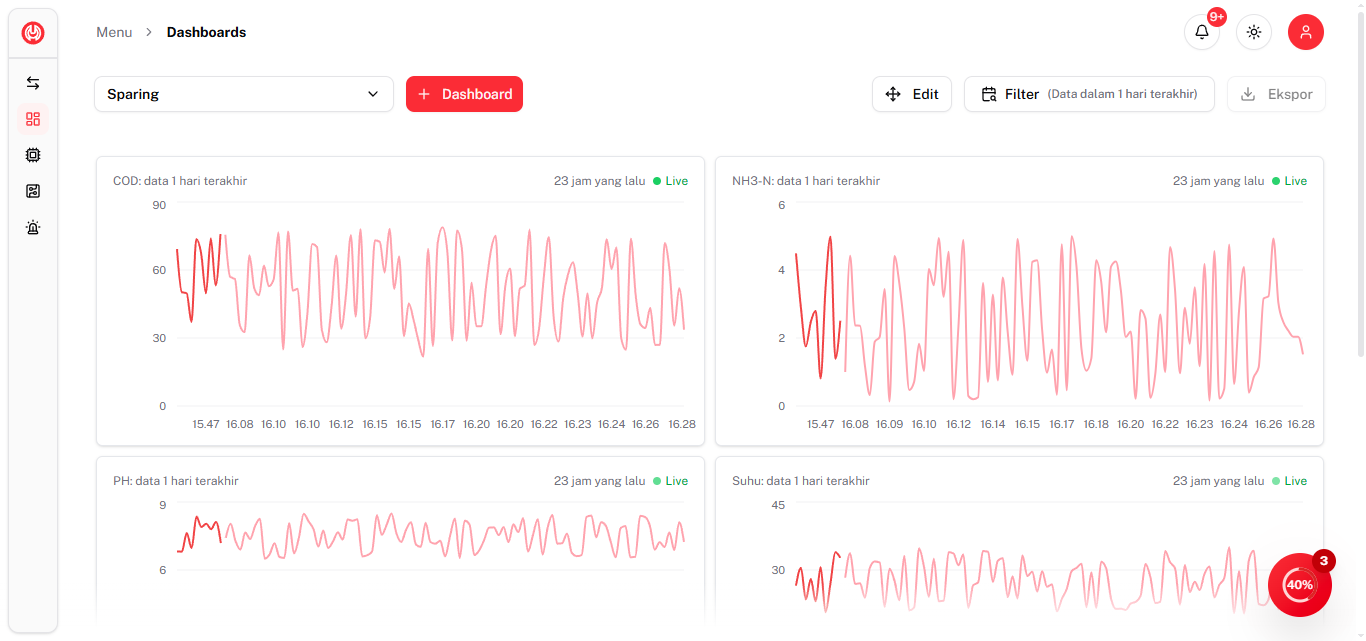


1. Indikator kolom kosong

(Sumber: Dokumentasi pribadi)

* + 1. Mengakses Halaman *Dashboard*

Pada halaman *Dashboard*, pengguna dapat melihat grafik yang telah ditambahkan sebelumnya. Data yang ditampilkan pada grafik dapat difilter berdasarkan rentang waktu tertentu dengan menekan tombol *Filter*. Untuk menambahkan grafik baru, pengguna harus terlebih dahulu memasuki mode *Edit*, dengan ketentuan bahwa *dashboard* telah dibuat sebelumnya. Selain itu, pada halaman ini, pengguna juga memiliki kemampuan untuk membuat lebih dari satu *dashboard* sesuai dengan kebutuhan visualisasi data.



1. Halaman *Dashboard*

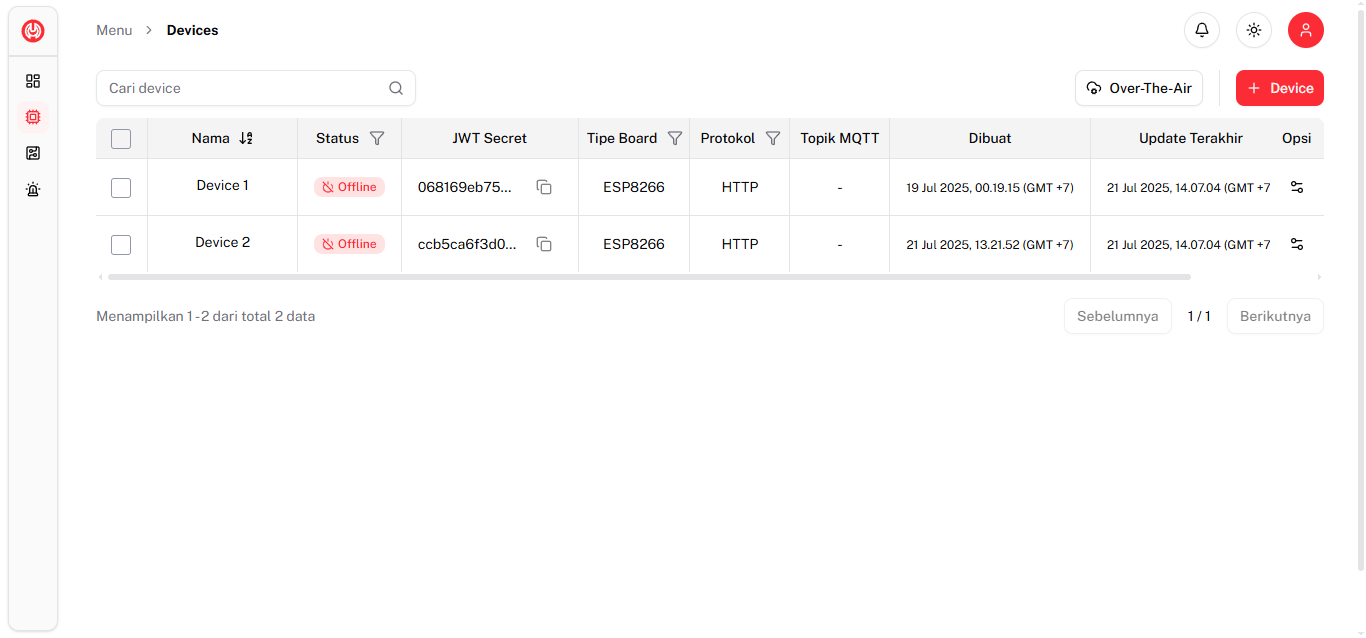
(Sumber: Dokumentasi pribadi)

* + 1. Mengakses Halaman *Devices*

Pada halaman *Devices*, pengguna dapat melihat daftar perangkat (*devices*) atau perangkat IoT yang telah ditambahkan sebelumnya. Tampilan halaman ini disusun dalam bentuk tabel yang memuat informasi terkait perangkat yang tersedia. Adapun isi dari tabel tersebut meliputi:

|  |  |  |
| --- | --- | --- |
| 1) | Nama : | Kolom ini menampilkan nama *device* atau perangkat IoT yang telah ditambahkan oleh pengguna. |
| 2) | Status : | Kolom ini menampilkan status konektivitas perangkat. Perangkat akan berstatus *Offline* apabila tidak terhubung dengan server, dan *Online* apabila perangkat berhasil terhubung dengan server. |
| 3) | JWT *Secret*  : | Kolom ini memuat token JWT yang diperlukan oleh perangkat untuk dapat terhubung dengan server. |
| 4) | Tipe *Board* : | Kolom ini mencantumkan jenis *board* yang digunakan oleh perangkat. |
| 5) | Protokol : | Kolom ini menunjukkan jenis protokol komunikasi yang digunakan oleh perangkat. |
| 6) | Topik MQTT : | Kolom ini berisi topik yang digunakan apabila perangkat menggunakan protokol MQTT untuk berkomunikasi dengan server. |
| 7) | Dibuat : | Kolom ini menunjukkan waktu saat data perangkat pertama kali dibuat. |
| 8) | Update Terakhir: | Kolom ini menampilkan waktu terakhir data perangkat diperbarui atau diubah. |
| 9) | Opsi : | Kolom ini menyediakan tombol untuk mengedit maupun menghapus data perangkat yang telah ditambahkan. |

Tampilan halaman *Devices* secara keseluruhan dapat dilihat pada Gambar berikut.

