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, 11 Agustus 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, 11 Agustus 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. Bapak Dr. Garup Lambang Goro, 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, 11 Agustus 2025

Penulis

ABSTRAK

*Imam Zaenal Abidin dan Muhammad Adriano Khairur Rizky 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, 99 halaman.*

Peningkatan kebutuhan terhadap sistem pemantauan limbah cair industri merupakan tantangan strategis dalam memastikan kepatuhan terhadap regulasi lingkungan dan mencegah pencemaran. Sebagian besar sistem yang telah ada belum mampu mengintegrasikan komunikasi multi-protokol, mekanisme keamanan berlapis, serta fitur notifikasi yang responsif. Penelitian ini merancang dan mengimplementasikan sistem pemantauan limbah cair berbasis *Internet of Things* (IoT) dengan JSON *Web* Token (JWT), mendukung protokol HTTP dan MQTT, dilengkapi enkripsi AES-128, HTTPS, serta antarmuka berbasis web dan Android. Metodologi yang digunakan adalah Agile-Scrum, mencakup tahap perencanaan, perancangan, implementasi, serta pengujian meliputi *black box*, keamanan, performa, notifikasi, dan *User Acceptance Testing* (UAT). Hasil penelitian menunjukkan sistem mampu memantau parameter debit, kekeruhan, pH, NH₃-N, dan COD secara *real-time*, dengan tingkat keberhasilan 100% pada 16 skenario pengujian *black box*. Pengujian performa mencatat waktu DOMContentLoaded <250 ms, penggunaan CPU Android <10%, dan konsumsi memori 20–31 MB. Entropi kunci AES-128 setara ketahanan teoretis ±10²¹ tahun terhadap serangan *brute force*. Notifikasi internal dan WhatsApp terbukti efektif memberikan peringatan saat parameter melampaui ambang batas. UAT menunjukkan tingkat penerimaan 93,25%, menandakan sistem memenuhi ekspektasi dari aspek antarmuka, dan pengalaman pengguna. Sistem direkomendasikan untuk industri berlimbah cair serta dapat diintegrasikan dengan *machine learning* pada penelitian mendatang guna prediksi kualitas limbah.

*ABSTRACT*

*Imam Zaenal Abidin dan Muhammad Adriano Khairur Rizky Setyawan.* ***“Industrial Wastewater Monitoring System Integrated with 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, 99 pages*

*The increasing need for reliable and secure industrial wastewater monitoring systems represents a strategic challenge in ensuring environmental regulatory compliance and preventing pollution. Most existing systems have not been able to integrate multi-protocol communication, layered security mechanisms, and responsive notification features. This research designs and implements an Internet of Things (IoT)-based wastewater monitoring system with JSON Web Token (JWT), supporting HTTP and MQTT protocols, equipped with AES-128 encryption, HTTPS, and web-based and Android interfaces. The methodology used is Agile-Scrum, covering planning, design, implementation, and testing phases including black box, security, performance, notification, and User Acceptance Testing (UAT). The research results show that the system is capable of monitoring flow rate, turbidity, pH, NH₃-N, and COD parameters in real-time, with a 100% success rate in 16 black box testing scenarios. Performance testing recorded DOMContentLoaded time <250 ms, Android CPU usage <10%, and memory consumption 20–31 MB. AES-128 key entropy provides theoretical resistance of ±10²¹ years against brute force attacks. Internal and WhatsApp notifications proved effective in providing alerts when parameters exceed threshold limits. UAT showed an acceptance rate of 93.25%, indicating the system meets expectations in terms of interface and user experience. The system is recommended for wastewater-producing industries and can be integrated with machine learning in future research for wastewater quality prediction.*

DAFTAR ISI

[PERNYATAAN KEASLIAN TUGAS AKHIR ii](#_Toc205898394)

[HALAMAN PERSETUJUAN iii](#_Toc205898395)

[HALAMAN PENGESAHAN iv](#_Toc205898396)

[PRAKATA v](#_Toc205898397)

[ABSTRAK vii](#_Toc205898398)

[*ABSTRACT* viii](#_Toc205898399)

[DAFTAR ISI ix](#_Toc205898400)

[DAFTAR GAMBAR xii](#_Toc205898401)

[DAFTAR TABEL xv](#_Toc205898402)

[DAFTAR LAMPIRAN xvi](#_Toc205898403)

[BAB I PENDAHULUAN 1](#_Toc205898404)

[1.1 Latar Belakang 1](#_Toc205898405)

[1.2 Rumusan Masalah 3](#_Toc205898406)

[1.3 Tujuan 3](#_Toc205898407)

[1.4 Manfaat 3](#_Toc205898408)

[1.5 Batasan Masalah 4](#_Toc205898409)

[1.6 Sistematika Penulisan 4](#_Toc205898410)

[BAB II TINJAUAN PUSTAKA DAN DASAR TEORI 6](#_Toc205898411)

[2.1 Tinjauan Pustaka 6](#_Toc205898412)

[2.2 Landasan Teori 8](#_Toc205898413)

[2.2.1 *Internet of Things* (IoT) 9](#_Toc205898414)

[2.2.2 *Application Programming Interface* (API) 11](#_Toc205898415)

[2.2.3 Protokol Komunikasi 11](#_Toc205898416)

[2.2.4 Git 12](#_Toc205898417)

[2.2.5 *Javascript* 12](#_Toc205898418)

[2.2.6 *JavaScript* *Object Notation* (JSON) 13](#_Toc205898419)

[2.2.7 Next.js 13](#_Toc205898420)

[2.2.8 *Database* 14](#_Toc205898421)

[2.2.9 MySQL 14](#_Toc205898422)

[2.2.10 *Message Broker* 15](#_Toc205898423)

[2.2.11 *Black Box Testing* 15](#_Toc205898424)

[2.2.12 JSON *Web Token* (JWT) 15](#_Toc205898425)

[2.2.13 Advanced Encryption Standard (AES) 17](#_Toc205898426)

[2.2.14 *Entropy* pada JWT *Secret* 18](#_Toc205898427)

[2.2.15 Serangan *Brute Force* pada JWT *Secret* 19](#_Toc205898428)

[2.2.16 Klasifikasi Performa Pemuatan Website 20](#_Toc205898429)

[2.2.17 *User Acceptance Testing* (UAT) 21](#_Toc205898430)

[BAB III METODOLOGI PENELITIAN 24](#_Toc205898431)

[3.1 Perencanaan 24](#_Toc205898432)

[3.2 Perancangan 26](#_Toc205898433)

[3.2.1 Arsitektur Sistem 26](#_Toc205898434)

[3.2.2 Diagram *Use Case* 28](#_Toc205898435)

[3.2.3 *Entity Relationship Diagram* (ERD) 28](#_Toc205898436)

[3.2.4 Diagram Alur 30](#_Toc205898437)

[3.2.5 Tampilan UI pada Aplikasi 32](#_Toc205898438)

[3.3 Pembuatan 36](#_Toc205898439)

[3.4 Pengujian 36](#_Toc205898440)

[3.4.1 *Black Box Testing* (Pengujian fungsionalitas) 36](#_Toc205898441)

[3.4.2 Pengujian Sistem Keamanan 37](#_Toc205898442)

[3.4.3 Pengujian Notifikasi 38](#_Toc205898443)

[3.4.4 Pengujian Performa 39](#_Toc205898444)

[3.5 *User Acceptance Testing* (UAT) 39](#_Toc205898445)

[3.6 Evaluasi 42](#_Toc205898446)

[BAB IV HASIL PENGUJIAN DAN ANALISIS 43](#_Toc205898447)

[4.1 Hasil Pengujian Fungsionalitas Aplikasi (*Black Box Testing*) 43](#_Toc205898448)

[4.2 Hasil Pengujian Sistem Keamanan 61](#_Toc205898449)

[4.3 Hasil Pengujian Notifikasi 72](#_Toc205898450)

[4.4 Hasil Pengujian Performa 73](#_Toc205898451)

[4.5 *User Acceptance Testing* (UAT) 81](#_Toc205898452)

[4.6 Evaluasi 83](#_Toc205898453)

[BAB V PENUTUP 86](#_Toc205898454)

[5.1 Kesimpulan 86](#_Toc205898455)

[5.2 Saran 87](#_Toc205898456)

[DAFTAR PUSTAKA 88](#_Toc205898457)

[LAMPIRAN 92](#_Toc205898458)

DAFTAR GAMBAR

[Gambar 2.1 Arsitektur IoT 9](#_Toc205898654)

[Gambar 3.1 Metode Agile-Scrum 24](#_Toc205898655)

[Gambar 3.2 Studi pustaka 25](#_Toc205898656)

[Gambar 3.3 Arsitektur sistem secara keseluruhan 27](#_Toc205898657)

[Gambar 3.4 Arsitektur sistem pada tugas akhir 28](#_Toc205898658)

[Gambar 3.5 *Use Case Diagram* 28](#_Toc205898659)

[Gambar 3.6 *Entity Relationship Diagram* 30](#_Toc205898660)

[Gambar 3.7 Diagram alur autentikasi dan otorisasi pengguna 31](#_Toc205898661)

[Gambar 3.8 Diagram alur penggunaan antarmuka aplikasi 31](#_Toc205898662)

[Gambar 3.9 Diagram alur *node* 32](#_Toc205898663)

[Gambar 4.1 Diagram hasil pengujian *Black Box* 43](#_Toc205898664)

[Gambar 4.2 Daftar panduan pengguna baru 44](#_Toc205898665)

[Gambar 4.3 Login gagal 45](#_Toc205898666)

[Gambar 4.4 Login gagal menggunakan akun Google 46](#_Toc205898667)

[Gambar 4.5 Indikator kolom kosong 46](#_Toc205898668)

[Gambar 4.6 Halaman dashboarddengan filter berdasarkan jumlah data 47](#_Toc205898669)

[Gambar 4.7 Halaman dashboarddengan filter berdasarkan waktu 47](#_Toc205898670)

[Gambar 4.8 Halaman devices 48](#_Toc205898671)

[Gambar 4.9 Formulir tambah device 49](#_Toc205898672)

[Gambar 4.10 Formulir tambah device ketika memilih protokol MQTT 50](#_Toc205898673)

[Gambar 4.11 Indikator berhasil menambahkan perangkat 50](#_Toc205898674)

[Gambar 4.12 Formulir edit device 51](#_Toc205898675)

[Gambar 4.13 Indikator berhasil mengubah data perangkat 51](#_Toc205898676)

[Gambar 4.14 Peringatan persetujuan hapus device 52](#_Toc205898677)

[Gambar 4.15 Indikator berhasil menghapus perangkat 52](#_Toc205898678)

[Gambar 4.16 Halaman datastreams 53](#_Toc205898679)

[Gambar 4.17 Formulir tambah datastream 53](#_Toc205898680)

[Gambar 4.18 Indikator berhasil menambahkan datastream 54](#_Toc205898681)

[Gambar 4.19 Formulir edit *datastream* 54](#_Toc205898682)

[Gambar 4.20 Indikator berhasil mengubah datastream 54](#_Toc205898683)

[Gambar 4.21 Peringatan persetujuan hapus datastream 55](#_Toc205898684)

[Gambar 4.22 Indikator berhasil hapus datastream 55](#_Toc205898685)

[Gambar 4.23 Mode edit halaman dashboard 56](#_Toc205898686)

[Gambar 4.24 Formulir pengaturan grafik (widget) 56](#_Toc205898687)

[Gambar 4.25 Indikator berhasil menyimpan dashboard 57](#_Toc205898688)

[Gambar 4.26 Formulir edit grafik (widget) 58](#_Toc205898689)

[Gambar 4.27 Formulir tambah *alarm* 59](#_Toc205898690)

[Gambar 4.28 Indikator berhasil menambahkan *alarm* 59](#_Toc205898691)

[Gambar 4.29 Indikator berhasil mengubah *alarm* 59](#_Toc205898692)

[Gambar 4.30 Formulis edit *alarm* 60](#_Toc205898693)

[Gambar 4.31 Peringatan persetujuan hapus *alarm* 60](#_Toc205898694)

[Gambar 4.32 Indikator berhasil hapus *alarm* 60](#_Toc205898695)

[Gambar 4.33 *Payload* HTTP pada Wireshark 65](#_Toc205898696)

[Gambar 4.34 *Payload* HTTPS pada Wireshark 65](#_Toc205898697)

[Gambar 4.35 Proses pemeriksaan payload HTTP 66](#_Toc205898698)

[Gambar 4.36 Proses Decode Token JWT pada Website Jwt.io 67](#_Toc205898699)

[Gambar 4.37 Proses pemeriksaan payload MQTT 67](#_Toc205898700)

[Gambar 4.38 Tampilan Menu *Cookie* pada *Browser* Pengguna 70](#_Toc205898701)

[Gambar 4.39 Hasil pengujian performa *website* 74](#_Toc205898702)

[Gambar 4.40 Hasil dari “Network Monitor” halaman *login* 75](#_Toc205898703)

[Gambar 4.41 Hasil dari “*Network Monitor*” halaman *dashboards* 76](#_Toc205898704)

[Gambar 4.42 Hasil dari “Network Monitor” halaman *devices* 77](#_Toc205898705)

[Gambar 4.43 Hasil dari “Network Monitor” halaman *datastreams* 78](#_Toc205898706)

[Gambar 4.44 Hasil dari “Network Monitor” halaman *alarms* 79](#_Toc205898707)

[Gambar 4.45 Hasil pengujian Android Profiler dari Android Studio 80](#_Toc205898708)

[Gambar 4.46 Hasil rata-rata tiap pernyataan 82](#_Toc205898709)

DAFTAR TABEL

[Tabel 2.1 Ringkasan Tinjauan Pustaka 8](#_Toc205898710)

[Tabel 2.2 Tabel Klasifikasi Performa Pemuatan *Website* 20](#_Toc205898711)

[Tabel 2.3 Kategori skor skala Likert pada kuesioner UAT 22](#_Toc205898712)

[Tabel 2.4 Interpretasi persentase tingkat penerimaan pengguna 23](#_Toc205898713)

[Tabel 3.1 Rancangan tampilan antarmuka aplikasi *website* 33](#_Toc205898714)

[Tabel 3.2 Rancangan tampilan antarmuka aplikasi Android 34](#_Toc205898715)

[Tabel 3.3 Rancangan pengujian fungsionalitas aplikasi websitedan Android 36](#_Toc205898716)

[Tabel 3.4 Rancangan pengujian sistem keamanan 38](#_Toc205898717)

[Tabel 3.5 Rancangan pengujian notifikasi 39](#_Toc205898718)

[Tabel 3.6 Rancangan pengujian performa aplikasi *website* 39](#_Toc205898719)

[Tabel 3.7 Daftar pernyataan kuesioner pengujian UAT 40](#_Toc205898720)

[Tabel 4.1 Hasil pengujian sistem keamanan 61](#_Toc205898721)

[Tabel 4.2 Hasil pengujian notifikasi 72](#_Toc205898722)

[Tabel 4.3 Komparasi hasil penelitian 83](#_Toc205898723)

DAFTAR LAMPIRAN

[Lampiran 1 Halaman *Landing Page* 92](#_Toc205898881)

[Lampiran 2 Halaman *Login* 92](#_Toc205898882)

[Lampiran 3 Halaman *Dashboards* 92](#_Toc205898883)

[Lampiran 4 Halaman *Dashboards* Mode *Edit* 93](#_Toc205898884)

[Lampiran 5 Halaman *Dashboards* Kosong 93](#_Toc205898885)

[Lampiran 6 Halaman *Devices* 93](#_Toc205898886)

[Lampiran 7 Halaman *Devices* Kosong 94](#_Toc205898887)

[Lampiran 8 Halaman *Datastreams* 94](#_Toc205898888)

[Lampiran 9 Halaman *Datastreams* Kosong 94](#_Toc205898889)

[Lampiran 10 Halaman *Alarms* 95](#_Toc205898890)

[Lampiran 11 Halaman *Alarms* Kosong 95](#_Toc205898891)

[Lampiran 12 Tampilan Pengaturan Akun 95](#_Toc205898892)

[Lampiran 13 Tampilan Riwayat Notifikasi 96](#_Toc205898893)

[Lampiran 14 Tampilan Fitur Ekspor 96](#_Toc205898894)

[Lampiran 15 Hasil Ekspor CSV Data Perangkat 96](#_Toc205898895)

[Lampiran 16 Hasil Ekspor PDF Data Perangkat 97](#_Toc205898896)

[Lampiran 17 Hasil Ekspor Riwayat Notifikasi 98](#_Toc205898897)

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). Seiring 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 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 air limbah industri 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 lebih dari satu 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 lebih dari satu protokol komunikasi.
3. Menerapkan sistem keamanan menggunakan JWT, AES 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 dan MQTT.
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 pemahaman. 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 METODOLOGI PENELITIAN**

Bab ini membahas mengenai proses perencanaan, perancangan hingga pengujian dari pembuatan 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 pembuatan 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

Pada penyusunan tugas akhir ini, penulis mengkaji berbagai penelitian terdahulu yang berkaitan dengan sistem pemantauan berbasis IoT beserta permasalahan yang menyertainya. Salah satu isu utama yang sering diangkat adalah keamanan protokol komunikasi IoT. Shakdher dkk. (2019) dalam penelitiannya yang berjudul “Security Vulnerabilities in Consumer IoT Applications” menemukan bahwa beberapa perangkat IoT masih mengirimkan data dalam format teks biasa (plaintext), sehingga rawan terhadap serangan Man In The Middle (MITM). Dengan menggunakan aplikasi Wireshark untuk menganalisis lalu lintas data, peneliti menekankan pentingnya penambahan fitur keamanan seperti hashing untuk melindungi data dari serangan siber.