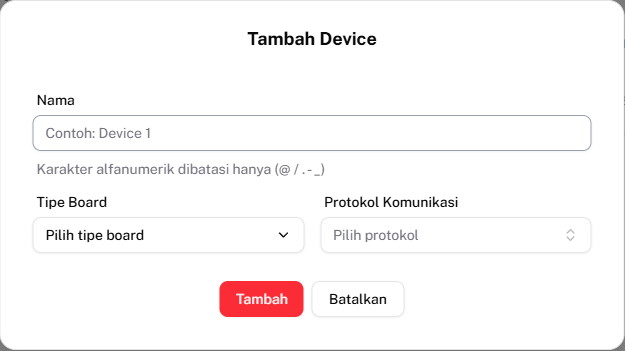


1. Halaman *Devices*

(Sumber: Dokumentasi pribadi)

* + 1. Menambahkan Perangkat IoT

Penambahan perangkat IoT dapat dilakukan dengan menekan tombol “+ Device” yang terletak di pojok kanan atas tabel pada halaman *Devices*. Setelah tombol tersebut ditekan, akan muncul formulir “Tambah Device” sebagaimana ditunjukkan pada Gambar berikut.

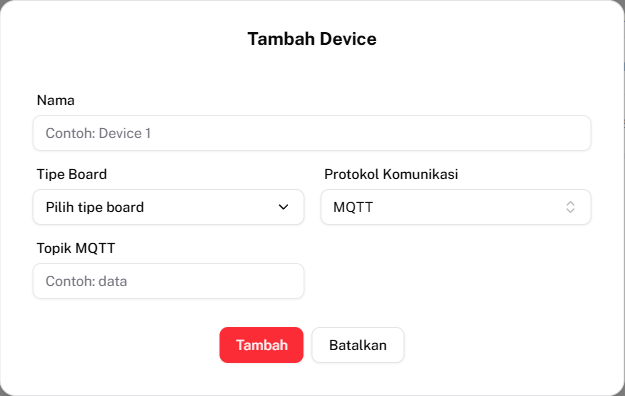


1. Formulir tambah *device*

(Sumber: Dokumentasi pribadi)

Formulir ini terdiri atas beberapa isian, antara lain:

|  |  |  |
| --- | --- | --- |
| 1) | Nama : | Kolom isian untuk memberikan nama pada perangkat (*device*) yang akan ditambahkan. |
| 2) | Tipe *Board* : | Kolom berupa pilihan jenis *board* yang akan digunakan, seperti ESP32, ESP8266, Arduino, Raspberry Pi, atau opsi lainnya. |
| 3) | Protokol Komunikasi : | Kolom berupa pilihan jenis protokol komunikasi yang digunakan oleh perangkat, yaitu HTTP atau MQTT. |
| 4) | Topik MQTT : | Kolom isian untuk menentukan *topic* yang akan digunakan oleh perangkat jika protokol yang dipilih adalah MQTT. Kolom ini hanya akan ditampilkan apabila pengguna memilih protokol MQTT, sebagaimana diperlihatkan pada Gambar 4.7. |



1. Formulir tambah *device* ketika memilih protokol MQTT

(Sumber: Dokumentasi pribadi)

Setelah perangkat (*device*) berhasil ditambahkan, akan ditampilkan indikator keberhasilan sebagaimana ditunjukkan pada Gambar berikut.



1. Indikator berhasil menambahkan perangkat

(Sumber: Dokumentasi Pribadi)

* + 1. Mengubah Data Perangkat IoT

Pengubahan data perangkat dapat dilakukan dengan menekan tombol *edit* pada kolom Opsi. Setelah tombol ditekan, sistem akan menampilkan formulir bertajuk “Edit [Nama Device]” yang memuat informasi perangkat yang telah ditambahkan sebelumnya, sebagaimana ditunjukkan pada Gambar berikut.



1. Formulir edit *device*

(Sumber: Dokumentasi pribadi)

Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar berikut.

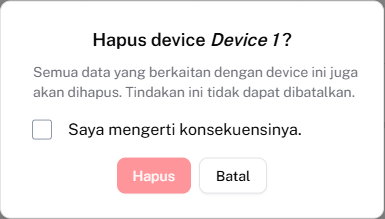


1. Indikator berhasil mengubah data perangkat

(Sumber: Dokumentasi pribadi)

* + 1. Menghapus Perangkat IoT

Pengguna dapat menghapus data perangkat IoT dengan menekan tombol “Hapus” yang tersedia pada kolom Opsi. Setelah tombol tersebut ditekan, sistem akan menampilkan dialog konfirmasi penghapusan yang berisi peringatan mengenai konsekuensi dari tindakan tersebut, yaitu seluruh data yang terkait dengan perangkat akan turut terhapus dan tindakan ini bersifat permanen serta tidak dapat dibatalkan, sebagaimana ditunjukkan pada Gambar berikut.



1. Peringatan persetujuan hapus *device*

(Sumber: Dokumentasi pribadi)

Setelah pengguna menyetujui konsekuensi dengan mencentang pernyataan persetujuan dan menekan tombol “Hapus”, sistem akan menampilkan indikator bahwa proses penghapusan telah berhasil, sebagaimana diperlihatkan pada Gambar berikut.



1. Indikator berhasil menghapus perangkat

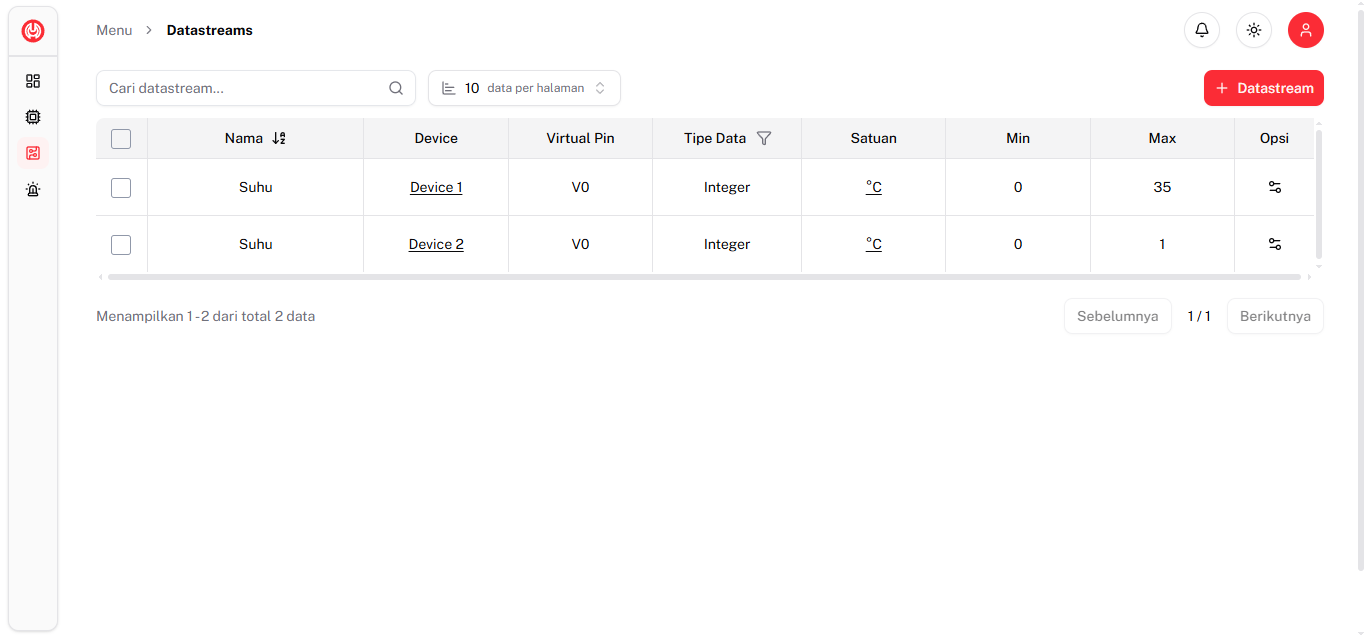
(Sumber: Dokumentasi pribadi)

* + 1. Menampilkan Halaman *Datastreams*

Pada halaman *Datastreams*, pengguna dapat melihat daftar *datastream* yang berasal dari perangkat IoT yang telah ditambahkan sebelumnya. Tampilan halaman ini disusun dalam bentuk tabel yang menyajikan informasi detail mengenai masing-masing *datastream* dari perangkat yang tersedia. Adapun isi dari tabel tersebut meliputi:

|  |  |  |
| --- | --- | --- |
| 1) | Nama : | Kolom ini menampilkan nama *datastream* atau parameter yang telah ditambahkan oleh pengguna. |
| 2) | *Device*  : | Kolom ini menunjukkan nama perangkat (*device*) yang terhubung dengan *datastream* tersebut. |
| 3) | *Virtual Pin* : | Kolom ini menampilkan nomor *virtual pin* yang digunakan oleh *datastream* untuk keperluan visualisasi pada halaman *dashboard*. |
| 4) | Tipe Data : | Kolom ini menunjukkan jenis data dari *datastream* atau parameter yang digunakan, seperti *Integer*, *String*, *Double*, atau *Boolean*. |
| 5) | Satuan : | Kolom ini menampilkan satuan pengukuran yang digunakan untuk data dari *datastream* atau parameter. |
| 7) | Min : | Kolom ini menunjukkan nilai minimum yang diperbolehkan untuk *datastream* atau parameter tersebut. |
| 8) | Max : | Kolom ini menunjukkan nilai maksimum yang diperbolehkan untuk *datastream* atau parameter tersebut. |
| 9) | Opsi : | Kolom ini menyediakan tombol untuk mengedit maupun menghapus data *datastream* yang telah ditambahkan. |

Tampilan halaman *Datastreams* secara keseluruhan dapat dilihat pada Gambar berikut.

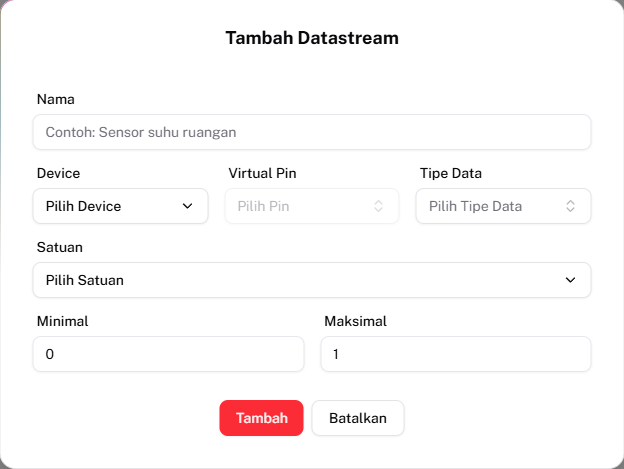


1. Halaman *Datastreams*

(Sumber: Dokumentasi pribadi)

* + 1. Menambahkan *Datastream*

Penambahan datastream dapat dilakukan dengan menekan tombol “+ Datastream” yang terletak di pojok kanan atas tabel pada halaman *Datastreams*. Setelah tombol tersebut ditekan, akan muncul formulir “Tambah Datastream” sebagaimana ditunjukkan pada Gambar berikut.



1. Formulir tambah *datastream*

(Sumber: Dokumentasi pribadi)

Setelah perangkat (*device*) berhasil ditambahkan, akan ditampilkan indikator keberhasilan sebagaimana ditunjukkan pada Gambar berikut.

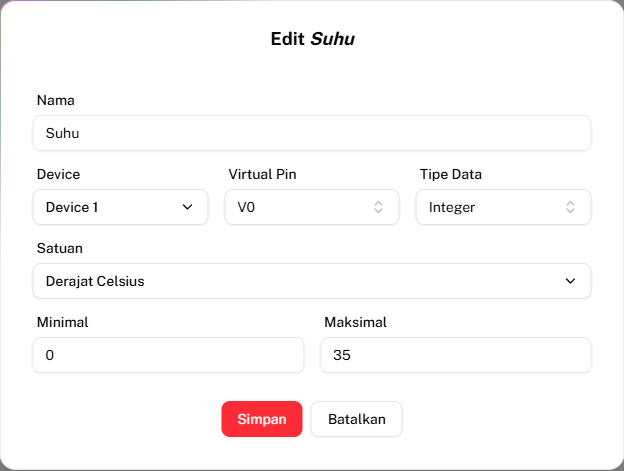


1. Indikator berhasil menambahkan *datastream*

(Sumber: Dokumentasi pribadi)

* + 1. Mengubah *Datastream*

Pengubahan data perangkat dapat dilakukan dengan menekan tombol *edit* pada kolom Opsi. Setelah tombol ditekan, sistem akan menampilkan formulir bertajuk “Edit [Nama Datastream]” yang memuat informasi perangkat yang telah ditambahkan sebelumnya, sebagaimana ditunjukkan pada Gambar berikut.



1. Formulir edit *datastream*

(Sember: Dokumentasi pribadi)

Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar berikut.

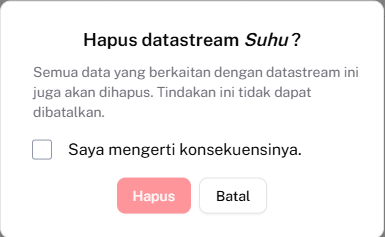


1. Indikator berhasil mengubah *datastream*

(Sumber: Dokumentasi pribadi)

* + 1. Menghapus *Datastream*

Pengguna dapat menghapus data *datastream* dengan menekan tombol “Hapus” yang tersedia pada kolom Opsi. Setelah tombol tersebut ditekan, sistem akan menampilkan dialog konfirmasi penghapusan yang berisi peringatan mengenai konsekuensi dari tindakan tersebut, yaitu seluruh data yang terkait dengan perangkat akan turut terhapus dan tindakan ini bersifat permanen serta tidak dapat dibatalkan, sebagaimana ditunjukkan pada Gambar berikut.



1. Peringatan persetujuan hapus *datastream*

(Sumber: Dokumentasi pribadi)

Setelah pengguna menyetujui konsekuensi dengan mencentang pernyataan persetujuan dan menekan tombol “Hapus”, sistem akan menampilkan indikator bahwa proses penghapusan telah berhasil, sebagaimana diperlihatkan pada Gambar berikut.

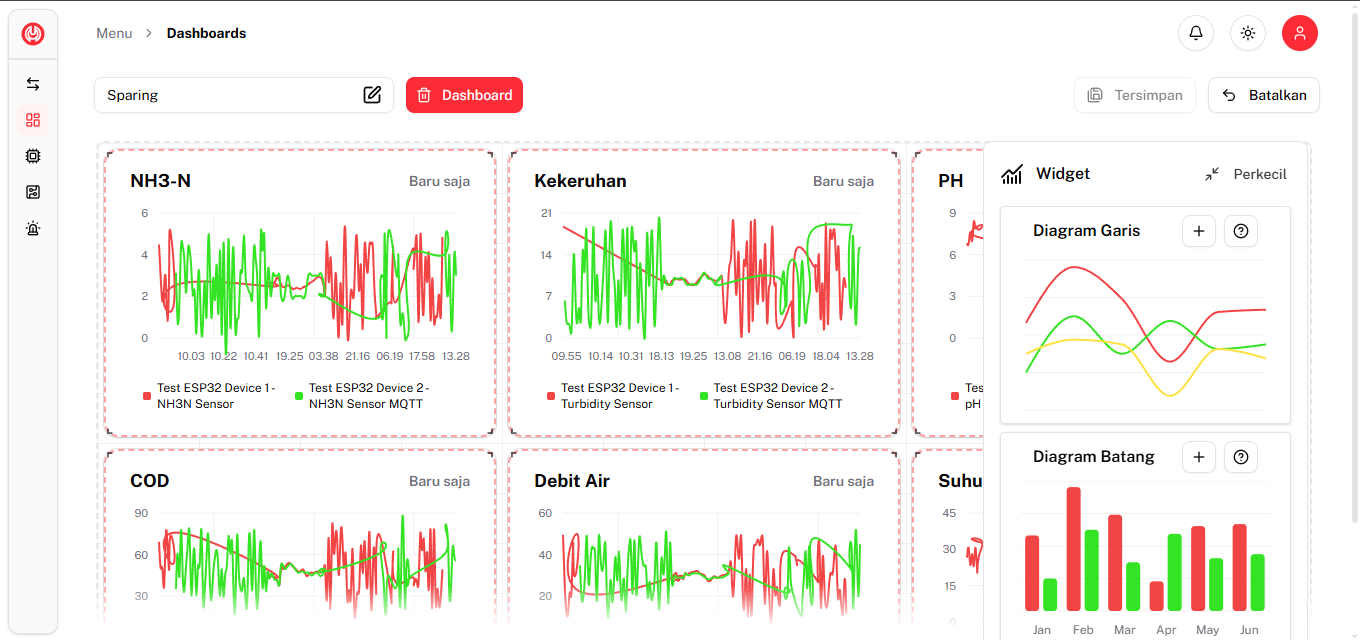


1. Indikator berhasil hapus *datastream*

(Sumber: Dokumentasi pribadi)

* + 1. Menambahkan Grafik Pemantauan

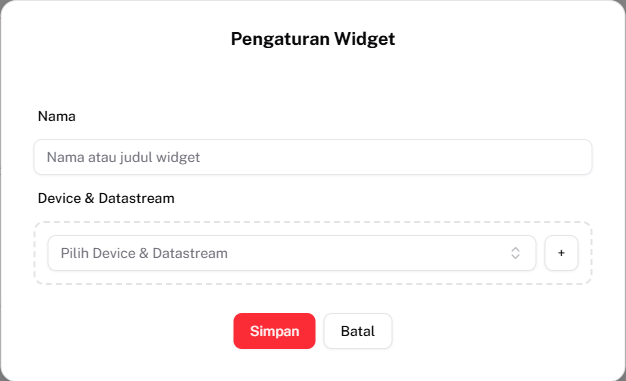
Penambahan grafik (*widget*) dapat dilakukan setelah pengguna terlebih dahulu membuat data perangkat dan *datastream* yang terkait. Proses ini dilakukan melalui halaman *Dashboard*, dengan langkah awal membuat *dashboard* baru dengan menekan tombol “+ Dashboard”. Setelah *dashboard* berhasil dibuat, pengguna dapat mengaktifkan mode pengeditan dengan menekan tombol *Edit* yang terletak di pojok kanan atas halaman. Pada mode ini, pengguna dapat menambahkan jenis grafik seperti *Line Chart*, *Bar Chart*, atau *Area Chart*, sebagaimana ditunjukkan pada Gambar berikut.