Menanggapi masalah tersebut, Shodiq dkk. (2021) melalui penelitian berjudul “Secure MQTT Authentication and Message Exchange Methods for IoT Constrained Device” mengusulkan mekanisme autentikasi berbasis JWT pada protokol MQTT. Tujuannya adalah untuk mencegah akses ilegal ke jaringan sekaligus meningkatkan keamanan data menggunakan enkripsi XXTEA. Hasil pengujian menunjukkan mekanisme ini efektif melawan serangan brute force dan MITM, serta tetap kompatibel dengan perangkat berdaya rendah. Secara kuantitatif, implementasi mekanisme JWT dengan panjang *secret* yang memiliki entropi sekitar 64 bit memberikan ruang kunci sebesar kemungkinan kombinasi. Dengan asumsi laju serangan *brute force* sebesar tebakan per detik, maka dibutuhkan waktu kurang lebih tahun untuk menebak dengan benar. Nilai ini menunjukkan bahwa tingkat keamanan yang diperoleh tidak hanya bergantung pada panjang kunci, tetapi juga pada distribusi acak karakter penyusunnya.

Penelitian di bidang pemantauan kualitas air juga telah banyak dilakukan. Bogdan dkk. (2023) dalam penelitiannya *“Low-Cost Internet-of-Things Water-Quality Monitoring System for Rural Areas”* mengembangkan sistem berbasis Arduino UNO dengan sensor untuk mengukur suhu, pH, *Total Dissolved Solid* (TDS), dan kekeruhan (*turbidity*). Data dikirimkan melalui Bluetooth ke aplikasi Android untuk pemantauan real-time. Meskipun bermanfaat, sistem ini memiliki keterbatasan jarak komunikasi.

Raman & Martin (2024) melalui penelitian *“IoT-Enabled Water Pollution Detection for Real-Time Monitoring and Pollution Source Identification with MQTT Protocol”* membangun platform IoT berbasis protokol MQTT yang dapat mengirimkan data sensor secara real-time, seperti komposisi kimia, pH, *Dissolved Oxygen* (DO), dan amonia (NH3-N). Sistem ini memudahkan identifikasi sumber pencemaran secara cepat sehingga tindakan pencegahan dapat dilakukan lebih efektif.

Di sisi lain, penelitian terkait pemantauan limbah cair juga berkembang pesat. Salem dkk. (2022) dalam penelitiannya *“An Industrial Cloud-Based IoT System for Real-Time Monitoring and Controlling of Wastewater”* menggunakan ESP8266 untuk memantau pH dan suhu limbah cair, dengan data dikirimkan ke platform ThingSpeak dan dilengkapi notifikasi SMS. Namun, metode ini belum menerapkan keamanan data berbasis *hashing*.

Febriana & Farros (2024) dalam penelitian *“Rancang Bangun Industrial Internet of Things Board Multiguna untuk Monitoring Limbah Cair”* merancang papan IIoT multiguna dengan konektivitas LoRa untuk memantau pH, *Chemical Oxygen Demand* (COD), *Total Suspended Solid* (TSS), NH3-N, dan debit air (*water flow meter*). Data disimpan pada kartu SD dan ditampilkan di Nextion *Display*, serta dapat dipantau melalui ThingSpeak. Meski demikian, sistem ini belum memiliki platform pemantauan berbasis *mobile* dan layanan notifikasi untuk memberi tahu perubahan signifikan pada nilai sensor.

Berdasarkan penelitian-penelitian tersebut, terlihat bahwa sebagian besar sistem yang ada memiliki kemiripan dengan tugas akhir ini, tetapi masih menyisakan kelemahan seperti keterbatasan dukungan protokol komunikasi, kurangnya sistem keamanan data, dan ketiadaan fitur notifikasi. Pada penelitian ini, keunggulan yang ditawarkan adalah penerapan *message broker* untuk integrasi protokol MQTT, penggunaan enkripsi HTTPS, serta *hashing* JWT guna menjaga keamanan dan integritas data. Selain itu, sistem juga dilengkapi notifikasi melalui *browser* dan WhatsApp agar pengguna dapat segera mengetahui adanya anomali pada nilai sensor. Tabel 2.1 merupakan tabel yang berisi ringkasan tinjauan pustaka yang telah dijelaskan di atas.

1. Ringkasan Tinjauan Pustaka

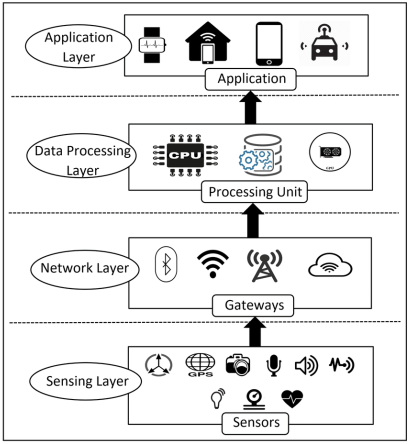
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Judul Penelitian** | **Sensor** | **Antarmuka** | **Protokol** | **Notifikasi** | **Keamanan** |
| (Febriana & Farros, 2024) | COD, pH,  TSS, NH3-N, Debit air | *Website* | HTTP | - | - |
| (Raman & Martin, 2024) | NH3-N, DO, pH, COD | *Website* | MQTT | - | - |
| (Salem dkk., 2022) | Suhu air, pH | *Website* | HTTP | SMS | - |
| (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 | *Built-In* (Bawaan) | JWT, Enkripsi AES, HTTPS |

1. Landasan 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 perangkat 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 *klien* 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 (Shodiq dkk., 2021).

1. *Application Programming Interface* (API)

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 (Mudassir & Mushtaq, 2024). Dalam tugas akhir ini, API digunakan untuk menjembatani komunikasi antara perangkat IoT dengan platform pemantauan. Melalui API, data dari sensor dapat dikirim ke *server* secara *real-time* dan disajikan kepada pengguna dalam bentuk yang informatif dan mudah diakses. (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 dua 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  (Ahmad dkk., 2022). 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).

*Hypertext Transfer Protocol Secure* (HTTPS): merupakan 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 (Shah & Correia, 2021).

1. Git

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*

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 (Loring dkk., 2015).

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

JSON adalah format data ringan yang digunakan untuk pertukaran data antara *server* dan klien (Nurseitov dkk., 2021). 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 (Bourhis dkk., 2020; json.org, 2006).

Format JSON digunakan baik dalam komunikasi API maupun dalam penyusunan *payload* pada JWT. JSON mendukung efisiensi dan kecepatan pertukaran data, yang sangat penting untuk sistem pemantauan *real-time* seperti sistem pemantauan limbah cair. Dalam tugas akhir ini, JSON memfasilitasi integrasi antara mikrokontroler ESP32 sebagai pengirim data dan *server*.

1. Next.js

Next.js adalah *framework open-source* berbasis React yang dirancang untuk membangun aplikasi *web* modern dengan dukungan *server-side rendering* dan *static site generation* (Jartarghar dkk., 2022; 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. (nextjs.org, 2024).

Dalam tugas akhir ini, Next.js digunakan sebagai *framework* pembuatan 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*. *Framework* ini juga kompatibel dengan cukup banyak *library,* salah satunya adalah next-pwa*. Library* next-pwa berfungsi untuk men-*compile* server web Next.js menjadi aplikasi Android berbasis *WebView*, sehingga pengguna dapat memasang versi Android dari sistem pemantauan dengan mudah. (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 (Pavlović dkk., 2021).

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 (Lee, 2022). Selain itu, SQL juga mendukung pengelompokan data, pengurutan, dan penggabungan antar tabel (Pavlović dkk., 2021).

1. MySQL

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 (Pavlović dkk., 2021).

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 *Atomicity*, *Consistency*, *Isolation*, *Durability* (ACID) *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, Mosquitto *broker*, dan Apache Kafka. Dalam Tugas Akhir ini, penulis menggunakan Mosquitto *broker* sebagai *message broker* yang akan dijalankan pada *server*.

1. *Black 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 (Ayuningtyas dkk., 2023; Myers dkk., 2011).

1. JSON *Web Token* (JWT)

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* (Darmawan dkk., 2023). Token JWT terdiri atas tiga komponen utama:

|  |  |  |
| --- | --- | --- |
| 1. | *Header* : | berisi informasi tipe token (*typ*) dan algoritma yang digunakan untuk menandatangani token (*alg*), misalnya HS256 atau RS256. |
| 2. | *Payload* : | memuat klaim (*claims*) yang berisi informasi pengguna atau perangkat, seperti *issuer* (iss), *subject* (sub), *audience* (aud), dan *expiration time* (exp). |
| 3. | *Signature* : | digunakan untuk memverifikasi keaslian token. *Signature* dihasilkan dengan menggabungkan *header* dan *payload* yang telah di-*encode* dalam format Base64URL, kemudian ditandatangani menggunakan algoritma yang sesuai dan kunci rahasia (*secret key*). |

Proses verifikasi JWT di sisi server dilakukan dengan cara menghitung kembali nilai *signature* menggunakan *header* dan *payload* yang diterima, lalu membandingkannya dengan *signature* pada token. Token dinyatakan valid jika hasil perhitungan sama persis dengan *signature* yang dikirimkan (Hosenkhan & Pattanayak, 2021). Secara umum, proses pembentukan JWT dapat ditulis dalam bentuk Persamaan (2.1) (Jwt.io, 2024).

|  |  |  |
| --- | --- | --- |
| *JWT* = *HMACSHA256( base64Url(H)* + "." + *base64Url(P)*, *K)* |  |  |

dengan:

algoritma *hashing*

algoritma *encoding*

*H = header*

*P = payload*

kunci *secret*

Pada Tugas Akhir ini, JWT digunakan untuk mengamankan akses antara pengguna maupun perangkat IoT terhadap sistem. Setiap permintaan ke *server* memerlukan token JWT yang telah ditandatangani dengan *secret key* tertentu. Token juga memiliki masa berlaku (*expiration time*) untuk membatasi risiko penyalahgunaan jika token jatuh ke pihak yang tidak berwenang.

1. Advanced Encryption Standard (AES)

AES merupakan salah satu algoritma kriptografi simetris yang banyak digunakan untuk melindungi data sensitif karena tingkat keamanannya yang tinggi dan efisiensinya dalam proses enkripsi maupun dekripsi. Algoritma AES bekerja dengan membagi data menjadi blok-blok dengan ukuran tertentu dan melakukan serangkaian transformasi yang melibatkan substitusi, permutasi, dan operasi aritmetika berbasis Galois Field GF(2⁸**)**. Penggunaan AES-128-CBC memberikan tingkat keamanan yang memadai apabila kunci rahasia dan IV dipilih secara acak dan tidak dapat diprediksi. Namun, keamanan ini sangat bergantung pada kerahasiaan kunci dan IV. Oleh karena itu, pemilihan panjang kunci yang tepat, serta penyimpanan yang aman menjadi aspek penting dalam penerapan algoritma ini pada sistem.

Pada Tugas Akhir ini digunakan AES dengan panjang kunci 128-bit (*AES-128*) dan mode operasi *Cipher Block Chaining* (CBC). Proses dekripsi AES-128-CBC dapat dijelaskan sebagai kebalikan dari proses enkripsi, di mana setiap blok *ciphertext* (Ci) diubah kembali menjadi *plaintext* (Pi​) menggunakan kunci *secret* (K) dan *Initialization Vector* (IV). Secara umum, proses dekripsi satu blok pada mode CBC dapat dihitung menggunakan Persamaan (2.2).

|  |  |  |
| --- | --- | --- |
|  | ….……………. |  |

dengan:

blok *plaintext* ke-i

blok *ciphertext* ke-i

fungsi dekripsi AES

kunci *secret*

operasi XOR (*exclusive OR)*

blok *ciphertext* sebelumnya (untuk blok pertama digantikan oleh IV)

Pada implementasi Tugas Akhir ini, data terenkripsi disimpan dalam format Base64, kemudian diubah menjadi *buffer* biner. Enkripsi dilakukan dengan menyisipkan IV sepanjang 16 byte pada bagian awal *ciphertext*, sehingga proses dekripsi dapat dilakukan tanpa harus menyimpan IV secara terpisah. *Secret key* diberikan dalam bentuk hexadecimal dan diambil 16 byte pertama untuk memenuhi ukuran kunci AES-128 (128 bit = 16 byte).

1. *Entropy* pada JWT *Secret*

*Entropy* dalam kriptografi adalah ukuran ketidakpastian atau kerandoman dalam sebuah string karakter yang digunakan untuk membentuk suatu kata (Zhang dkk., 2022). *Secret key* dalam JWT berfungsi untuk menghasilkan tanda tangan digital (*signature*) yang menjamin integritas dan autentikasi *payload*. Semakin tinggi nilai entropi, semakin sulit untuk menebak atau menduplikasi kunci tersebut, sehingga sistem menjadi lebih aman dari serangan. *Entropy* diukur dalam satuan bit dan dapat dihitung menggunakan Persamaan (2.3).

|  |  |  |
| --- | --- | --- |
|  | ………….…………. |  |

di mana:

*entropy* dalam bit

panjang *secret*

jumlah kemungkinan karakter unik dalam himpunan karakter

Sebagai contoh, untuk kata dengan panjang 16 karakter dan karakter set sebanyak 94 (ASCII *printable*), maka nilai entropinya adalah sekitar 104.8 bit. Nilai ini menunjukkan seberapa besar jumlah kombinasi kemungkinan yang harus ditebak oleh pihak yang mencoba membobol sistem secara *brute force*. Menurut Zhang dkk. (2022), entropy yang memadai dalam kunci kriptografi merupakan syarat untuk mencapai perfect secrecy dalam sistem keamanan, dengan standar industri umumnya menggunakan minimal 128 bit entropy untuk sistem autentikasi yang aman. Temuan ini juga diperkuat oleh penelitian Saputra, Sugiarti, Junianto, & Suhartono (2025) yang menyatakan bahwa nilai entropi sangat berperan dalam menentukan waktu yang dibutuhkan dalam serangan *brute force*, serta secara langsung memengaruhi kekuatan suatu sandi atau kunci dalam sistem keamanan.