1. Mode edit halaman *Dashboard*

(Sumber: Dokumentasi pribadi)

Grafik dapat ditambahkan dengan dua cara, yaitu menekan tombol “+” atau menarik jenis grafik yang diinginkan ke dalam kanvas *dashboard*. Setelah grafik diletakkan pada kanvas, sistem akan menampilkan formulir konfigurasi grafik seperti diperlihatkan pada Gambar berikut.



1. Formulir pengaturan grafik (*widget*)

(Sumber: Dokumentasi pribadi)

Formulir ini terdiri dari beberapa komponen isian, yaitu:

|  |  |  |
| --- | --- | --- |
| 1) | Nama : | Kolom isian untuk memberikan nama pada grafik (*widget*) yang akan ditambahkan. |
| 2) | *Device* & *Datastream* : | Kolom pilihan untuk menentukan dari perangkat (*device*) mana yang ingin ditambahkan beserta *datastream*-nya. |

Setelah mengisi seluruh komponen pada formulir konfigurasi, pengguna dapat menyesuaikan posisi dan ukuran grafik sesuai kebutuhan. Jika telah selesai, pengguna dapat menekan tombol “Simpan” yang terletak di pojok kanan atas halaman. Setelah proses penyimpanan berhasil, sistem akan menampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar berikut.

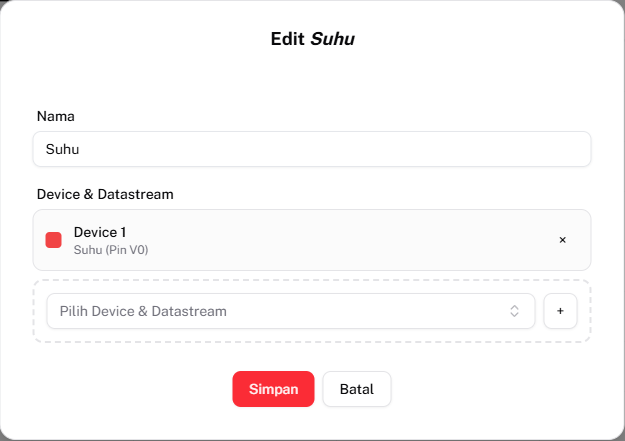


1. Indikator berhasil menyimpan *dashboard*

(Sumber: Dokumentasi pribadi)

* + 1. Mengubah Grafik Pemantauan

Pengubahan grafik (*widget*) dapat dilakukan dengan memasuki mode *edit* pada halaman *Dashboard*. Dalam mode ini, pengguna dapat melakukan penyesuaian terhadap ukuran, posisi, serta data yang terkait oleh grafik, termasuk data perangkat (*device*) maupun parameter (*datastream*) yang digunakan. Untuk mengubah data yang ditampilkan pada grafik, pengguna dapat menekan tombol *Opsi* yang terletak di pojok kanan atas grafik saat berada dalam mode *edit*. Setelah tombol tersebut ditekan, sistem akan menampilkan formulir pengaturan seperti yang ditunjukkan pada Gambar berikut.



1. Formulir edit grafik (*widget*)

(Sumber: Dokumentasi pribadi)

Setelah proses penyimpanan berhasil, sistem akan menampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar 4.22.

* + 1. Menghapus Grafik Pemantauan

Penghapusan grafik (*chart*) dapat dilakukan oleh pengguna dengan menekan tombol “Hapus” yang terletak di pojok kanan atas grafik saat berada dalam mode *Edit* pada halaman *Dashboard*. Setelah grafik dihapus dari kanvas *dashboard* dan pengguna menekan tombol “Simpan”, sistem akan menghapus grafik tersebut dari tampilan *dashboard* secara permanen dan menampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar 4.22.

* + 1. Menambahkan Alarm Notifikasi

Penambahan *alarm* dapat dilakukan dengan menekan tombol “Tambah Alarm” pada halaman *Alarm*. Setelah tombol tersebut ditekan, sistem akan menampilkan formulir konfigurasi seperti yang ditunjukkan pada Gambar berikut.



1. Formulir tambah *alarm*

(Sumber: Dokumentasi pribadi

Formulir ini terdiri atas beberapa komponen isian, antara lain:

|  |  |  |
| --- | --- | --- |
| 1) | Deskripsi : | Kolom isian untuk memberikan deskripsi pada *alarm* yang akan ditambahkan. |
| 2) | *Device*  : | Kolom pilihan untuk menentukan perangkat (*device*) yang akan dikaitkan dengan *alarm*. |
| 3) | *Datastream* : | Kolom pilihan untuk memilih *datastream* dari perangkat yang telah ditentukan sebelumnya. |
| 4) | Tunggu : | Kolom isian untuk menetapkan waktu jeda pengiriman ulang notifikasi *alarm* dalam satuan menit. |
| 5) | Kondisi : | Kolom isian untuk menetapkan ambang batas nilai suatu parameter yang akan memicu *alarm*. |
| 6) | Status : | Tombol pengaturan untuk menentukan apakah *alarm* dalam kondisi aktif atau tidak aktif guna menerima notifikasi. |

Setelah *alarm* berhasil ditambahkan, akan ditampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar berikut.



1. Indikator berhasil menambahkan *alarm*

(Sumber: Dokumentasi pribadi)

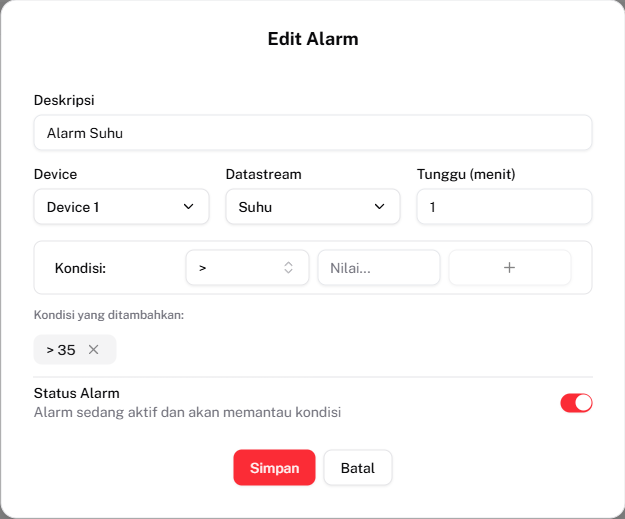
* + 1. Mengubah Alarm Notifikasi

Pengubahan data *alarm* dapat dilakukan dengan menekan tombol *Edit* pada kolom Opsi. Setelah tombol ditekan, sistem akan menampilkan formulir bertajuk “Edit Alarm” yang memuat informasi *alarm* yang telah ditambahkan sebelumnya, sebagaimana ditunjukkan pada Gambar 4.27. Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar berikut.



1. Indikator berhasil mengubah *alarm*

(Sumber: Dokumentasi pribadi)

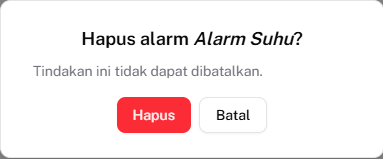


1. Formulis edit *alarm*

(Sumber: Dokumentasi pribadi)

* + 1. Menghapus Alarm Notifikasi

Pengguna dapat menghapus data *alarm* dengan menekan tombol “Hapus” yang tersedia pada kolom Opsi. Setelah tombol tersebut ditekan, sistem akan menampilkan dialog konfirmasi penghapusan yang berisi peringatan mengenai tindakan ini tidak dapat dibatalkan, sebagaimana ditunjukkan pada Gambar berikut.



1. Peringatan persetujuan hapus *alarm*

(Sumber: Dokumentasi pribadi)

Setelah pengguna menyetujui dengan menekan tombol “Hapus”, sistem akan menampilkan indikator bahwa proses penghapusan telah berhasil, sebagaimana diperlihatkan pada Gambar berikut.