1. Serangan *Brute Force* pada JWT *Secret*

*Brute force* merupakan salah satu metode serangan kriptografi yang dilakukan dengan mencoba seluruh kemungkinan kombinasi dari sebuah kunci atau secret hingga ditemukan yang sesuai (Saputra, dkk., 2025). Dalam konteks JWT, *brute force* digunakan untuk menebak nilai *secret* yang digunakan dalam proses penandatanganan token, dengan mencocokkan hasil tanda tangan terhadap token yang valid. Jumlah kemungkinan kombinasi kata dihitung dengan Persamaan (2.4) dan estimasi waktu yang dibutuhkan untuk menyelesaikan proses *brute force* juga dapat dihitung menggunakan Persamaan (2.5).

|  |  |  |
| --- | --- | --- |
|  | ………….……………… |  |

dengan:

jumlah total kemungkinan kombinasi

panjang *secret*

jumlah kemungkinan karakter unik dalam himpunan karakter

|  |  |  |
| --- | --- | --- |
|  | ……………………. |  |

dengan:

jumlah tebakan yang dapat dilakukan per detik

Sebagai contoh, apabila sebuah secret terdiri dari 6 karakter huruf kecil (N = 26), maka jumlah kemungkinan kombinasi adalah sekitar 308 juta. Jika proses *brute force* dilakukan dengan kecepatan satu juta tebakan per detik, maka waktu yang diperlukan untuk menjangkau seluruh kemungkinan adalah sekitar 309 detik atau sekitar 5 menit. Penelitian yang dilakukan oleh Saputra, dkk. (2025) menunjukkan bahwa semakin rendah kompleksitas suatu kunci atau *secret*, maka semakin cepat proses *brute force* dapat dilakukan. Oleh karena itu, pemilihan *secret* dengan panjang karakter yang cukup dan keragaman karakter set yang tinggi menjadi langkah penting untuk memperkuat keamanan sistem terhadap serangan semacam ini.

1. Klasifikasi Performa Pemuatan Website

Waktu pemuatan halaman (*page load time*) merupakan salah satu metrik utama dalam mengevaluasi performa sebuah situs *web*. Metrik ini menggambarkan lamanya waktu yang dibutuhkan oleh *browser* untuk menampilkan seluruh konten halaman secara utuh hingga siap digunakan oleh pengguna akhir. Kinerja waktu pemuatan yang baik berkontribusi langsung terhadap peningkatan pengalaman pengguna (*user experience*), kepuasan pengunjung, serta tingkat retensi dan konversi (Google, 2023). Guna memastikan pengalaman akses yang efisien dan responsif, sejumlah pedoman telah dikembangkan oleh Google untuk menilai kualitas waktu pemuatan halaman secara objektif. Adapun kategori tersebut disusun berdasarkan total waktu hingga halaman selesai dimuat, yang dibagi menjadi empat tingkatan, sebagaimana disajikan pada Tabel berikut.

1. Tabel Klasifikasi Performa Pemuatan *Website*

|  |  |  |
| --- | --- | --- |
| **Kategori** | **Waktu Pemuatan (Finish)** | **Keterangan** |
| Sangat Baik | < 1 detik | Hampir tidak terasa oleh pengguna. Memberikan pengalaman terbaik. |
| Baik | 1 – 2,5 detik | Masih dalam batas optimal dan responsif. |
| Cukup | 2,5 – 4 detik | Performa mulai menurun, pengguna mulai menyadari keterlambatan. |
| Buruk | > 4 detik | Berisiko tinggi menyebabkan *bounce rate* meningkat. |

Google juga memperkenalkan beberapa terminologi penting terkait proses pemuatan halaman, antara lain:

|  |  |  |
| --- | --- | --- |
| 1. | *DOMContentLoad* : | Waktu ketika dokumen HTML telah sepenuhnya di-*parse* dan struktur DOM tersedia, namun belum termasuk *file* eksternal seperti CSS, JavaScript, dan gambar. |
| 2. | *Load* : | Menunjukkan bahwa seluruh elemen eksternal telah dimuat secara lengkap. |
| 3. | *Finish* : | Mencerminkan total waktu hingga tidak ada lagi permintaan jaringan aktif, dan halaman sepenuhnya tenang secara trafik. |

Pemahaman terhadap metrik-metrik tersebut memungkinkan pengembang untuk mengidentifikasi titik-titik kritis dalam siklus pemuatan halaman dan menentukan strategi optimalisasi yang tepat. Kinerja pemuatan yang buruk dapat meningkatkan potensi pengguna meninggalkan situs sebelum halaman selesai ditampilkan, sebuah fenomena yang dikenal sebagai *bounce* (Aladwani, 2020; Google, 2023). Dengan demikian, waktu pemuatan bukan hanya persoalan teknis, tetapi juga elemen strategis yang memengaruhi efektivitas interaksi pengguna terhadap sistem pemantauan.

1. *User Acceptance Testing* (UAT)

UAT merupakan tahapan akhir dalam proses pengembangan sistem informasi sebelum sistem tersebut diterapkan secara penuh oleh pengguna akhir. UAT bertujuan untuk memastikan bahwa sistem telah memenuhi kebutuhan, ekspektasi, dan spesifikasi yang telah disepakati oleh pengguna (Marnewick, 2016). Pada tahap ini, pengguna yang mewakili kelompok target dari sistem melakukan serangkaian pengujian untuk menilai kesesuaian fungsionalitas sistem terhadap skenario penggunaan riil.

UAT sangat penting dalam siklus hidup pengembangan perangkat lunak karena dapat mengungkapkan ketidaksesuaian antara desain sistem dan kebutuhan aktual pengguna, yang tidak selalu terdeteksi pada tahap pengujian teknis seperti *unit testing atau integration testing* (Sabherwal dkk., 2019). Selain itu, hasil UAT juga menjadi dasar evaluasi terhadap kesiapan sistem untuk diluncurkan secara operasional. Metode yang umum digunakan dalam pelaksanaan UAT adalah penyebaran kuesioner kepada pengguna untuk memperoleh umpan balik terhadap pengalaman pengguna atau *user experience* (*UX*) dan tingkat kepuasan terhadap sistem. Salah satu pendekatan yang sering digunakan dalam evaluasi kuesioner adalah Skala Likert, yang memungkinkan responden memberikan penilaian berdasarkan tingkat persetujuan mereka terhadap sejumlah pernyataan (Joshi dkk., 2015). Skala ini biasanya terdiri dari 4 hingga 7 poin, yang mencerminkan rentang dari sangat tidak setuju hingga sangat setuju. Adapun skala yang digunakan dalam penelitian ini terdiri dari empat tingkat penilaian, seperti ditunjukkan pada Tabel berikut:

1. Kategori skor skala Likert pada kuesioner UAT

|  |  |  |
| --- | --- | --- |
| **Skor** | **Kategori** | **Interpretasi** |
| 1 | Sangat Tidak Setuju | Responden sangat tidak menyetujui pernyataan yang diberikan |
| 2 | Tidak Setuju | Responden tidak menyetujui pernyataan yang diberikan |
| 3 | Setuju | Responden menyetujui pernyataan yang diberikan |
| 4 | Sangat Setuju | Responden sangat menyetujui pernyataan yang diberikan |

Untuk mengetahui rata-rata penerimaan pengguna, dilakukan pengolahan data kuesioner dengan perhitungan nilai rata-rata setiap item pernyataan menggunakan Persamaan (2.6) (Yaguana, 2023).

|  |  |  |
| --- | --- | --- |
|  | …………………………. |  |

dengan:

nilai rata-rata

skor masing-masing responden

jumlah responden

Tingkat penerimaan pengguna terhadap sistem juga dapat dihitung dengan mengonversi nilai rata-rata menjadi persentase, menggunakan rumus:

|  |  |  |
| --- | --- | --- |
|  | …………. |  |

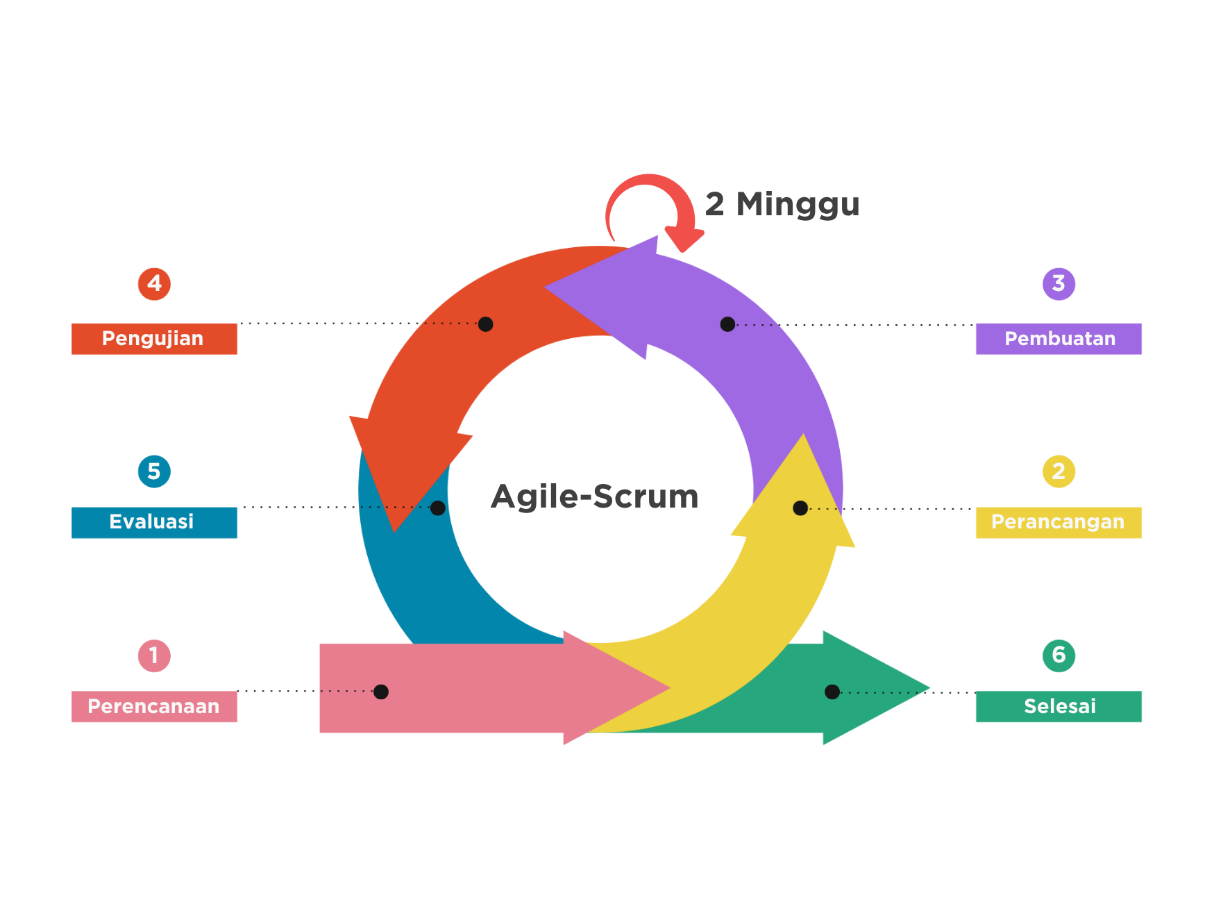
Skor maksimal pada skala *Likert* empat poin adalah 4, maka nilai persentase yang mendekati 100% menunjukkan tingkat penerimaan yang tinggi. Untuk memberikan acuan dalam menafsirkan tingkat penerimaan pengguna berdasarkan nilai persentase tersebut, berikut disajikan Tabel 2.4 yang menunjukkan kategori interpretasi tingkat penerimaan sistem berdasarkan rentang persentase.

1. Interpretasi persentase tingkat penerimaan pengguna

|  |  |
| --- | --- |
| **Persentase** | **Interpretasi** |
| 81% – 100% | Sangat Baik (Diterima) |
| 61% – 80% | Baik |
| 41% – 60% | Cukup |
| 21% – 40% | Kurang |
| 0% – 20% | Tidak Layak |

BAB III  
METODOLOGI PENELITIAN

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 pada pengembangan 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 (Avancha dkk., 2024). Tahapan-tahapan metode penelitian dalam tugas akhir ini dapat dilihat pada Gambar 3.1.

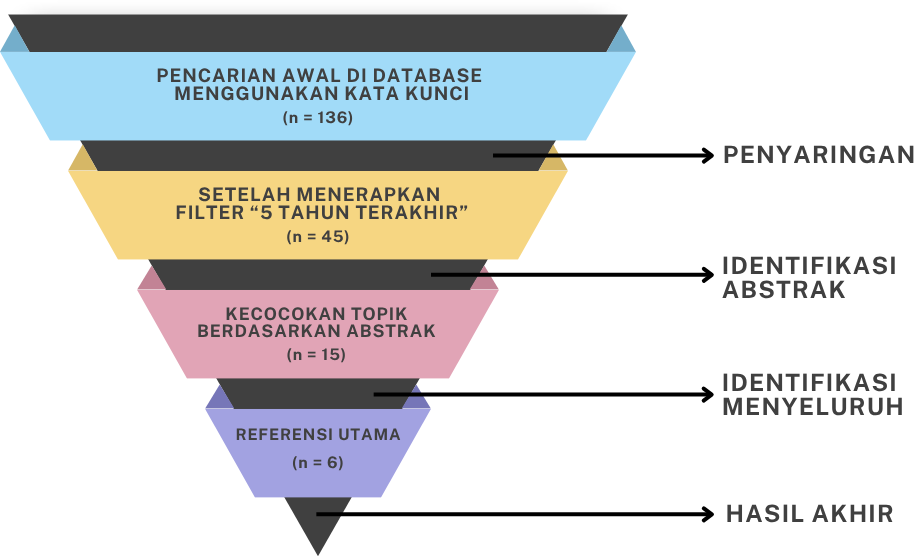


1. Metode Agile-Scrum

(Sumber: Dokumentasi Pribadi)

1. Perencanaan

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 (Avancha dkk., 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. Hasil studi pustaka dapat dilihat pada Gambar 3.2.



1. Studi pustaka

(Sumber: Dokumentasi pribadi)

Proses pencarian difokuskan pada artikel yang dipublikasikan dalam rentang waktu 2020 hingga 2024 dengan menggunakan beberapa kata kunci seperti “*IoT* *Platform*”, “*Wastewater* *Monitoring*”, dan “JSON *Web* Token”. Dari hasil pencarian, terkumpul sebanyak 136 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 6 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:

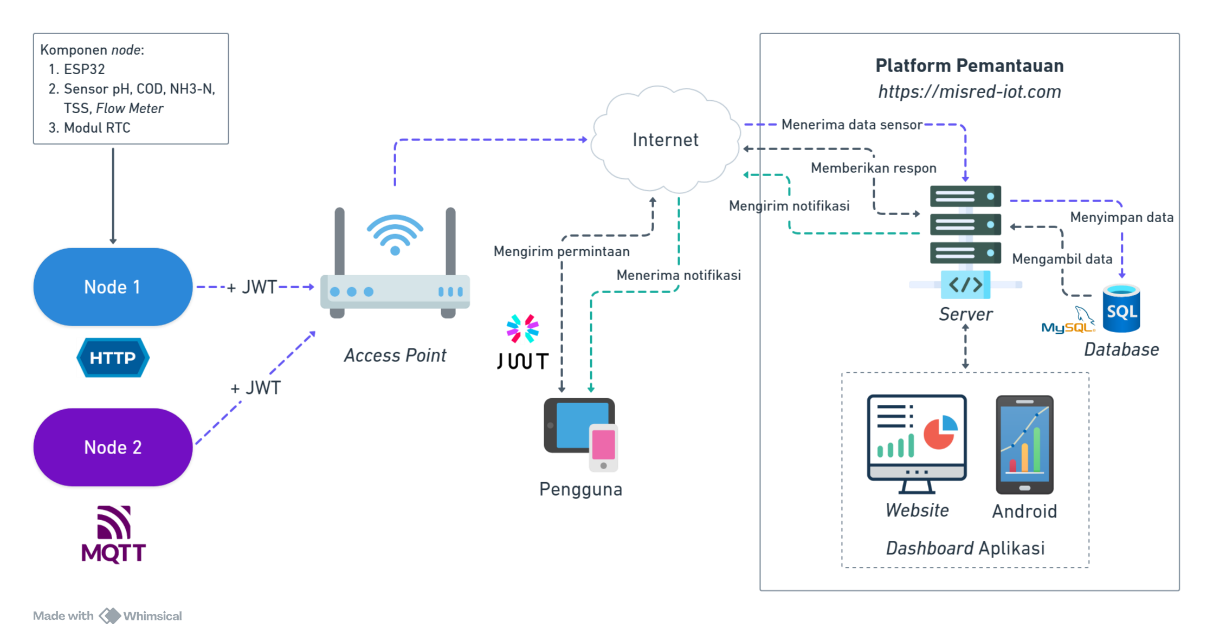
* 1. Visual Studio Code.
  2. Android Studio.
  3. *Browser.*
  4. Figma.
  5. Wireshark

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, rancangan diagram alur, dan rancangan tampilan antarmuka pengguna atau *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 dua buah *node* dengan tiga jenis protokol yang akan digunakan untuk berkomunikasi dengan *server. Node* merupakan perangkat IoT yang bertugas mengumpulkan data (Rana dkk., 2023). *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. 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 dapat dilihat pada Gambar 3.3



1. Arsitektur sistem secara keseluruhan

(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 3.4 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.