1. Indikator berhasil hapus *alarm*

(Sumber: Dokumentasi pribadi)

1. Hasil Pengujian Sistem Keamanan

Bagian ini menyajikan hasil dari serangkaian pengujian yang dilakukan untuk mengevaluasi aspek keamanan sistem pemantauan limbah cair industri yang telah dikembangkan. Pengujian difokuskan pada validasi mekanisme perlindungan data, khususnya pada proses transmisi dan otentikasi, guna memastikan bahwa informasi yang dikirim antara klien dan server tetap aman, utuh, dan hanya dapat diakses oleh entitas yang berwenang. Tujuan utama dari pengujian ini adalah untuk mengukur sejauh mana sistem mampu menangani berbagai skenario autentikasi menggunakan JSON Web Token (JWT), serta menjamin bahwa payload tidak dapat dimodifikasi atau diakses secara tidak sah selama proses komunikasi berlangsung. Hasil pengujian dirangkum dalam Tabel 4.3.

1. Hasil pengujian sistem keamanan

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Deskripsi Pengujian** | **Gambar** | **Status** |
| 1. | Melakukan pengecekan enkripsi pada *payload* | Proses pengecekan payload HTTP    Proses pengecekan payload MQTT |  |
| 2. | Mengakses halaman aplikasi menggunakan token JWT yang valid. |  |  |
| 3. | Mengakses halaman aplikasi menggunakan token JWT yang dimodifikasi. |  |  |
| 4. | Mengakses halaman aplikasi tanpa disertai *refresh token* JWT |  |  |

Proses pengiriman data dari perangkat ESP32 ke *server* diawali dengan pengecekan atribut pada *header* permintaan. *Server* terlebih dahulu memverifikasi keberadaan atribut *Authorization* dalam *header*, yang harus memuat token JWT dengan format standar, yaitu Bearer <eyJ…>. Setelah memastikan bahwa format token JWT telah sesuai, *server* melanjutkan proses dengan melakukan verifikasi terhadap token tersebut.

Langkah pertama dalam verifikasi adalah melakukan *decoding* terhadap token JWT untuk memperoleh informasi yang terdapat pada bagian *payload*, khususnya nilai *sub*, yang merepresentasikan ID unik perangkat yang telah terdaftar pada sistem. Berdasarkan ID tersebut, *server* akan mengambil nilai *secret key* yang terasosiasi dengan perangkat dari *database*. Selanjutnya, *server* menggunakan nilai *secret key* tersebut untuk memverifikasi tanda tangan digital (*signature*) dari token JWT yang dikirim oleh perangkat ESP32.

Apabila proses verifikasi tanda tangan berhasil, *server* akan melanjutkan dengan mendekripsi isi *payload* yang berisi data sensor, menggunakan algoritma *Advanced Encryption Standard* (AES). Data sensor yang berhasil didekripsi kemudian disimpan ke dalam tabel *raw\_payloads* sebagai cadangan atau *backup* untuk keperluan pengelolaan dan analisis data lebih lanjut.

Setelah proses pengiriman data dari perangkat ke server berhasil dilakukan, sistem akan melanjutkan dengan pengelolaan autentikasi dan otorisasi pengguna melalui penggunaan token JWT (*JSON Web Token*). Mekanisme ini memastikan bahwa hanya pengguna yang telah terverifikasi yang dapat mengakses sumber daya sistem.

Setelah pengguna berhasil melakukan proses *login*, sistem akan memberikan dua jenis token JWT, yaitu *access token* dan *refresh token*. Kedua token tersebut disimpan dalam *cookie* browser dengan konfigurasi sebagai berikut:

|  |  |  |
| --- | --- | --- |
| 1. | *Name* : | Menyatakan nama variabel dari token JWT. Dalam sistem ini digunakan nama *access\_token* dan *refresh\_token* sebagai label *cookie*. |
| 2. | *Value*  : | Merupakan isi token JWT yang diterima dari *server*, yang disimpan sebagai nilai utama dalam *cookie*. |
| 3. | *Domain* : | Menunjukkan *domain* tempat asal token, yaitu *domain* dari server yang mengeluarkan token tersebut. |
| 4. | *Max-Age* : | Menentukan batas waktu kedaluwarsa token dalam satuan detik, yang merepresentasikan durasi validitas token sejak waktu penyimpanan (format UTC). |
| 5. | *HttpOnly* : | Parameter ini bernilai *true* agar *cookie* tidak dapat diakses melalui skrip *JavaScript* di sisi klien, melainkan hanya melalui permintaan HTTP. Pengaturan ini penting untuk mencegah potensi serangan XSS (*Cross-Site Scripting*). |
| 6. | *Secure* : | Jika bernilai *true*, maka *cookie* hanya akan dikirimkan melalui koneksi HTTPS yang terenkripsi, guna memastikan keamanan data dalam transmisi. |
| 7. | *SameSite* : | Merupakan parameter pengendali pengiriman *cookie* dalam konteks *cross-site*. Sistem ini menggunakan nilai None agar *cookie* tetap dapat dikirim dari *server* ke klien meskipun berasal dari domain yang berbeda. |

Konfigurasi *HttpOnly* = *true* memiliki peran yang sangat krusial dalam konteks keamanan, karena memastikan bahwa token JWT tidak dapat diakses secara langsung melalui *JavaScript* di sisi klien. Dengan demikian, token hanya dapat digunakan melalui permintaan HTTP yang dikirim ke endpoint API pada api.misred-iot.com, sehingga mengurangi risiko penyalahgunaan token oleh pihak yang tidak berwenang.

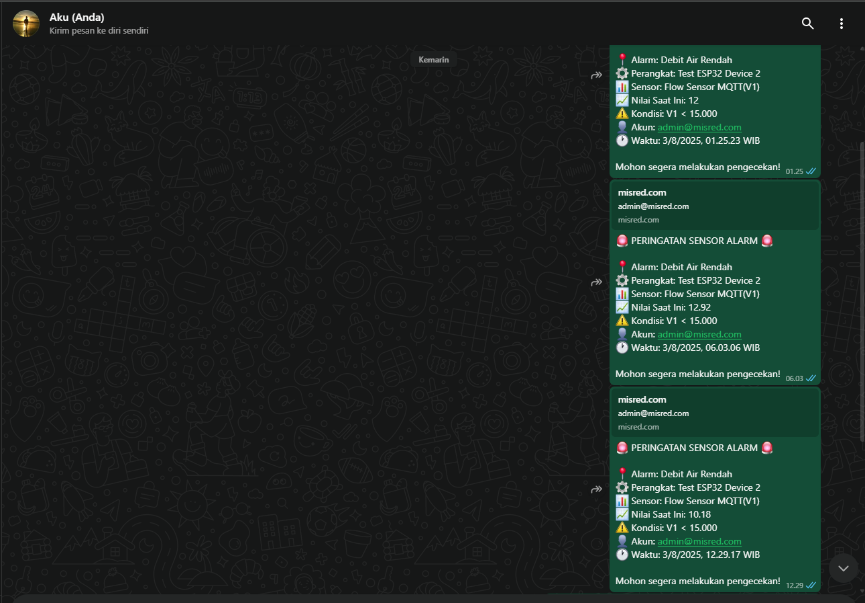
1. Hasil Pengujian Notifikasi

Pengujian notifikasi dilakukan untuk menilai kemampuan sistem dalam memberikan respons secara real-time melalui tampilan notifikasi pada browser dan perangkat Android ketika parameter limbah cair berada di luar batas normal. Tujuan dari pengujian ini adalah untuk memastikan bahwa sistem dapat merespons kondisi parameter lingkungan yang berada di bawah ambang batas yang ditetapkan, serta menyampaikan peringatan secara akurat dan tepat waktu. Hasil pengujian terhadap masing-masing skenario kondisi parameter ditampilkan pada Tabel 4.4.

1. Hasil pengujian notifikasi

|  |  |  |
| --- | --- | --- |
| **No.** | **Deskripsi Pengujian** | **Gambar** |
| 1. | Nilai datastream memicu alarm *browser* |  |
| 2. | Nilai datastream memicu alarm WhatsApp |  |

Pengguna akan menerima notifikasi apabila nilai suatu parameter melebihi atau berada di bawah ambang batas yang telah ditentukan sebelumnya melalui halaman *Alarms*. Notifikasi ini dapat disampaikan melalui dua media, yaitu melalui tampilan pada browser maupun melalui aplikasi WhatsApp. Pada tampilan browser, notifikasi akan muncul di pojok kanan atas halaman dalam bentuk ikon lonceng. Sementara itu, notifikasi melalui WhatsApp hanya akan dikirim apabila pengguna telah menambahkan nomor telepon yang terhubung dengan akun WhatsApp pada bagian *pengaturan profil akun* serta mengaktifkan fitur notifikasi tersebut. Tampilan notifikasi yang diterima melalui aplikasi WhatsApp ditunjukkan pada Gambar berikut.



1. Notifikasi pada akun WhatsApp

(Sumber: Dokumentaasi pribadi)

1. Hasil Pengujian Performa

Pengujian performa dilakukan untuk mengidentifikasi sejauh mana sistem yang dikembangkan mampu beroperasi secara efisien dalam memanfaatkan sumber daya perangkat, baik pada platform website maupun aplikasi Android. Tujuan dari pengujian ini adalah untuk mengukur tingkat penggunaan memori, prosesor (CPU), serta jaringan, guna memastikan bahwa sistem dapat berjalan secara optimal dalam berbagai kondisi operasional. Pada aplikasi berbasis website, pengujian dilakukan dengan memanfaatkan fitur *Network Monitor* pada browser Google Chrome untuk memperoleh data waktu pemuatan halaman, meliputi waktu pemuatan struktur HTML (*DOMContentLoaded*), waktu pemuatan halaman utama tanpa ikon (*load*), serta waktu pemuatan keseluruhan (*finish*). Sementara itu, pengujian performa pada aplikasi Android menggunakan fitur *Android Profiler* pada Android Studio, yang memberikan informasi terkait aktivitas CPU, dan penggunaan memori, selama aplikasi dijalankan. Hasil pengujian tersebut disajikan pada Tabel 4.5.

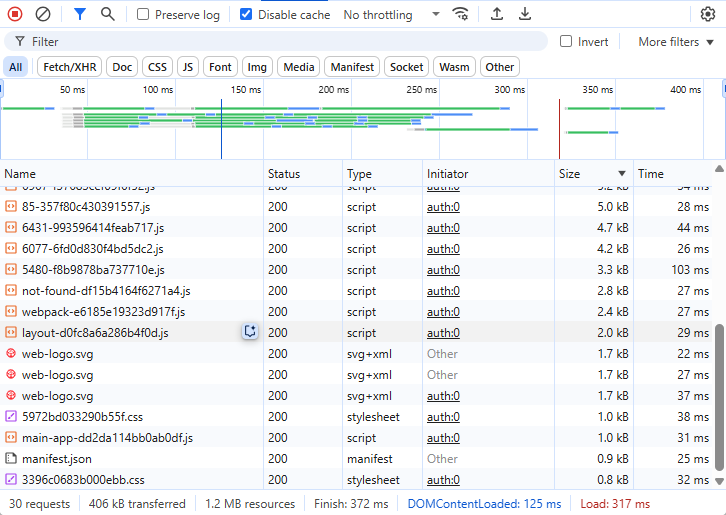
1. Hasil pengujian performa aplikasi *website*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Halaman** | **Waktu memuat (ms)** | | |
| ***DOMContentLoaded*** | ***Load*** | ***Finish*** |
| 1. | Login | 125 | 317 | 372 |
| 2. | Dashboards | 179 | 895 | 3060 |
| 3. | Devices | 155 | 1090 | 3320 |
| 4. | Datastreams | 203 | 939 | 3320 |
| 5. | Alarms | 165 | 819 | 2960 |

Berdasarkan hasil pengujian performa aplikasi website yang ditampilkan pada Tabel 4.5, pengujian dilakukan terhadap lima halaman utama, yaitu:

* + 1. Halaman *Login*

Halaman ini berfungsi untuk melakukan autentikasi pengguna melalui input kredensial berupa email dan kata sandi. Hasil dari “Network Monitor” adalah sebagai berikut.



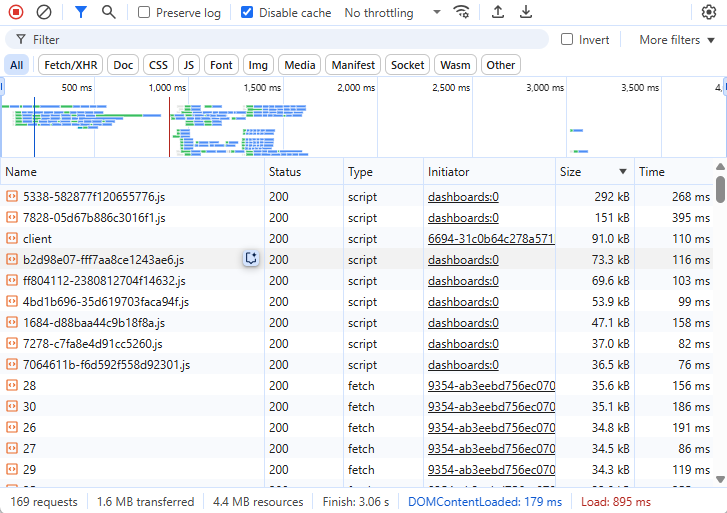
1. Hasil dari “Network Monitor” halaman *login*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil di atas, terdapat 30 permintaan yang dibutuhkan untuk memuat halaman, diantaranya yaitu beberapa *file* *JavaScript*, CSS, *font*, logo, dan satu dokumen untuk halaman *auth*. Keseluruhan halaman berukuran 1,2 MB dengan *file* terbesarnya adalah dokumen *auth* yang berukuran 0,5 kB.

* + 1. Halaman *Dashboards*

Halaman ini menampilkan grafik data dari parameter-parameter yang telah ditambahkan oleh pengguna. Hasil dari “Network Monitor” adalah sebagai berikut.



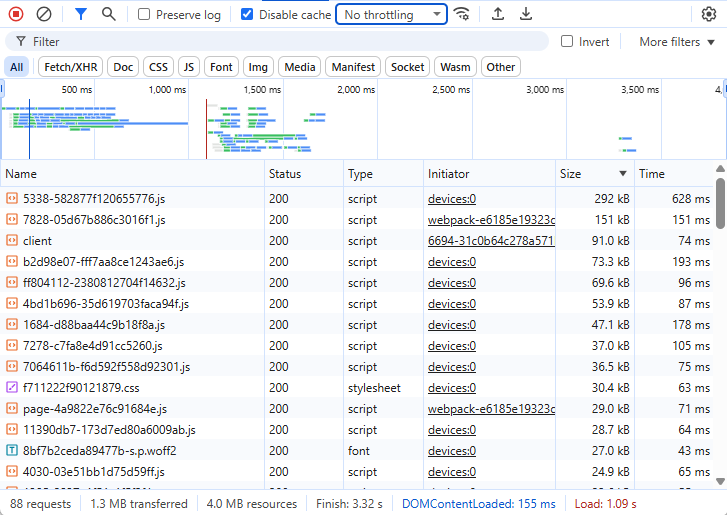
1. Hasil dari “Network Monitor” halaman *dashboards*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil di atas, terdapat 169 permintaan yang dibutuhkan untuk memuat halaman, diantaranya yaitu beberapa *file JavaScript*, CSS, *font*, logo, permintaan data, dan satu dokumen untuk halaman *dashboard*. Permintaan data dilakukan untuk mengambil ringkasan parameter suhu, debit air, kekeruhan air, pH air, NH3-N (nitrogen ammonia), dan COD (*Chemical Oxygen Demand*). Keseluruhan halaman berukuran 4,4 MB dengan *file* terbesarnya adalah *file JavaScript* yang berukuran 292 kB dengan total waktu yang dibutuhkan browser untuk menyelesaikan permintaan adalah 268 ms.

* + 1. Halaman *Devices*

Halaman ini menyajikan data dalam bentuk tabel mengenai perangkat IoT yang telah ditambahkan oleh pengguna. Hasil dari “Network Monitor” adalah sebagai berikut.