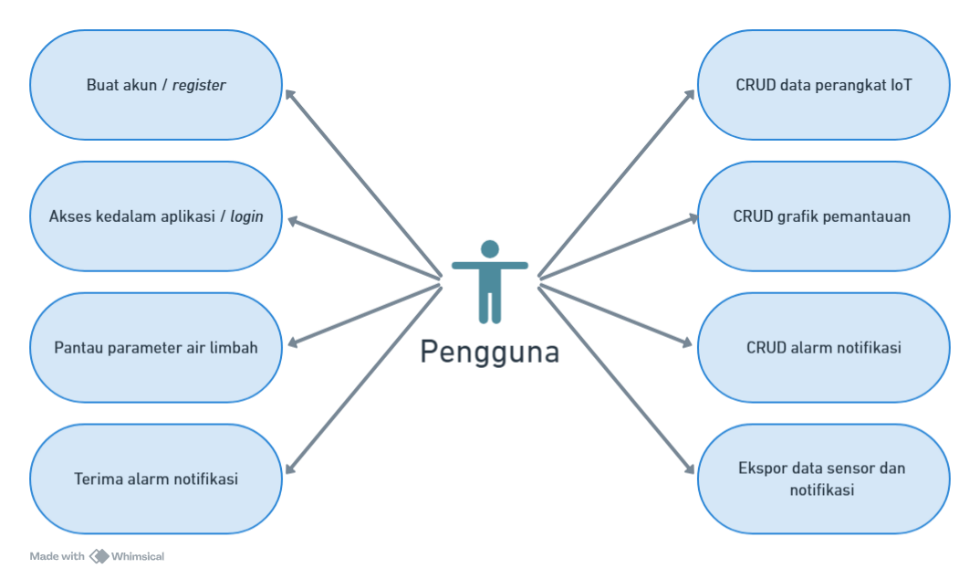


1. Arsitektur sistem pada tugas akhir

(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, grafik pemantauan parameter air limbah, serta alarm notifikasi. Pengguna juga dapat menerima notifikasi pada *browser* atau nomor WhatsApp saat ada alarm yang terpicu. Selain itu, pengguna juga dapat mengekspor data sensor dan notifikasi. Gambar 3.5 menunjukkan tentang diagram ini.



1. *Use Case Diagram*

(Sumber: Dokumentasi Pribadi)

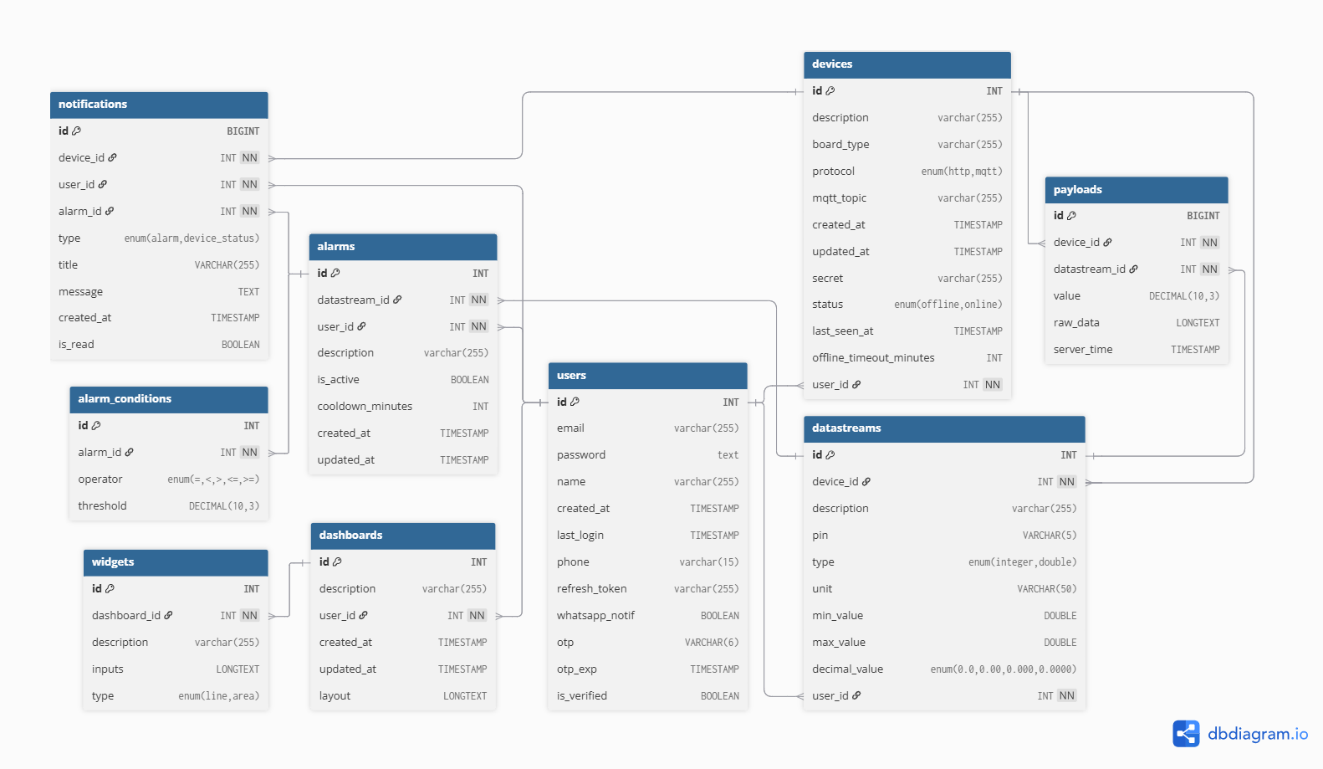
1. *Entity Relationship Diagram* (ERD)

Pada tugas akhir ini, digunakan sembilan tabel basis data yang dirancang untuk mendukung sistem pemantauan IoT yang komprehensif. Tabel *users* berfungsi menyimpan kredensial pengguna aplikasi termasuk *email*, *password*, nama, OTP, dan token JWT untuk sistem autentikasi yang aman. Tabel *dashboards* menyimpan konfigurasi dashboard yang dibuat pengguna, termasuk tata letak grafik, nama dashboard, dan pengaturan tampilan untuk visualisasi data yang optimal.

Tabel *devices* menyimpan informasi lengkap perangkat IoT yang terdaftar pada platform, mencakup deskripsi perangkat, jenis *board* (ESP32/Arduino), protokol komunikasi (MQTT/HTTP), MQTT topic, status aktif, dan JWT secret unik untuk autentikasi komunikasi *node*-*server*. Tabel *datastreams* berfungsi sebagai konfigurasi *virtual* *pin* yang mendefinisikan aliran data dari setiap perangkat, menyimpan tipe data sensor, unit pengukuran, nilai maksimum-minimum, dan deskripsi datastream untuk fleksibilitas pemetaan data sensor. Tabel *payloads* merupakan tabel penyimpanan utama untuk seluruh data sensor yang dikirimkan perangkat IoT, menyimpan nilai sensor, *timestamp* pengiriman, dan referensi datastream untuk analisis historis dan pemantauan *real*-*time*.

Tabel *widgets* menyimpan konfigurasi visualisasi *dashboard* pengguna, termasuk informasi *dashboard*, *datastream* yang digunakan, tipe grafik, dan tata letak grafik pada *dashboard* untuk tampilan pemantauan. Tabel *alarms* digunakan untuk menyimpan pengaturan sistem peringatan yang dibuat pengguna, mencakup deskripsi alarm, status aktif, referensi datastream yang dipantau, dan kondisi pemicu untuk integrasi dengan sistem notifikasi real-time. Tabel *alarms***\_***conditions* berfungsi sebagai tabel yang menyimpan kondisi spesifik setiap alarm, termasuk operator perbandingan (>, <, =, >=, <=), nilai ambang batas, dan logika kombinasi antar kondisi untuk pembuatan alarm yang fleksibel. Tabel *notifications* menyimpan riwayat lengkap notifikasi yang dikirimkan kepada pengguna, mencakup pesan notifikasi *browser* dan WhatsApp, status pembacaan, waktu pengiriman, dan referensi ke perangkat serta pengguna terkait.

Relasi antar tabel menunjukkan struktur hierarkis dimana setiap pengguna (*users*) dapat memiliki satu atau banyak perangkat (*devices*) dengan relasi *one-to-many.* Setiap perangkat kemudian dapat memiliki banyak *datastreams*, *widgets*, *payloads*, *alarms*, *notifications*, dan *dashboards*, membentuk struktur yang memungkinkan skalabilitas sistem. Tabel *alarms\_conditions* memiliki relasi *many-to-one* dengan tabel *alarms*, memungkinkan satu alarm memiliki beberapa kondisi pemicu sekaligus untuk logika yang kompleks. Struktur relasi ini dirancang untuk memberikan fleksibilitas maksimal dalam pengelolaan data IoT, mulai dari pengumpulan data sensor, visualisasi *dashboard*, hingga sistem notifikasi otomatis yang dapat disesuaikan dengan kebutuhan spesifik setiap pengguna. Gambar 3.6 menunjukkan diagram hubungan entitas dari tugas akhir kami.

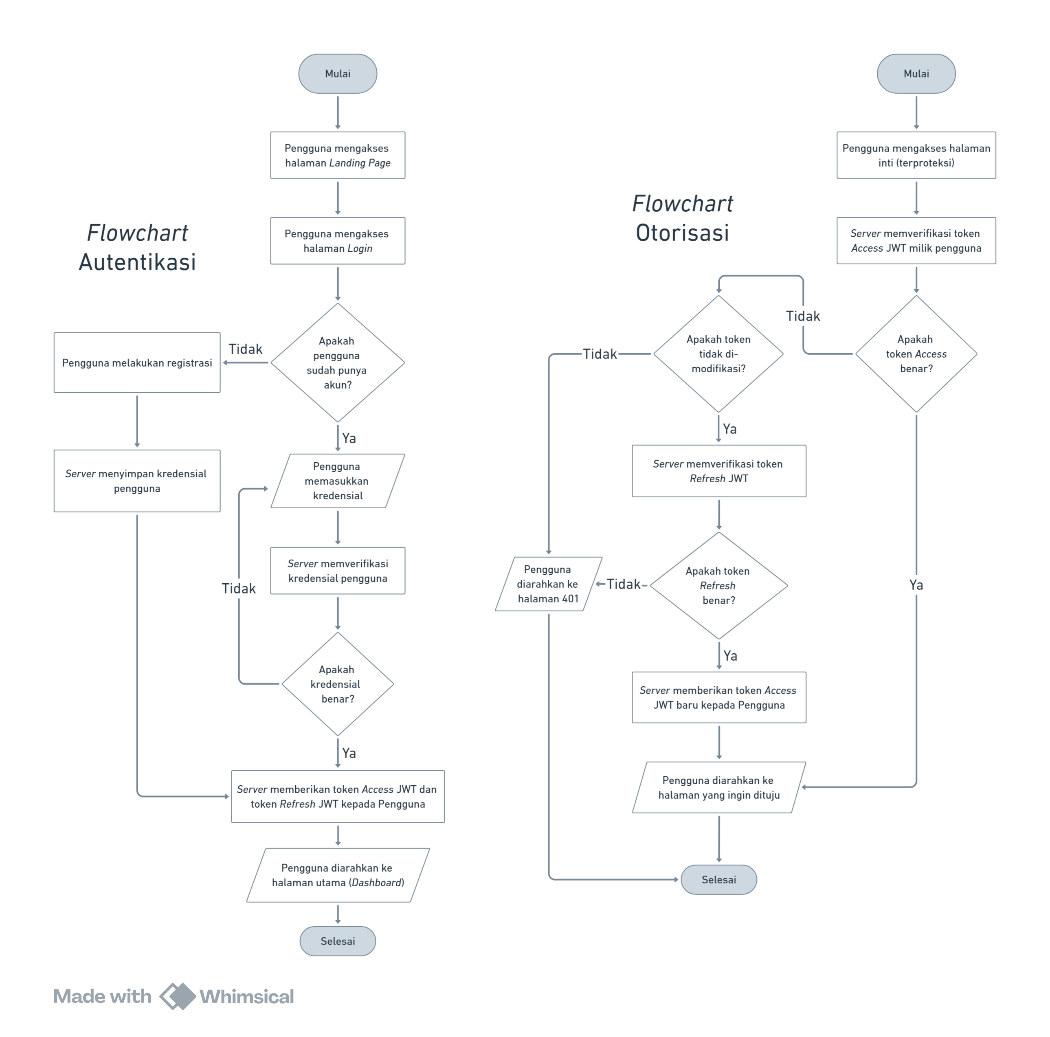


1. *Entity Relationship Diagram*

(Sumber: Dokumentasi Pribadi)

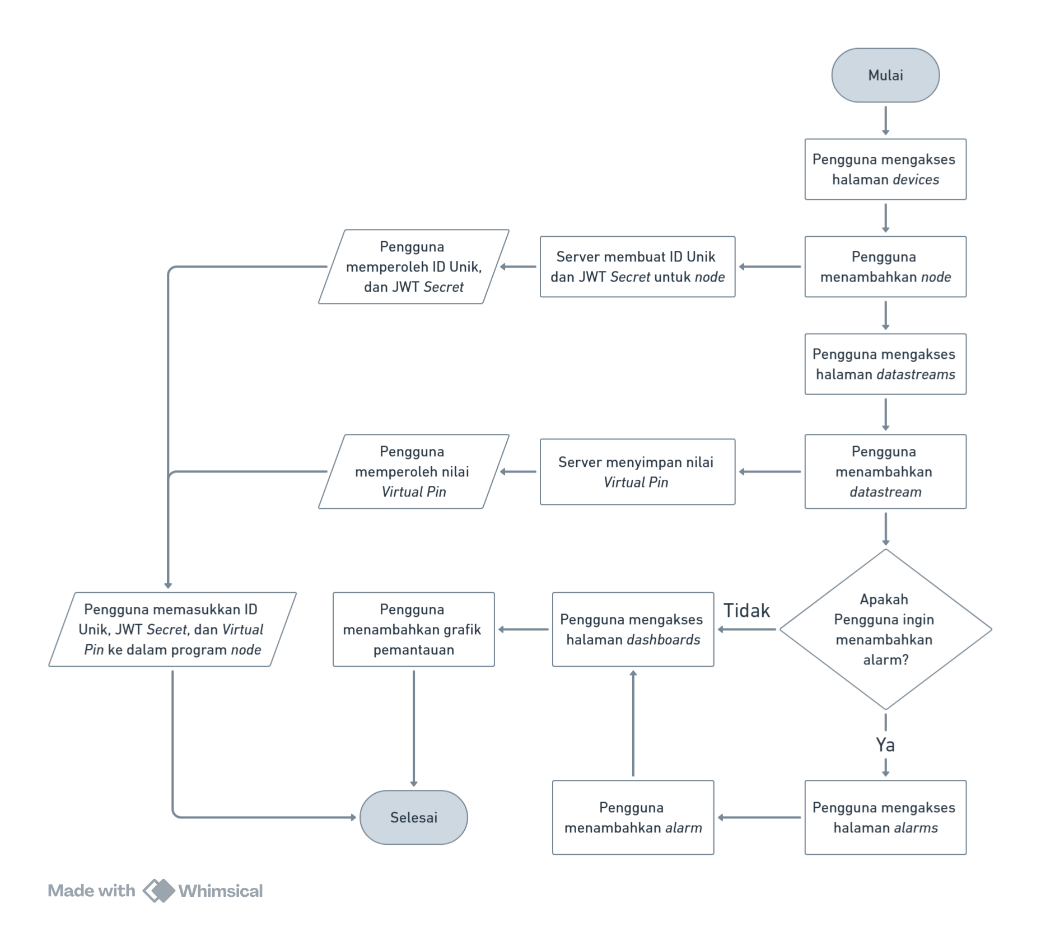
1. Diagram Alur

Berdasarkan arsitektur sistem dari tugas akhir, dibuatlah diagram alur untuk desain cara kerja dari sistem pemantauan. Adapun diagram alur dari sistem terbagi menjadi tiga, yaitu diagram alur autentikasi dan otorisasi pengguna, diagram alur penggunaan antarmuka aplikasi, dan diagram alur *node*. Diagram alur autentikasi dan otorisasi pengguna menggambarkan proses keamanan akses sistem, diagram ini dapat dilihat pada Gambar 3.7. Diagram alur penggunaan antarmuka aplikasi menunjukkan navigasi dan fitur sistem pemantauan, diagram ini dapat dilihat pada Gambar 3.8. Diagram alur *node* menjelaskan proses pengumpulan dan pengiriman data sensor dari perangkat IoT ke *server*, diagram ini dapat dilihat pada Gambar 3.9. Ketiga diagram alur ini memberikan gambaran komprehensif tentang alur kerja sistem pemantauan IoT mulai dari level pengguna hingga level perangkat keras.