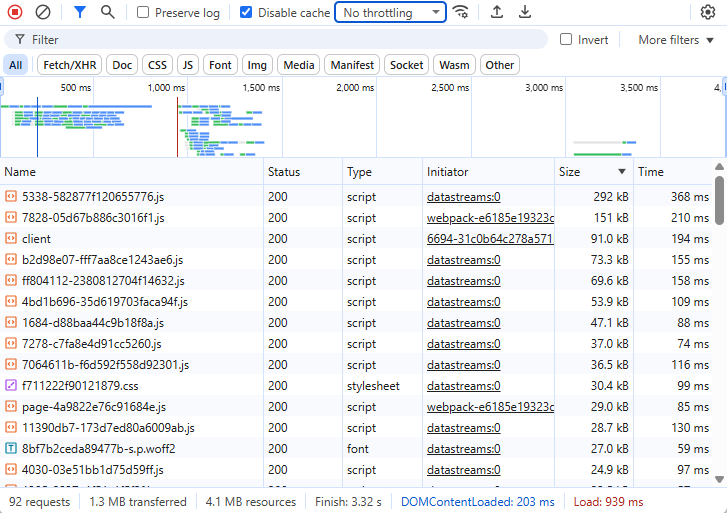
1. Hasil dari “Network Monitor” halaman *devices*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil di atas, terdapat 88 permintaan yang dibutuhkan untuk memuat halaman, diantaranya yaitu beberapa *file JavaScript*, CSS, *font*, logo, permintaan data, dan satu dokumen untuk halaman *devices*. Permintaan data dilakukan untuk memverifikasi token otoriasasi dan mengambil data perangkat. Keseluruhan halaman berukuran 4 MB dengan *file* terbesarnya adalah *file JavaScript* yang berukuran 292 kB dengan total waktu yang dibutuhkan browser untuk menyelesaikan permintaan adalah 628 ms.

* + 1. Halaman *Datastreams*

Halaman ini menampilkan data tabel dari parameter-parameter (datastream) perangkat IoT yang telah didaftarkan. Hasil dari “Network Monitor” adalah sebagai berikut.



1. Hasil dari “Network Monitor” halaman *datastreams*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil di atas, terdapat 92 permintaan yang dibutuhkan untuk memuat halaman, diantaranya yaitu beberapa *file JavaScript*, CSS, *font*, logo, permintaan data, dan satu dokumen untuk halaman *datastreams*. Permintaan data dilakukan untuk memverifikasi token otoriasasi dan mengambil data perangkat beserta parameternya. Keseluruhan halaman berukuran 4,1 MB dengan *file* terbesarnya adalah *file JavaScript* yang berukuran 292 kB dengan total waktu yang dibutuhkan browser untuk menyelesaikan permintaan adalah 368 ms.

* + 1. Halaman *Alarms*

Halaman ini menampilkan data tabel mengenai alarm, yaitu pengaturan ambang batas terhadap parameter dari perangkat IoT. Hasil dari “Network Monitor” adalah sebagai berikut.

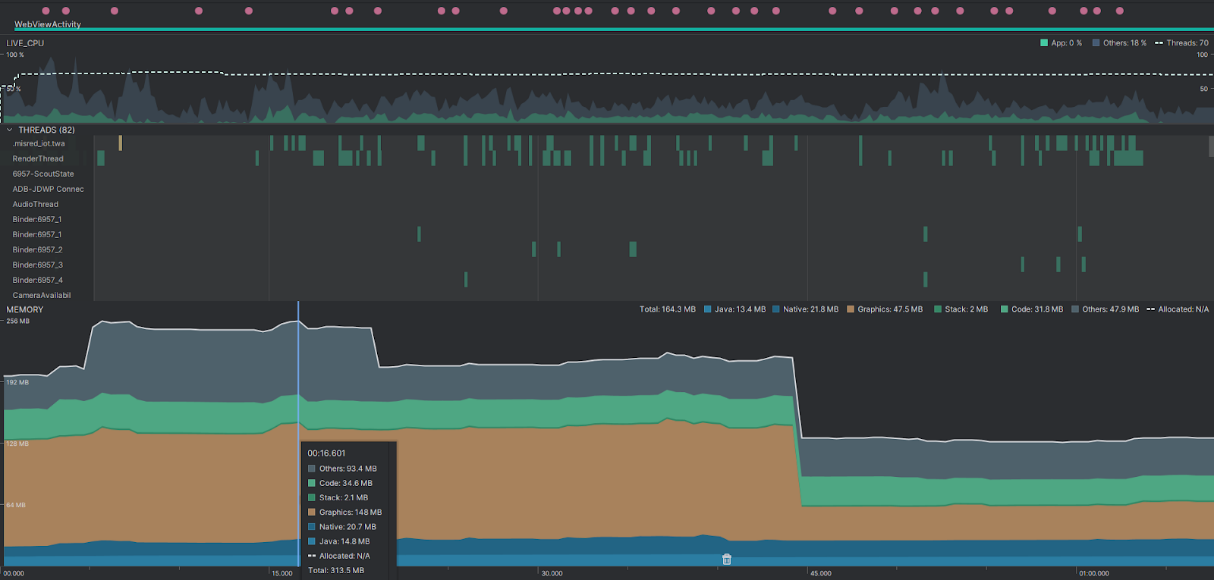


1. Hasil dari “Network Monitor” halaman *alarms*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil di atas, terdapat 85 permintaan yang dibutuhkan untuk memuat halaman, diantaranya yaitu beberapa *file JavaScript*, CSS, *font*, logo, permintaan data, dan satu dokumen untuk halaman *alarms*. Permintaan data dilakukan untuk memverifikasi token otoriasasi dan mengambil data alarm dari perangkat beserta parameternya. Keseluruhan halaman berukuran 4 MB dengan *file JavaScript* yang berukuran 292 kB dengan total waktu yang dibutuhkan browser untuk menyelesaikan permintaan adalah 311 ms.

Pengujian pada perangkat Android dilakukan menggunakan perangkat lunak Android Studio, khususnya melalui fitur Android Profiler dengan memanfaatkan *task View Live Telemetry*. Namun, hasil yang diperoleh dari Android Profiler bersifat terbatas karena aplikasi Android yang dikembangkan menggunakan pendekatan *Trusted Web Activity* (TWA), yang pada dasarnya hanya membungkus situs web kami, yaitu https://misred-iot.com. Akibatnya, seluruh aktivitas permintaan (request) berlangsung di dalam lingkungan WebView, sehingga Android Profiler hanya dapat merekam dan menampilkan informasi terkait aktivitas CPU, *threads*, dan penggunaan memori. Hasil pengujian tersebut disajikan pada gambar berikut.



1. Hasil pengujian Android Profiler dari Android Studio

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil pengujian menggunakan fitur *View Live Telemetry* pada Android Profiler, diperoleh gambaran umum mengenai performa aplikasi selama dijalankan dalam mode Trusted Web Activity (TWA). Grafik menunjukkan bahwa aktivitas utama yang terekam meliputi penggunaan CPU, aktivitas *thread*, dan konsumsi memori. Penggunaan CPU terlihat relatif stabil dengan rata-rata di bawah 10%, menandakan bahwa aplikasi berjalan dengan beban kerja yang ringan tanpa adanya lonjakan proses yang signifikan. Pada bagian *threads*, tercatat aktivitas dari 82 *thread* sistem yang aktif, di antaranya adalah *RenderThread*, *Binder*, *GCM* (Google Cloud Messaging), *WebView* Core, serta komponen *CamerashellActivity* yang menunjukkan bahwa proses rendering antarmuka, komunikasi antarproses, dan pemrosesan WebView berjalan selama periode pengujian.

Sementara itu, alokasi memori menunjukkan penurunan signifikan sekitar detik ke 45, dari total penggunaan sebesar 31,5 MB menjadi sekitar 20 MB. Rincian memori pada titik awal sebelum penurunan mencakup: Java sebesar 13,4 MB, *Native* sebesar 2,1 MB, *Graphics* sebesar 4,5 MB, *Stack* sebesar 2 MB, *Code* sebesar 3,1 MB, dan *Others* sebesar 6,4 MB. Setelah penurunan, total memori yang digunakan hanya berkisar di bawah 20 MB, mengindikasikan adanya proses *garbage collection* atau pembersihan memori oleh sistem.

Secara keseluruhan, performa aplikasi dalam lingkungan TWA dapat dikategorikan efisien dan stabil dari sisi sistem Android. Namun demikian, informasi yang ditampilkan terbatas pada lapisan Android karena seluruh aktivitas permintaan (*request*) terjadi di dalam lingkungan WebView. Hal ini menyebabkan profiler tidak dapat merekam trafik jaringan atau aktivitas internal dari situs web yang dibungkus, sehingga tidak memberikan representasi menyeluruh terhadap perilaku aplikasi secara *end-to-end*.

1. Evaluasi

………….

BAB V

PENUTUP

DAFTAR PUSTAKA

Ahmed, S., & Mahmood, Q. (2019). An authentication based scheme for applications using JSON web token. In *2019 22nd international multitopic conference (INMIC)* (pp. 1-6). IEEE.