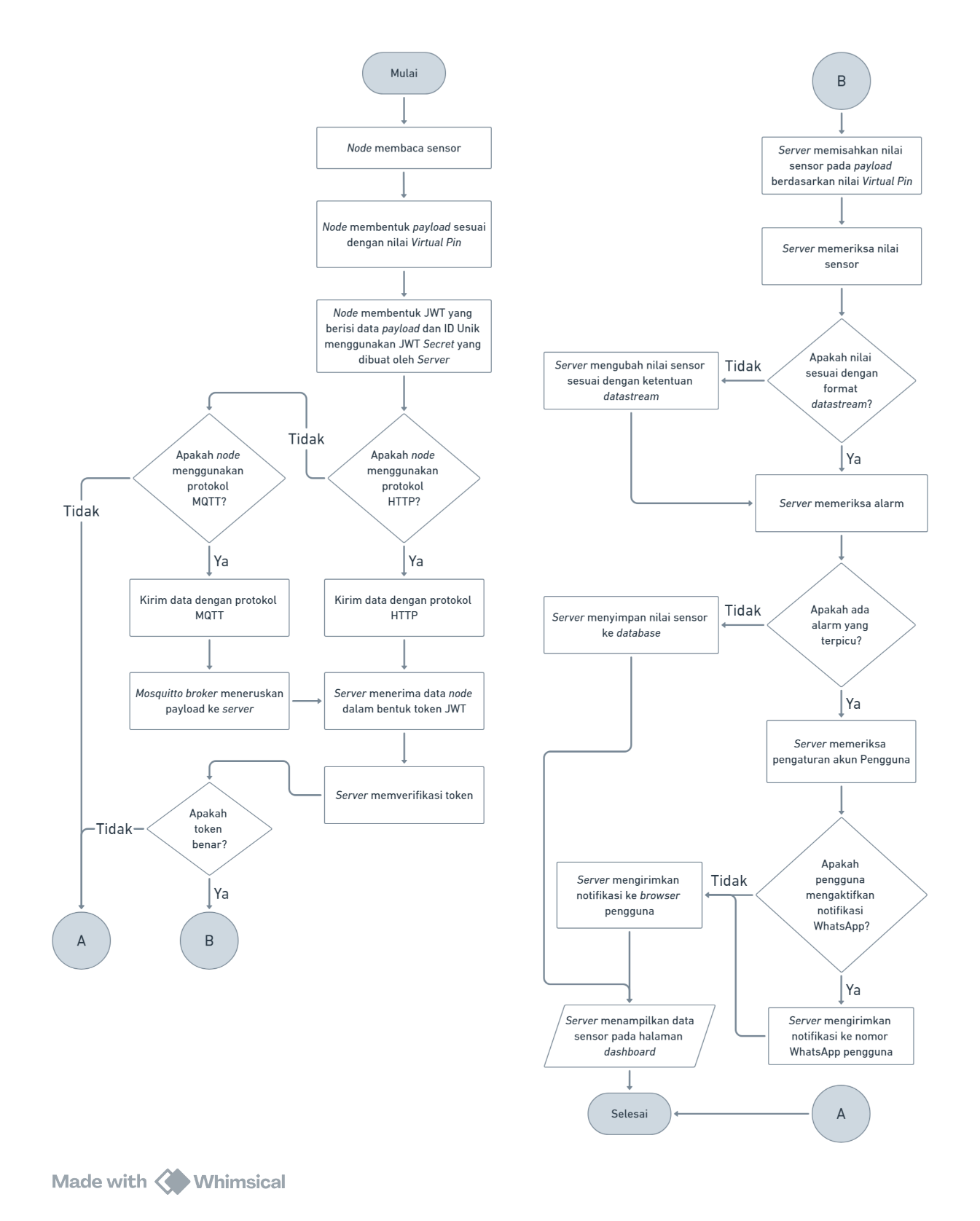
1. Diagram alur autentikasi dan otorisasi pengguna

(Sumber: Dokumentasi Pribadi)



1. Diagram alur penggunaan antarmuka aplikasi

(Sumber: Dokumentasi pribadi)



1. Diagram alur *node*

(Sumber: Dokumentasi pribadi)

1. Tampilan UI pada Aplikasi

Rancangan UI 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 3.6.
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 websitedan Android

| **No** | **Deskripsi pengujian** | **Parameter yang diharapkan** | **Kesesuaian hasil pengujian** |
| --- | --- | --- | --- |
| 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** |
| --- | --- | --- |
| 1. | Mengakses halaman aplikasi menggunakan token JWT yang valid. |  |
| 2. | Mengakses halaman aplikasi menggunakan token JWT yang dimodifikasi. |  |
| 3. | Mengakses halaman aplikasi tanpa disertai *refresh token* JWT |  |
| 4. | Node mengirim data sensor menggunakan token yang benar |  |
| 5. | Node mengirim data sensor menggunakan token yang kadaluarsa |  |
| 6. | Node mengirim data sensor menggunakan token yang dimodifikasi |  |

Selain pengujian-pengujian tersebut, dilakukan pula pengujian tambahan yang bertujuan untuk mengevaluasi kekuatan secret key yang digunakan dalam proses penandatanganan token JWT. Aspek yang diuji meliputi analisis entropi secret JWT untuk menilai tingkat kerandoman serta kompleksitasnya secara matematis, dan simulasi serangan brute force terhadap token JWT untuk mengukur ketahanan secret terhadap eksploitasi berbasis enumerasi kunci. Pengujian ini memberikan pemahaman kuantitatif dan praktis mengenai seberapa aman secret yang digunakan terhadap upaya pemalsuan token. Pengukuran entropi dilakukan menggunakan Persamaan (2.3). Sementara itu, jumlah kemungkinan kombinasi karakter dalam brute force dihitung dengan Persamaan (2.4) dan estimasi waktu brute force dihitung menggunakan Persamaan (2.5).

1. Pengujian Notifikasi

Pengujian selanjutnya adalah pengujian notifikasi. Pengujian ini dilakukan untuk menguji keberhasilan aplikasi dalam memunculkan notifikasi pada *browser* dan WhatsApp. Tabel 3.5 menunjukkan rancangan pengujian ini.

1. Rancangan pengujian notifikasi

|  |  |  |
| --- | --- | --- |
| **No.** | **Deskripsi Pengujian** | **Gambar** |
| 1. | Menampilkan pesan notifikasi pada nomor *browser* saat ada alarm yang terpicu atau ada peringatan perangkat yang *offline* |  |
| 2. | Menampilkan pesan notifikasi pada nomor WhatsAppsaat ada alarm yang terpicu atau ada peringatan perangkat yang *offline* |  |

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 3.6 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 dan memori saat aplikasi dijalankan.

1. *User Acceptance Testing* (UAT)

UAT dilakukan untuk memperoleh umpan balik langsung dari pengguna terhadap kualitas antarmuka dan pengalaman pengguna (UI/UX) dari sistem berbasis web yang telah dikembangkan. Dalam pengujian ini, penyusun menggunakan pendekatan kualitatif dengan menyebarkan kuesioner yang terdiri dari beberapa pernyataan yang relevan dengan aspek UI/UX kepada sejumlah responden terpilih. Setiap responden memberikan tanggapan terhadap setiap pernyataan menggunakan Skala Likert empat poin, yaitu: sangat tidak setuju, tidak setuju, setuju, dan sangat setuju.

Data yang diperoleh dari hasil kuesioner kemudian diolah untuk memperoleh nilai rata-rata dari setiap pernyataan menggunakan Persamaan (2.1). Selanjutnya, nilai rata-rata tersebut dikonversi ke dalam bentuk persentase penerimaan dengan menggunakan Persamaan (2.2). Dengan menggunakan dua rumus tersebut, hasil dari kuisioner dapat dianalisis untuk mengetahui sejauh mana sistem diterima oleh pengguna dari sisi kenyamanan visual, kemudahan navigasi, kecepatan akses, kejelasan informasi, dan elemen-elemen UX lainnya.

Pernyataan-pernyataan yang digunakan dalam kuesioner disusun berdasarkan prinsip evaluasi antarmuka dan pengalaman pengguna yang mencakup aspek visual, navigasi, kecepatan akses, serta keterbacaan informasi. Daftar lengkap pernyataan yang diajukan kepada responden disajikan pada Tabel 3.7.

1. Daftar pernyataan kuesioner pengujian UAT

| **No.** | **Pernyataan** | **Aspek** |
| --- | --- | --- |
| 1. | Data grafik yang ditampilkan secara visual mudah dipahami. | Visualisasi dan Antarmuka |
| 2. | Penggunaan warna pada grafik (seperti merah dan hijau) sudah cukup jelas untuk membedakan perangkat sensor. | Visualisasi dan Antarmuka |
| 3. | Keterangan pada grafik (judul, label sumbu, dan legenda) mudah dimengerti oleh pengguna. | Visualisasi dan Antarmuka |
| 4. | Antarmuka halaman *Devices* terlihat rapi dan mudah dipahami. | Visualisasi dan Antarmuka |
| 5. | Antarmuka halaman *Datastreams* terlihat rapi dan mudah dipahami. | Visualisasi dan Antarmuka |
| 6. | Antarmuka halaman *Alarms*terlihat rapi dan mudah dipahami. | Visualisasi dan Antarmuka |
| 7. | Riwayat notifikasi dapat diakses dengan mudah oleh pengguna. | Navigasi dan Akses Informasi |
| 8. | Pengguna dapat dengan mudah beralih antara mode gelap dan terang. | Visualisasi dan Antarmuka |
| 9. | Mode gelap memiliki kontras warna yang baik dan nyaman untuk dilihat. | Visualisasi dan Antarmuka |
| 10. | Teks dan ikon tetap jelas terlihat dalam kedua mode (gelap maupun terang). | Visualisasi dan Antarmuka |
| 11. | Fitur filter data bekerja dengan baik dan mudah digunakan. | Fungsionalitas dan Interaksi |
| 12. | Pengaturan widget saat menambahkan grafik baru (seperti mengisi nama dan memilih datastream) mudah dipahami. | Fungsionalitas dan Interaksi |
| 13. | Perubahan tata letak dan ukuran grafik terasa fleksibel dan responsif. | Fungsionalitas dan Interaksi |
| 14. | Proses ekspor berjalan lancar tanpa hambatan teknis. | Fungsionalitas dan Interaksi |
| 15. | Opsi “Sesuai Filter” dan “Semua Data” membantu dalam menyesuaikan kebutuhan ekspor. | Fungsionalitas dan Interaksi |
| 16. | Tombol “Edit” dan “Tambah Dashboard” mudah ditemukan pada halaman. | Navigasi dan Akses Informasi |
| 17. | Fungsi pencarian *device* mudah digunakan dan efektif. | Navigasi dan Akses Informasi |
| 18. | Pengguna dapat memahami daftar *device* tanpa perlu penjelasan tambahan. | Navigasi dan Akses Informasi |
| 19. | Fungsi pencarian *datastream*mudah digunakan dan efektif. | Navigasi dan Akses Informasi |
| 20. | Pengguna dapat memahami daftar *datastream*tanpa perlu penjelasan tambahan. | Navigasi dan Akses Informasi |
| 21. | Fungsi pencarian *alarm*mudah digunakan dan efektif | Navigasi dan Akses Informasi |
| 22. | Pengguna dapat memahami daftar *alarm*tanpa perlu penjelasan tambahan. | Navigasi dan Akses Informasi |
| 23. | Sistem memberikan umpan balik yang cukup setelah perubahan disimpan. | Umpan Balik Sistem |
| 24. | Sistem memberikan umpan balik yang cukup setelah *device* ditambahkan. | Umpan Balik Sistem |
| 25. | Sistem memberikan umpan balik yang cukup saat *device* akan dihapus. | Umpan Balik Sistem |
| 26. | Sistem memberikan umpan balik yang cukup setelah *datastream*ditambahkan. | Umpan Balik Sistem |
| 27. | Sistem memberikan umpan balik yang cukup saat *datastream*akan dihapus. | Umpan Balik Sistem |
| 28. | Sistem memberikan umpan balik yang cukup setelah *alarm*ditambahkan. | Umpan Balik Sistem |
| 29. | Sistem memberikan umpan balik yang cukup saat *alarm*akan dihapus. | Umpan Balik Sistem |
| 30. | Formulir penambahan *device* memiliki label dan instruksi yang jelas. | Formulir dan Kejelasan Instruksi |
| 31. | Formulir penambahan *datastream*memiliki label dan instruksi yang jelas. | Formulir dan Kejelasan Instruksi |
| 32. | Formulir penambahan *alarm*memiliki label dan instruksi yang jelas. | Formulir dan Kejelasan Instruksi |
| 33. | Notifikasi muncul dengan jelas saat alarm terpicu. | Notifikasi dan Respons Sistem |
| 34. | Sistem memberikan informasi notifikasi yang mudah dipahami. | Notifikasi dan Respons Sistem |

Tabel 3.7 menunjukkan bahwa setiap pernyataan telah dirancang untuk menggambarkan persepsi pengguna terhadap elemen-elemen utama dari UI/UX sistem. Dengan menggunakan Skala Likert, responden dapat mengungkapkan tingkat kesetujuan mereka terhadap setiap pernyataan tersebut.

1. Evaluasi

Pada tahap ini dilakukan proses *debugging* dan perilisan, kemudian dilanjutkan dengan pembahasan hasil penelitian*. 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.com/. Adapun aplikasi Android dapat dipasang pada perangkat Android melalui implementasi Android WebView, dengan cara menekan menu ”Tambahkan ke Layar Utama” pada *browser*.

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 skenario. Pada pengujian ini hanya berfokus pada fungsionalitas dan keluaran aplikasi. Hasil pengujian fungsionalitas dapat dilihat dengan grafik pada Gambar 4.1.

1. Diagram hasil pengujian *Black Box*

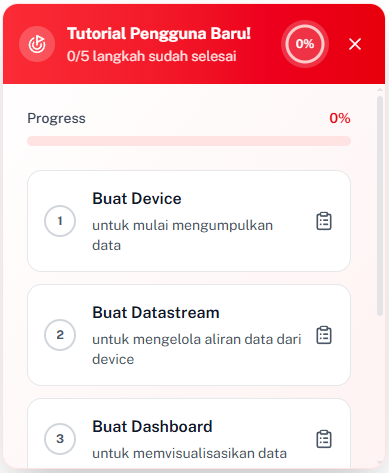
(Sumber: Dokumentasi pribadi)

Berdasarkan hasil pengujian black box terhadap sistem, telah dilakukan 16 skenario pengujian yang mencakup seluruh fitur utama pada aplikasi, antara lain proses login, manajemen perangkat IoT, pengelolaan datastream, visualisasi grafik pemantauan, serta pengaturan alarm notifikasi. Hasil pengujian disajikan dalam bentuk *diagram pie* yang menggambarkan distribusi status pengujian terhadap masing-masing fungsi. Dari keseluruhan pengujian, seluruh skenario menunjukkan status “Sesuai” (100%), yang berarti semua fungsi berjalan sesuai dengan parameter yang diharapkan. Hasil ini menunjukkan bahwa dari sisi fungsionalitas, sistem telah memenuhi kebutuhan dasar pengguna sebagaimana dirancang pada tahap perancangan sistem. Setiap input yang diberikan menghasilkan *output* yang sesuai, serta tidak ditemukan kesalahan dalam alur proses maupun tampilan antar muka pengguna pada saat pengujian dilakukan.

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 4.2.



1. Daftar panduan pengguna baru

(Sumber: Dokumentasi pribadi)

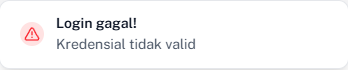
Pengguna baru dapat mengikuti panduan tersebut dengan menekan daftar langkah yang tersedia. Terdapat lima langkah utama dalam panduan ini, yaitu:

|  |  |  |
| --- | --- | --- |
| 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* (grafik) 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

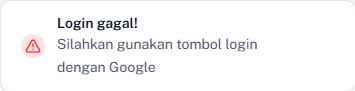
Proses *login* dapat dinyatakan gagal apabila pengguna memasukkan alamat email atau kata sandi yang tidak sesuai. Dalam kondisi tersebut, sistem akan menampilkan indikator berupa pesan peringatan “Login gagal!” disertai dengan keterangan “Kredensial tidak valid”, sebagaimana ditunjukkan pada Gambar 4.3.



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 4.4.



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 4.5.



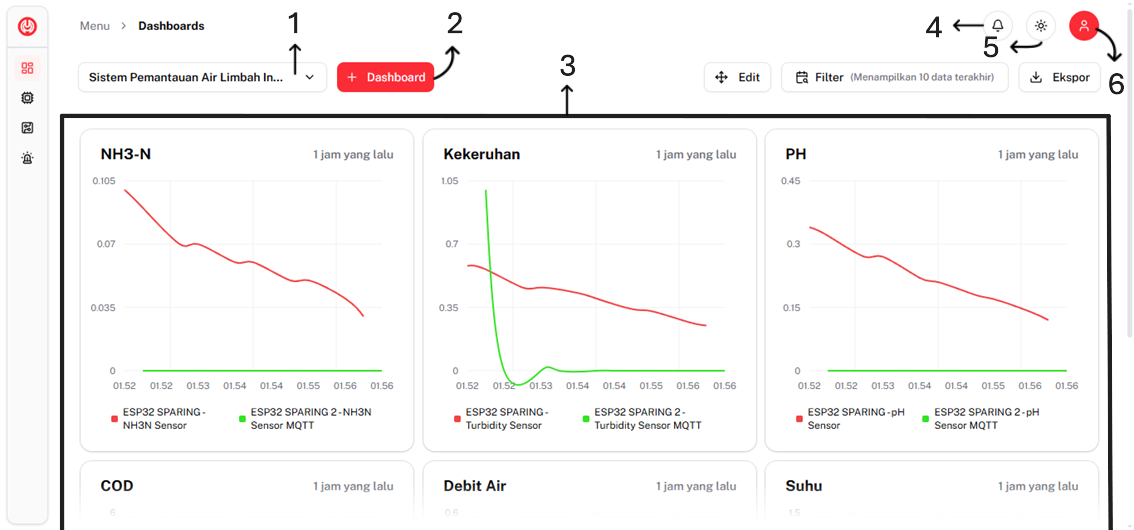
1. Indikator kolom kosong

(Sumber: Dokumentasi pribadi)

* + 1. Mengakses Halaman *Dashboard*

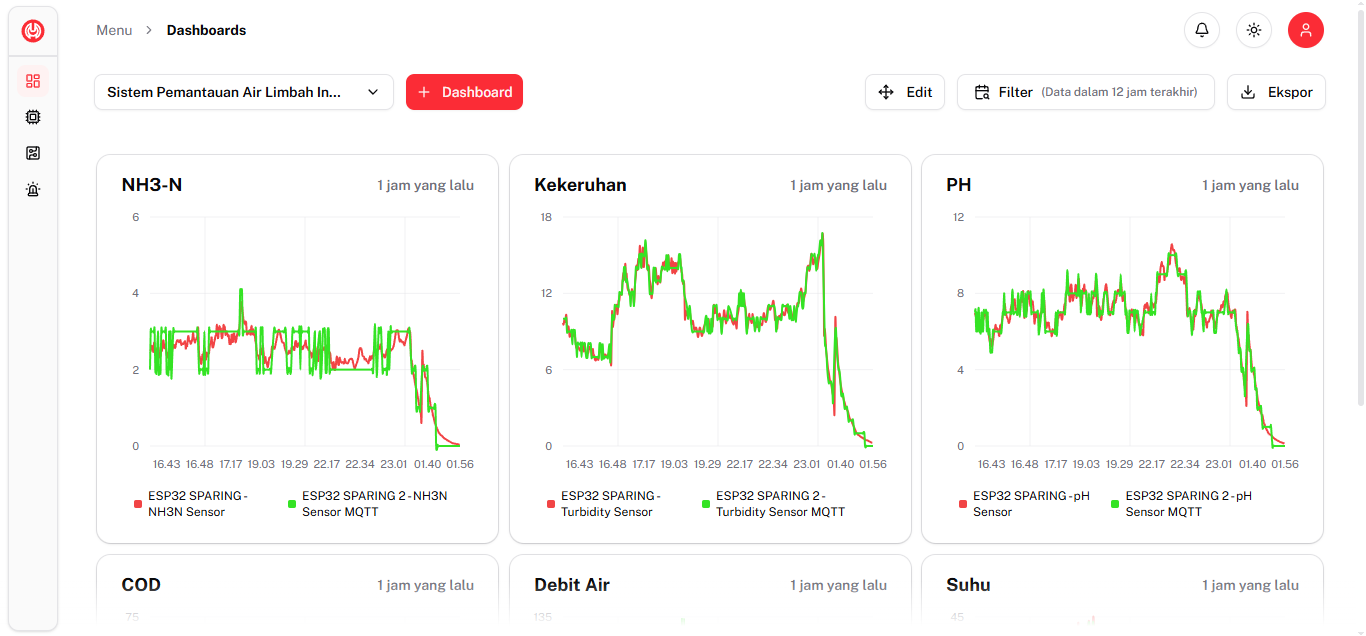
Pada halaman *Dashboard*, pengguna dapat melihat grafik yang telah ditambahkan sebelumnya. Data yang ditampilkan pada grafik dapat difilter berdasarkan rentang waktu tertentu dengan menekan tombol Filter, sebagaimana ditunjukkan pada Gambar 4.6 dengan filter berdasarkan jumlah data dan Gambar 4.7 dengan filter berdasarkan waktu. 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. Pada halaman *dashboard* terdapat beberapa komponen seperti pada Gambar 4.6, yaitu:

|  |  |  |
| --- | --- | --- |
| 1) | Nama *Dashboard* : | Kolom ini menampilkan nama *dashboard* yang telah ditambahkan oleh pengguna. |
| 2) | Tombol : | Tombol ini digunakan untuk menambahkan *dashboard*. |
| 3) | Kanvas *Dashboard* : | Ini merupakan bagian yang menampilkan grafik yang telah ditambahkan. |
| 4) | Ikon Notifikasi : | Ikon ini untuk menampilkan dan melihat daftar riwayat notifikasi. |
| 5) | Ikon Tema : | Ikon untuk merubah tema latar menjadi gelap atau terang. |
| 6) | Ikon Profil : | Ikon untuk melihat profil pengguna dan terdapat tombol keluar. |



1. Halaman dashboarddengan filter berdasarkan jumlah data

(Sumber: Dokumentasi pribadi)

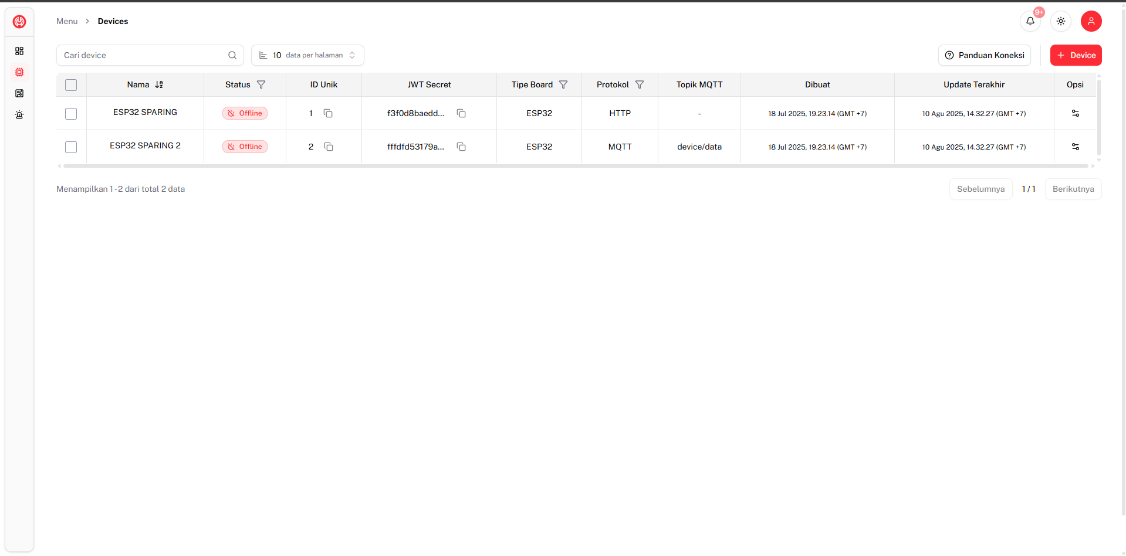


1. Halaman dashboarddengan filter berdasarkan waktu

(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, sebagaimana ditunjukkan pada Gambar 4.8.



1. Halaman devices

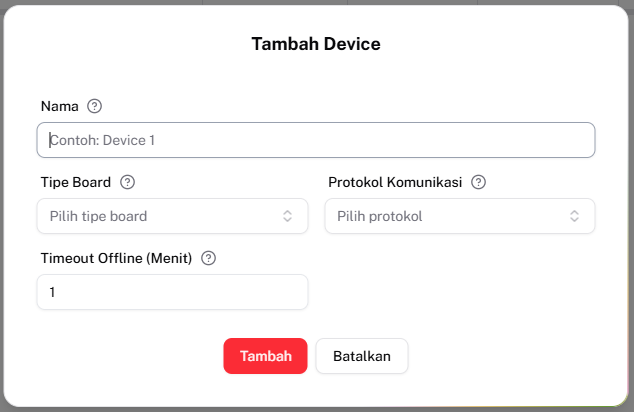
(Sumber: Dokumentasi pribadi)

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. |

* + 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 4.9.

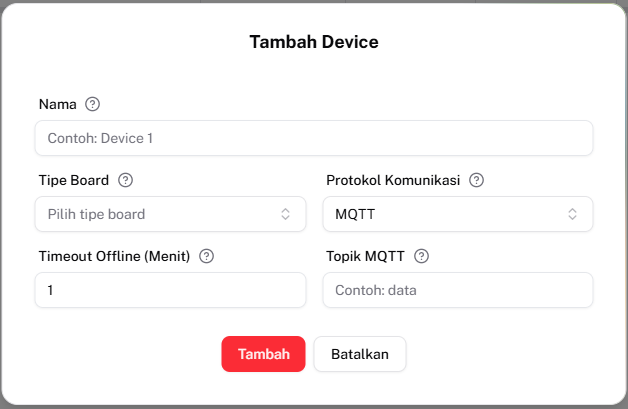


1. Formulir tambah device

(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 4.10. |
| 5) | *Timeout Offline* | Kolom berupa isian untuk menentukan waktu jeda pengiriman data. Jika perangkat tidak mengirimkan data apapun ke *server* dalam rentang waktu ini, maka *server* akan menganggap perangkat tersebut tidak aktif atau *offline*. |



1. Formulir tambah device ketika memilih protokol MQTT

(Sumber: Dokumentasi pribadi)

Setelah perangkat (*device*) berhasil ditambahkan, akan ditampilkan indikator keberhasilan sebagaimana ditunjukkan pada Gambar 4.11.



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 4.12.



1. Formulir edit device

(Sumber: Dokumentasi pribadi)

Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar 4.13.

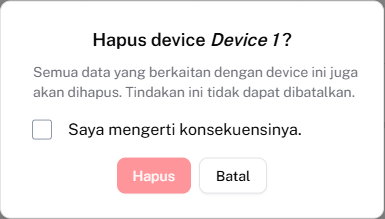


1. Indikator berhasil mengubah data perangkat

(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 14.



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 4.15.



1. Indikator berhasil menghapus perangkat

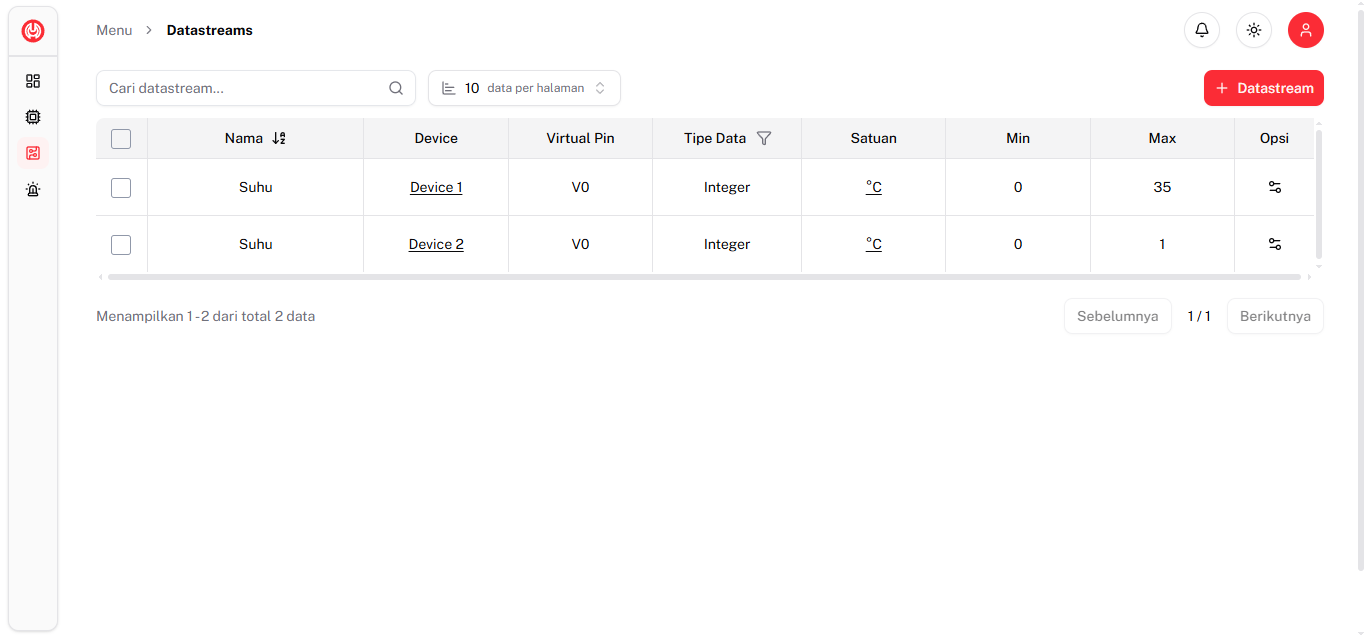
(Sumber: Dokumentasi pribadi)

* + 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 4.16.

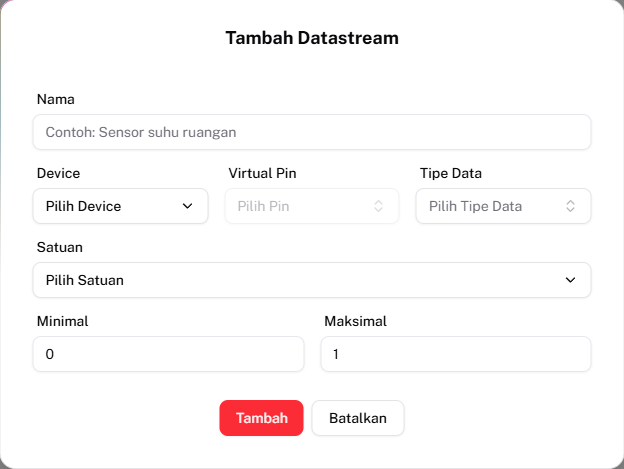


1. Halaman datastreams

(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 4.17.



1. Formulir tambah datastream

(Sumber: Dokumentasi pribadi)

Setelah perangkat (*device*) berhasil ditambahkan, akan ditampilkan indikator keberhasilan sebagaimana ditunjukkan pada Gambar 4.18.