Akasiadis, C., Tryferidis, A., & Tzovaras, D. (2019). Internet of Things architecture for combined applications. *IEEE Internet of Things Journal, 6*(2), 2055-2070. <https://doi.org/10.1109/JIOT.2018.2877660>

Ariata, C. (2023). Apa itu MySQL? pengertian MySQL, cara kerja, dan kelebihannya. *Hostinger*. <https://www.hostinger.co.id/tutorial/apa-itu-mysql>

AWS. (2023). *What is a relational database?*. Retrieved from<https://aws.amazon.com/id/relational-database/>

Biznetgio.com. (2024). Mengenal agile development, metode yang cocok diterapkan developer. *Biznetgio*. <https://www.biznetgio.com/news/apa-itu-agile-development>

Bogdan, R., Paliuc, C., Crisan-Vida, M., Nimara, S., & Barmayoun, D. (2023). Low-cost Internet-of-Things water-quality monitoring system for rural areas. *Sensors, 23*(8), 3919.<https://doi.org/10.3390/s23083919>

Brahmbhatt, M., & Vora, H. (2022). Building High-Performance Web Applications using Next.js. *International Journal of Scientific Research in Engineering and Management*, 6(3), 1–5.

Centenaro, M., Vangelista, L., Zanella, A., & Zorzi, M. (2016). Long-range communications in unlicensed bands: The rising stars in the IoT and smart city scenarios. *IEEE Wireless Communications*, *23*(5), 60-67. <https://doi.org/10.1109/MWC.2016.7721743>

Chirpstack.io. (2024). ChirpStack open-source LoRaWAN(R) Network Server. *GitHub.* <https://github.com/chirpstack/chirpstack>

Code.visualstudio.com. (2024). Visual Studio Code: Code editing. Redefined. *Microsoft*. <https://code.visualstudio.com/>

Dahiya, S., & Malik, S. (2020). A review on secure authentication techniques using JWT for web applications. *International Journal of Scientific & Technology Research*, 9(3), 6615–6619.

Darmawan, I., Mansyur, M. U., Imam, K. Z., Syahdan, M., & Fawaid, A. (2023). Evaluasi keamanan privilege terintegerasi JSON Web Token pada sistem informasi akademik. *Jurnal Informasi dan Teknologi*, 120-128.

Developer.android.com. (2024). Android Developers - The official site for Android developers. *Google.* <https://developer.android.com/>

Nextjs.org. (2024). *The React Framework for the Web*.<https://nextjs.org/>

Febriana A., Farros R.A. (2024). Rancang bangun industrial internet of things (IIoT) board multiguna untuk monitoring limbah cair. *Tugas Akhir*, Politeknik Negeri Semarang, 1-91.

Fielding, R. T., & Taylor, R. N. (2002). Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, *2*(2), 115-150.

Flanagan, D. (2011). JavaScript: The definitive guide: Activate your web pages. *O'Reilly Media, Inc*. <https://dl.acm.org/doi/10.5555/2029041>

Gillis, A. S. (2024). *What is SQL (Structured Query Language)?* TechTarget. Retrieved from https://www.techtarget.com/searchdatamanagement/definition/SQL

Gonzalez, L., Perez, M., & Romero, E. (2021). Effectiveness of real-time alert systems in IoT monitoring: A review. *Sensors,* *21*(15), 5189. <https://doi.org/10.3390/s21155189>

Hanif, L. (2024). HTTPS adalah: pengertian, fungsi, dan cara mengaktifkannya. *Rumahweb.* <https://www.rumahweb.com/journal/https-adalah/>

IBM. (2024). *What is SQL?* Retrieved from https://www.ibm.com/topics/sql

Jáquez, A. D. B., Herrera, M. T. A., Celestino, A. E. M., Ramírez, E. N., & Cruz, D. A. M. (2023). Extension of LoRa coverage and integration of an unsupervised anomaly detection algorithm in an IoT water quality monitoring system. *Water, 15*(7), 1351.<https://doi.org/10.3390/w15071351>

Json.org. (2006). JSON: The fat-free alternative to XML. *Json.Org.* <https://json.org>

Jwt.io. (2024). JSON web token. *Jwt.Io.* <https://jwt.io/>

Kim, J., Park, S., & Choi, Y. (2021). Design of a RESTful API System for an IoT Platform. *Sensors*, 21(14), 4642. https://doi.org/10.3390/s21144642

Mirani, A. A., Velasco-Hernandez, G., Awasthi, A., & Walsh, J. (2022). Key Challenges and Emerging Technologies in Industrial IoT Architectures: A Review. *Sensors*, *22*(15), 5836. https://doi.org/10.3390/s22155836

Myers, G. J., Sandler, C., & Badgett, T. (2011). The art of software testing. *ISBN: 0-471-46912-2, pp 234. Softw. Test., Verif. Reliab. 15.* 136-137. doi: 10.1002/stvr.322.

Nurbaya, S. (2019). *PERATURAN MENTERI LINGKUNGAN HIDUP DAN KEHUTANAN*. https://ppkl.menlhk.go.id/website/filebox/885/200306144608 Permen LHK tentang SPARING.pdf

Nurul F., Hidayat, H., & Eniati, E. (2020). Analysis of COD, BOD, and DO Levels at the Wastewater Treatment at the Balai PIALAM DPUP-ESDM Special Region of Yogyakarta. *IJCR-Indonesian Journal of Chemical Research, 5*(2), 80–89. <https://doi.org/10.20885/ijcer.vol5.iss2.art5>

Pramukantoro, E. K. (2020). Internet of Things dengan Python, ESP32, dan Raspberry PI: *Teori dan Praktik. Universitas Brawijaya Press*. Available at:<https://www.google.co.id/books/edition/Internet_of_Things_dengan_Python_ESP32_d/_1_UDwAAQBAJ?hl=en&gbpv=1&dq=Internet%20of%20Things%20dengan%20Python%2C%20ESP32%2C%20dan%20Raspberry%20PI%3A%20Teori%20dan%20Praktik&pg=PA46&printsec=frontcover>

Raman, R., & Martin, N. (2024). IoT-enabled water pollution detection for real-time monitoring and pollution source identification with MQTT protocol. *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, Chennai, India, 1–6. <https://doi.org/10.1109/ADICS58448.2024.10533607>

Salem, R. M. M., Saraya, M. S., & Ali-Eldin, A. M. T. (2022). An industrial cloud-based IoT system for real-time monitoring and controlling of wastewater. *IEEE Access, 10*, 6528–6540. <https://doi.org/10.1109/ACCESS.2022.3141977>

Sari, D. R. (2024). Analisis keamanan sistem informasi dalam era Internet of Things (IoT). *Technologia Journal: Jurnal Informatika*, *Vol.1, No.2,* pp. 1-10. Universitas Jambi, Indonesia.

Sawitri, D. (2023). Internet of Things memasuki era society 5.0. *KITEKTRO: Jurnal Komputer, Informasi Teknologi, dan Elektro*, *Vol.8 No.1*, pp. 31-35. Universitas Harapan Medan. e-ISSN: 2252-7036

Shakdher, A., Agrawal, S., & Yang, B. (2019). Security vulnerabilities in consumer IoT applications. 2019 IEEE 5th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 1-6. <https://doi.org/10.1109/BigDataSecurity-HPSC-IDS.2019.00012>

Shodiq, F. A., Pahlevi, R. R., & Sukarno, P. (2021). Secure MQTT authentication and message exchange methods for IoT constrained device. *2021 International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA)*, Bandung, Indonesia, 70–74. <https://doi.org/10.1109/ICICyTA53712.2021.9689126>

Sidiq, D. (2022). Berkenalan dengan figma. *Kemenkeu Learning Center.* <https://klc2.kemenkeu.go.id/kms/knowledge/berkenalan-dengan-figma-639a1df3/detail/>

Tarisa, D. (2022). Rangkuman materi diagram praktek pengalaman lapangan. *Scribd*. <https://id.scribd.com/document/541533166/Rangkuman-materi-PPL-Flowchart-usecase-activity-diagram-ERD-Sequence-diagram>

Thangavel, D., Ma, X., Valera, A., Tan, H. X., & Tan, C. K. Y. (2014). Performance evaluation of MQTT and CoAP via a common middleware. In *2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP)* (pp. 1-6). IEEE. <https://doi.org/10.1109/ISSNIP.2014.6827678>

Ushkov, A. N., Strelkov, N. O., Krutskikh, V. V., & Chernikov, A. I. (2023). Industrial Internet of Things platform for water resource monitoring. *2023 International Russian Smart Industry Conference (SmartIndustryCon)*, Sochi, Russian Federation, 593–599. https://doi.org/10. 1109/SmartIndustryCon57312.2023.10110776

Vercel. (2023). *Introduction to Next.js*. https://vercel.com/solutions/nextjs

Widiarto, S. (2024). RabbitMQ: pengertian, fungsi, kelebihan dan kekurangannya. *Rumahweb*. <https://www.rumahweb.com/journal/rabbitmq-adalah/>

W3Schools. (2024). *SQL Introduction*. Retrieved from <https://www.w3schools.com/sql/sql_intro.asp>

LAMPIRAN