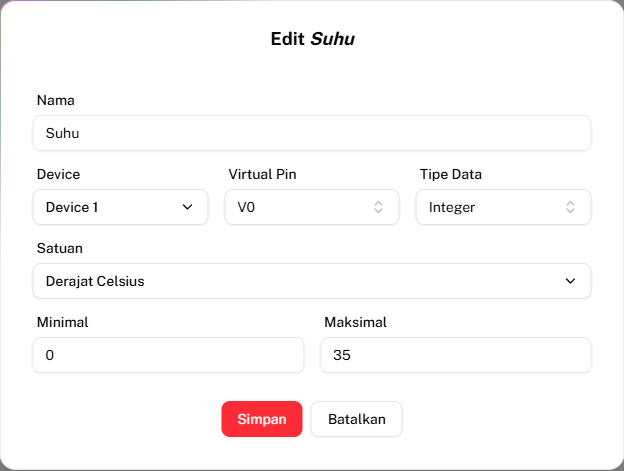


1. Indikator berhasil menambahkan datastream

(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 berjudul “Edit [Nama *Datastream*]” yang memuat informasi perangkat yang telah ditambahkan sebelumnya, sebagaimana ditunjukkan pada Gambar 4.19.



1. Formulir edit *datastream*

(Sember: Dokumentasi pribadi)

Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar 4.20.

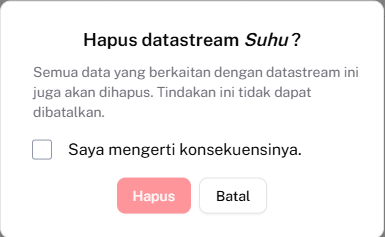


1. Indikator berhasil mengubah datastream

(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 4.21.



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 4.22.

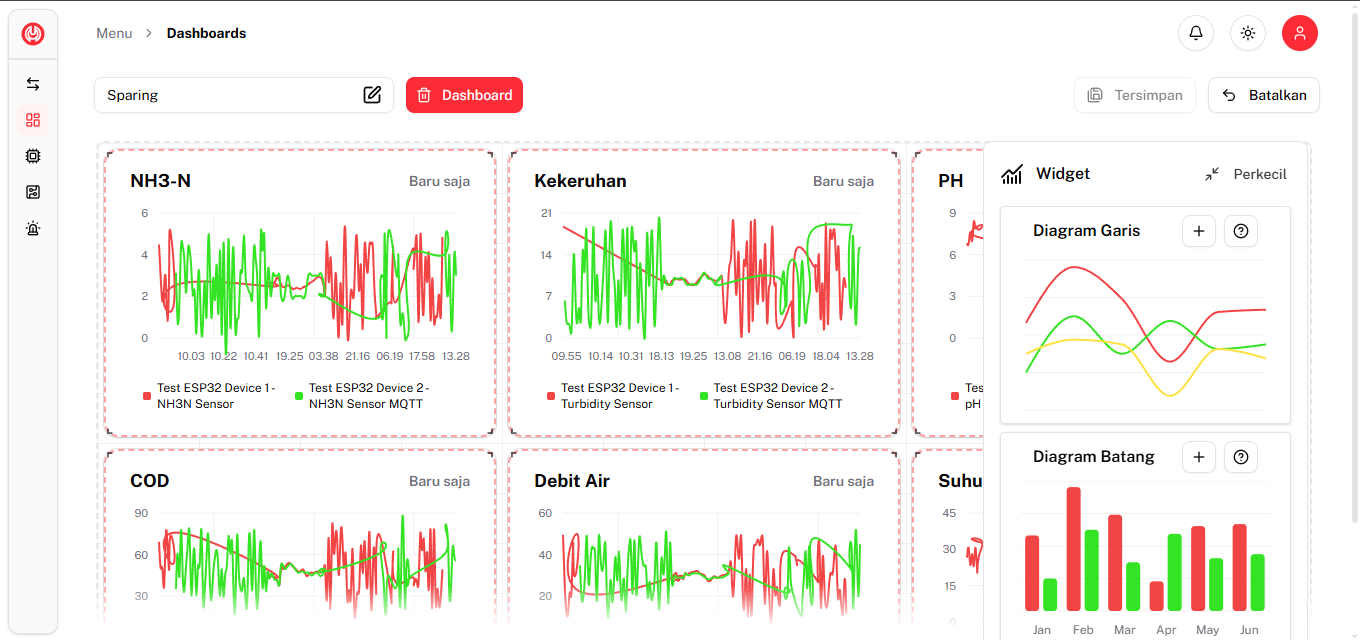


1. Indikator berhasil hapus datastream

(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 4.23.



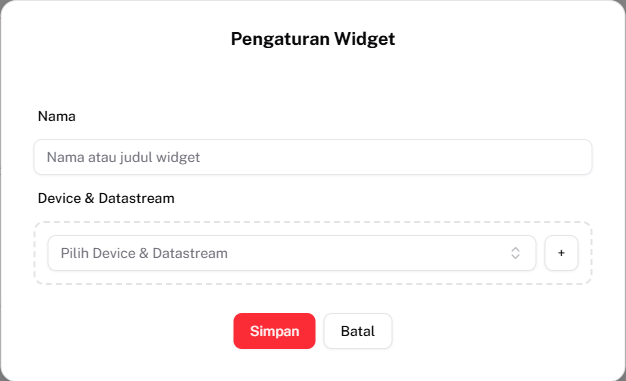
1. Mode edit halaman dashboard

(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 4.24. 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.



1. Formulir pengaturan grafik (widget)

(Sumber: Dokumentasi pribadi)

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 4.25.

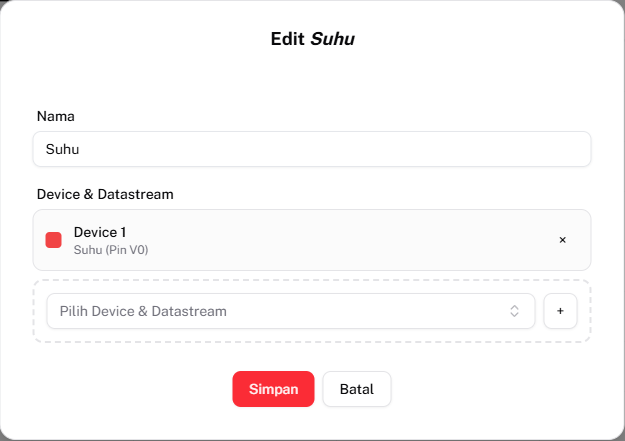


1. Indikator berhasil menyimpan dashboard

(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 4.26. Setelah proses penyimpanan berhasil, sistem akan menampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar 4.25.



1. Formulir edit grafik (widget)

(Sumber: Dokumentasi pribadi)

* + 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 4.25.

* + 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 4.27. 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. |



1. Formulir tambah *alarm*

(Sumber: Dokumentasi pribadi)

Setelah *alarm* berhasil ditambahkan, akan ditampilkan indikator keberhasilan, sebagaimana ditunjukkan pada Gambar 4.28.



1. Indikator berhasil menambahkan *alarm*

(Sumber: Dokumentasi pribadi)

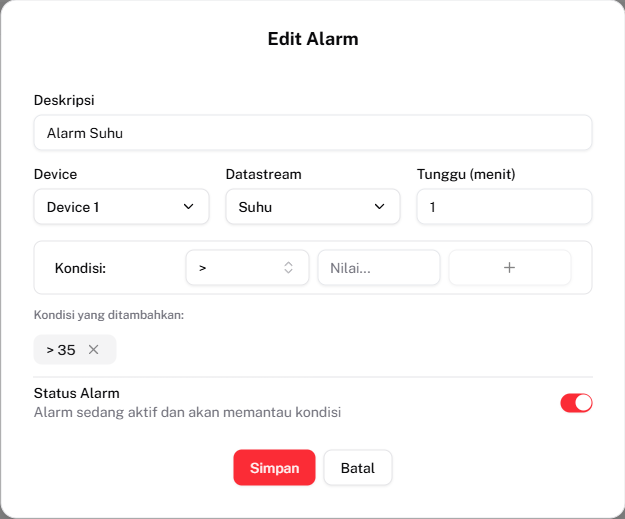
* + 1. Mengubah Alarm Notifikasi

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 4.30. Setelah proses penyimpanan data berhasil dilakukan, sistem akan menampilkan indikator sebagai konfirmasi keberhasilan, seperti diperlihatkan pada Gambar 4.29.



1. Indikator berhasil mengubah *alarm*

(Sumber: Dokumentasi pribadi)

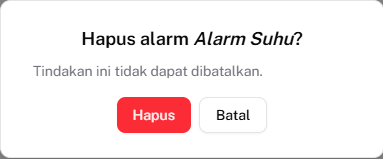


1. Formulis edit *alarm*

(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 4.31.



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 4.32.



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 4.1.

1. Hasil pengujian sistem keamanan

| **No.** | **Deskripsi Pengujian** | **Gambar** |
| --- | --- | --- |
| 1. | Pengguna mengakses halaman aplikasi menggunakan token JWT yang benar. |  |
| 2. | Pengguna mengakses halaman aplikasi menggunakan token JWT yang dimodifikasi. |  |
| 3. | Pengguna mengakses halaman aplikasi tanpa disertai *refresh token* JWT |  |
| 4. | Node mengirim data sensor menggunakan token yang benar |  |
| 5. | Node mengirim data sensor menggunakan token yang kadaluarsa |  |
| 6. | Node mengirim data sensor menggunakan token yang dimodifikasi |  |

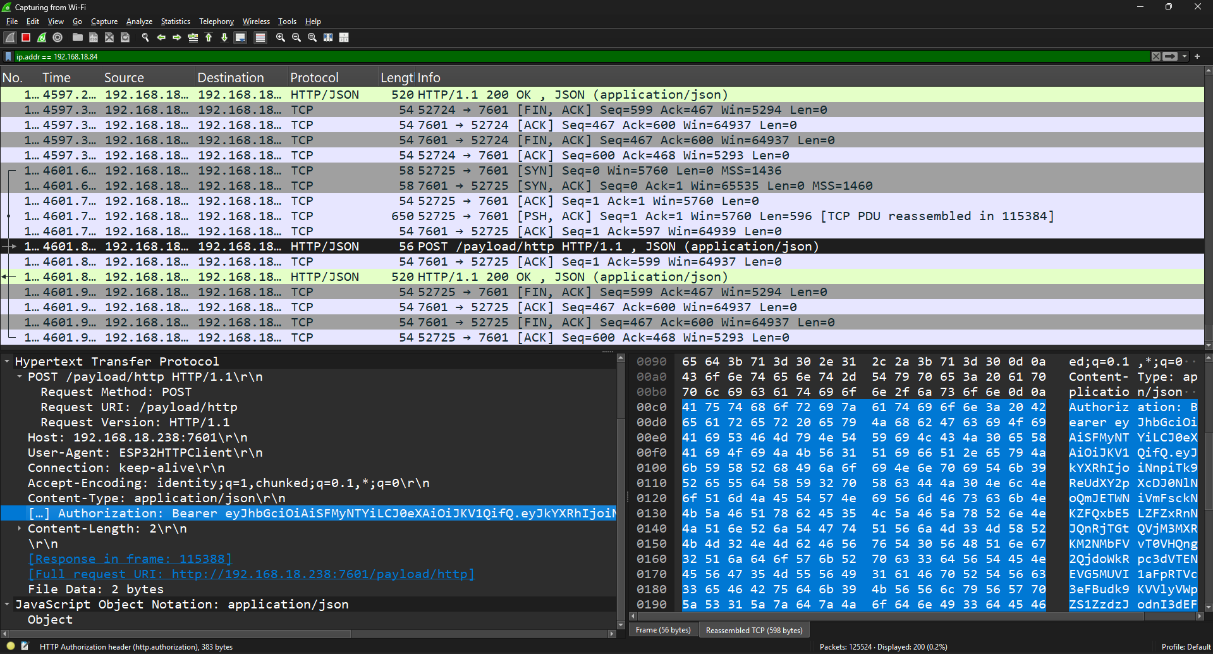
Pada pengujian pertama (nomor 1), pengguna berhasil mengakses halaman aplikasi dengan menggunakan token JWT yang valid. Sistem menampilkan *dashboard* utama dengan grafik pemantauan limbah cair. Hal ini membuktikan bahwa sistem autentikasi berfungsi dengan baik ketika token yang digunakan masih aktif dan belum kedaluwarsa. Komunikasi antara klien dan *server* berjalan lancar tanpa ada pesan error atau penolakan akses.

Pengujian kedua (nomor 2) menunjukkan hasil yang berbeda ketika token JWT dimodifikasi secara manual. Sistem langsung mendeteksi bahwa token telah diubah dan menolak akses dengan menampilkan pesan "Akses ditolak!" disertai ikon keamanan yang menandakan adanya pelanggaran autentikasi. Respons sistem ini mengonfirmasi bahwa mekanisme verifikasi integritas token berfungsi dengan baik, di mana setiap perubahan pada struktur atau isi token akan langsung terdeteksi melalui proses validasi *signature*.

Pada pengujian ketiga (nomor 3), ketika pengguna mencoba mengakses aplikasi tanpa menyertakan token JWT dalam request, sistem kembali menampilkan pesan "Akses ditolak!". Hal ini mendemonstrasikan bahwa sistem telah mengimplementasikan mekanisme autentikasi yang tepat, di mana setiap permintaan yang masuk akan diperiksa keberadaan token-nya sebelum diizinkan mengakses halaman yang dilindungi.

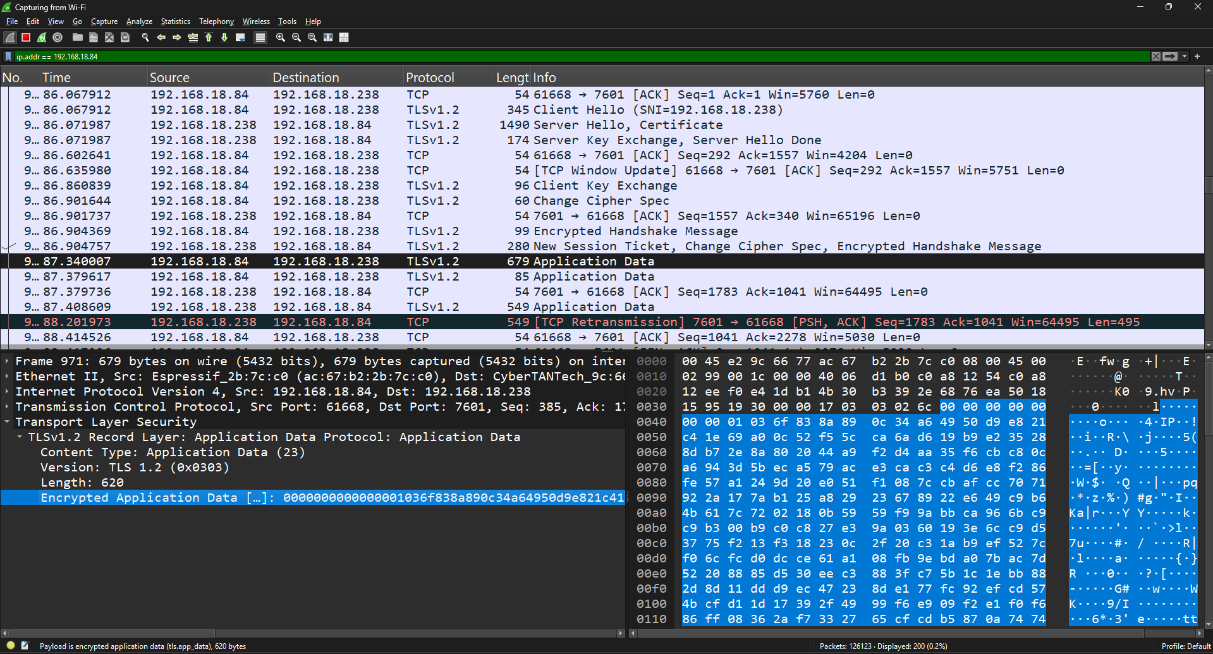
Pengujian nomor 4, 5, dan 6 menunjukkan berbagai skenario respons dari sisi *server* saat menerima data sensor dari *node*. Terlihat bahwa *server* dapat membedakan antara token yang valid, token yang kadaluwarsa, dan token yang dimodifikasi. Setiap kondisi menghasilkan respons yang sesuai, baik berupa pengiriman data sensor untuk token yang valid maupun penolakan akses untuk token yang bermasalah. Pengiriman data sensor pada pengujian-pengujian ini dilakukan masing-masing dua kali menggunakan protokol yang berbeda, yaitu HTTP dan HTTPS.

Gambar 4.33 menunjukkan hasil *capture* paket menggunakan aplikasi Wireshark ketika data dikirim melalui protokol HTTP biasa. Terlihat jelas bahwa seluruh *payload* HTTP, termasuk token JWT yang berisi informasi sensitif, dapat dibaca dengan mudah dalam bentuk plain text. Hal ini membuat token JWT rentan terhadap serangan MITM dan *packet sniffing*, di mana penyerang dapat dengan mudah mengintip dan mencuri token yang sedang ditransmisikan.



1. *Payload* HTTP pada Wireshark

(Sumber: Dokumentasi pribadi)

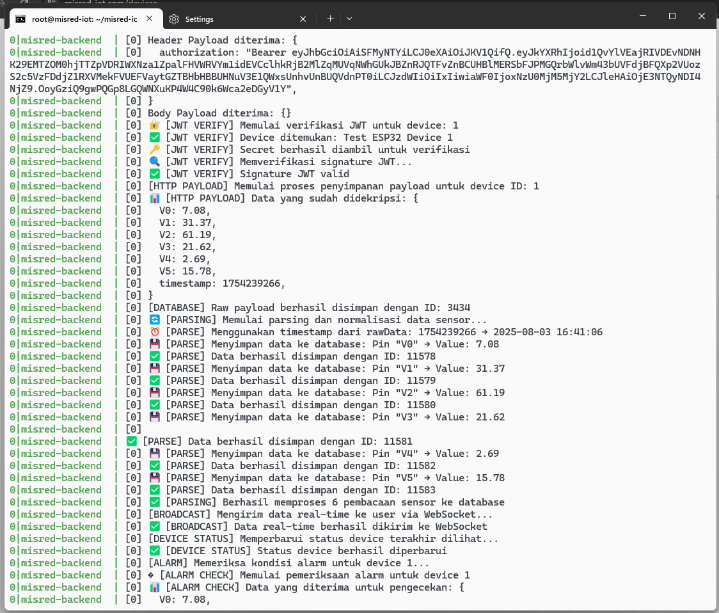


1. *Payload* HTTPS pada Wireshark

(Sumber: Dokumentasi pribadi)

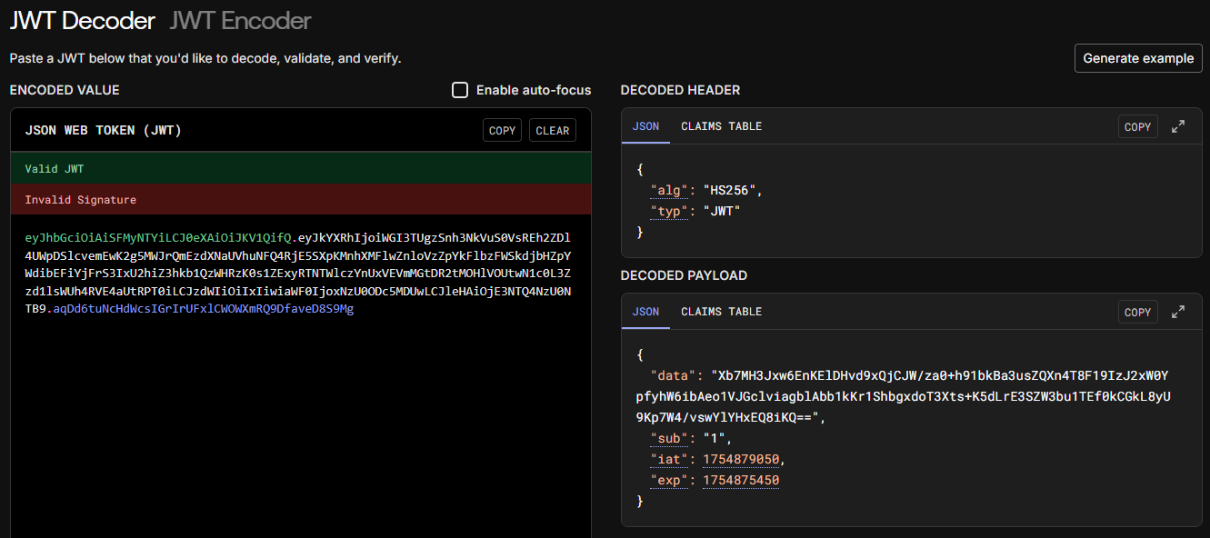
Sebaliknya, pada Gambar 4.34 yang menunjukkan pengiriman data melalui protokol HTTPS, terlihat bahwa seluruh *payload* aplikasi telah terenkripsi menggunakan TLS (*Transport Layer Security*). Data yang sebelumnya dapat dibaca dengan jelas pada HTTP, kini tampil sebagai rangkaian karakter terenkripsi yang tidak dapat dipahami tanpa kunci dekripsi yang sesuai. Enkripsi ini memastikan bahwa token JWT dan data sensitif lainnya terlindungi selama proses transmisi, sehingga meskipun paket data berhasil dicegat oleh pihak yang tidak berwenang, informasi di dalamnya tetap tidak dapat diakses atau digunakan untuk tujuan yang merugikan.

Gambar 4.35 menunjukkan *log* dari *server* yang berisi tentang bagaimana data sensor dari *node* diproses oleh *server.*  *Server* mengawali dengan melakukan pemeriksaan atribut pada ” *Authorization*” pada *header* permintaan, 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.



1. Proses pemeriksaan payload HTTP

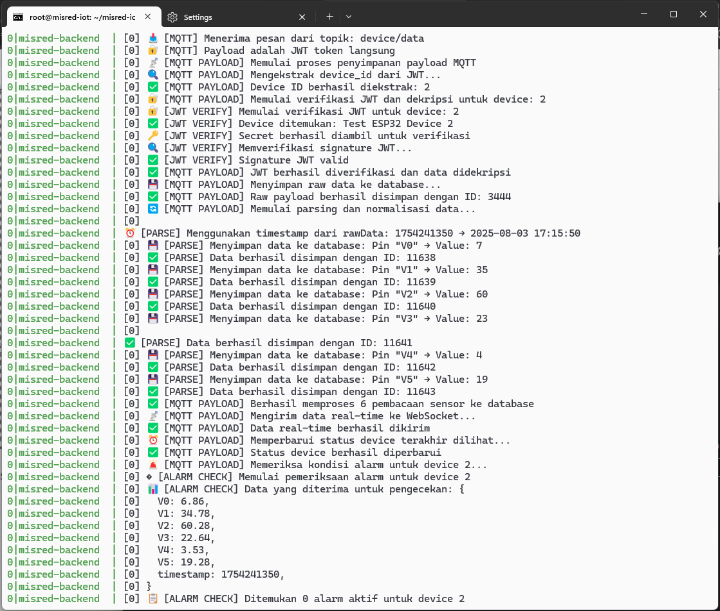
(Sumber: Dokumentasi pribadi)



1. Proses Decode Token JWT pada Website Jwt.io

(Sumber: Dokumentasi pribadi)

Proses verifikasi token diawali dengan 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. Gambar 4.6 menunjukkan tampilan visual dari token saat berhasil di-*decode.* 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 pemrosesan data lebih lanjut agar dapat menyimpan data *sensor* dengan benar pada tabel *payloads*, termasuk melakukan dekripsi AES pada data sensor, hingga pemeriksaan alarm. Hal ini juga berlaku sama pada pengiriman menggunakan protokol MQTT, *server* akan memproses data yang dikirimkan oleh *node* pada topik yang sudah ditentukan. Gambar 4.7 menunjukkan *log* dari *server* saat menerima data dari perangkat MQTT yang melakukan *publish* ke topik ”device/data”.



1. Proses pemeriksaan payload MQTT

(Sumber: Dokumentasi Pribadi)

Proses dekripsi data terenkripsi yang dikirimkan oleh perangkat IoT dilakukan untuk mengubah *ciphertext* menjadi *plaintext* yang dapat dibaca dan diproses oleh sistem. Data yang diterima berada dalam format *Base64*, sehingga langkah pertama yang dilakukan adalah melakukan *decoding* Base64 untuk memperoleh data biner asli. Data biner tersebut terdiri dari dua bagian utama, yaitu:

1. *Initialization Vector* (IV) – diambil dari 16 *byte* pertama hasil *decoding*. IV diperlukan pada algoritma AES-CBC untuk proses pembalikan enkripsi pada blok pertama.
2. *Ciphertext* (Ci) – sisa data setelah *byte* ke-16 hingga akhir, yang merupakan hasil enkripsi data sensor.

Secara konseptual, proses ini dapat digambarkan sebagai berikut:

Base64Decode( Xb7MH3Jxw6EnKElDHvd9xQjCJW/za0+h91bkBa3usZQXn4T8F19IzJ2xW0YpfyhW6ibAeo1VJGclviagblAbb1kKr1ShbgxdoT3Xts+K5dLrE3SZW3bu1TEf0kCGkL8yU9Kp7W4/vswYlYHxEQ8iKQ==  
) = [**IV**] [**C1**] [**C2**] [**C3**] [**C4**] [**C5**] [**C6**]

Terdapat enam blok *ciphertext* dikarenakan *node* mengirimkan enam data. Selanjutnya, *server* menyiapkan JWT *secret* milik perangkat untuk proses dekripsi. Dalam analisis ini, penulis hanya akan menganalisis blok *ciphertext* pertama. Nilai *secret* (K) dan IV, serta blok ciphertext pertama yang digunakan pada pengujian ini adalah:

*K =* 23050c3dcef3c669690aab113a21c3b2

*IV =* 5dbecc1f7271c3a1272849431ef77dc5

*C1 =* 08c2256ff36b4fa1f756e405adeeb194

Proses dekripsi kemudian dilakukan menggunakan algoritma AES-128 dalam mode operasi CBC (D) menggunakan fungsi ”createDecipheriv” dari *library* bernama *crypto*, dengan nilai IV dan K yang telah ditentukan. Algoritma AES-CBC kemudian memproses blok-blok data secara berantai, mengikuti Persamaan (2.2). Pada blok pertama, hasil dekripsi di-*XOR* dengan blok IV, sedangkan pada blok-blok berikutnya, hasil dekripsi di-*XOR* dengan blok *ciphertext* sebelumnya. Contoh perhitungan pada blok pertama adalah sebagai berikut:

|  |  |  |
| --- | --- | --- |
|  | .......................................................................... | (2.2) |

08c2256ff36b4fa1f756e405adeeb194, 23050c3dcef3c669690aab113a21c3b2  
 ) 5dbecc1f7271c3a1272849431ef77dc5

7b0a225630223a20372e30382c0a225631223a200a

Hasil P1 kemudian diubah ke ASCII menjadi: ” *{"V0":7.08,"V1":*”. Untuk membaca keseluruhan data secara lengkap, dilanjutkanlah proses dekripsi hingga blok keenam (P6). Setelah semua blok selesai diproses, *padding* dihapus untuk mendapatkan *plaintext* asli. Hasil akhir dekripsi berupa *plaintext* dalam format JSON, yang berisi data pembacaan sensor dari perangkat IoT sebagai berikut:

{

"V0": 7.08,

"V1": 31.37,

"V2": 61.19,

"V3": 21.62,

"V4": 2.69,

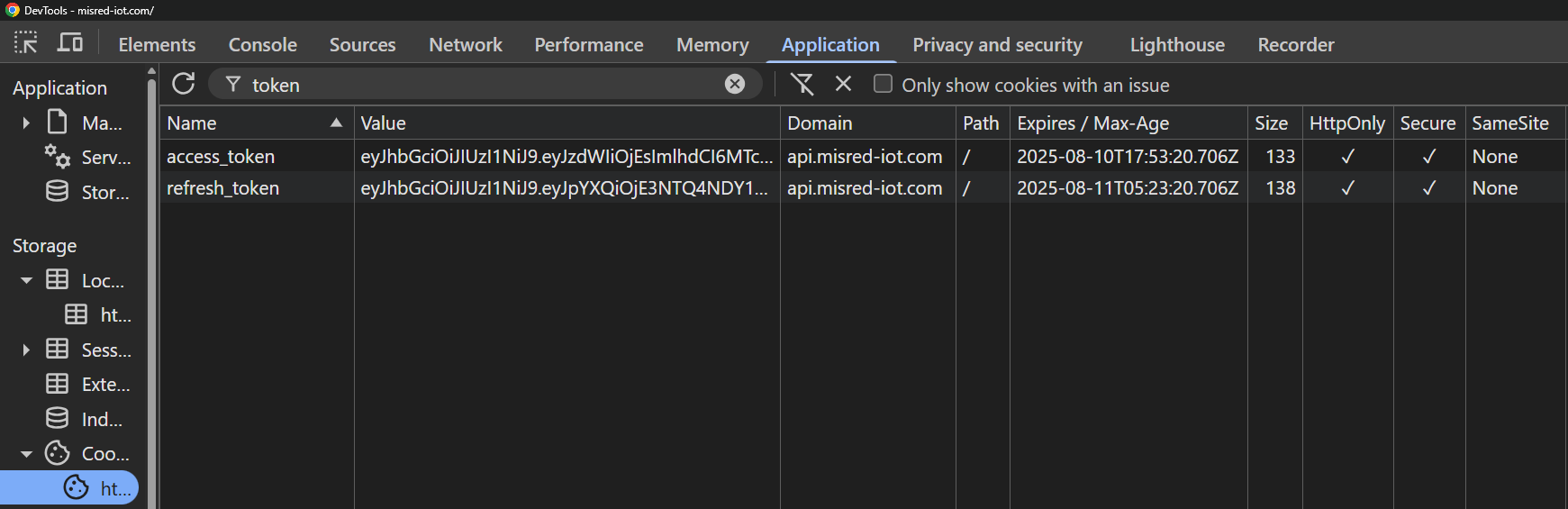
"V5": 15.78,

"timestamp": 1754239266

}

Data ”V0” hingga ”V5” pada hasil dekripsi merupakan data hasil pengukuran dari masing-masing sensor yang telah didaftarkan pada halaman *datastream*,sedangkan ”timestamp” menunjukkan waktu pengambilan data dalam format *epoch time.* Data ini merupakan data yang dibentuk oleh *node* sebelum melakukan proses enkripsi*.* Dengan mekanisme ini, data yang dikirim secara terenkripsi dapat dipastikan keasliannya serta tetap terjaga kerahasiaannya selama proses transmisi.

Pada kasus verifikasi pengguna, server juga akan melakukan autentikasi dan otorisasi melalui token JWT yang digunakan oleh pengguna saat mengakses aplikasi. Mekanisme ini memastikan bahwa hanya pengguna yang telah terverifikasi yang dapat mengakses sumber daya sistem.



1. Tampilan Menu *Cookie* pada *Browser* Pengguna

(Sumber: Dokumentasi pribadi)

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 *HttpOnly* = *true*. Konfigurasi ini 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.

Sebagai bagian dari upaya untuk memastikan keamanan sistem secara menyeluruh, dilakukan pula pengujian terhadap kekuatan *secret key* yang digunakan dalam penandatanganan token JWT. Pengujian ini mencakup dua aspek utama, yaitu (1) evaluasi tingkat entropi sebagai indikator kerandoman *secret*, dan (2) estimasi ketahanan terhadap serangan *brute force*, yang menggambarkan kemampuan kunci untuk bertahan terhadap upaya pemecahan secara sistematis.

Pada sistem yang dikembangkan, *secret key* untuk JWT digenerasi secara otomatis oleh server dalam bentuk string acak heksadesimal sepanjang 32 karakter. Contoh *secret key* yang diuji adalah sebagai berikut:

|  |
| --- |
| 23050c3dcef3c669690aab113a21c3b2 |

Karakteristik teknis dari *secret* tersebut dapat dijabarkan sebagai berikut:

* 1. Panjang (L) = 32 karakter
  2. Karakter set = hexadecimal (0–9, a–f)
  3. Jumlah karakter unik (N) = 16

Tingkat entropi dari *secret* dihitung menggunakan rumus entropi Shannon seperti yang disajikan dalam Persamaan (2.3):

|  |  |  |
| --- | --- | --- |
|  | ................................................ | (2.3) |

Nilai entropi sebesar 128 bit ini telah memenuhi rekomendasi minimal untuk kekuatan kunci kriptografi yang aman, sebagaimana diusulkan oleh Zhang dkk. (2004), serta sesuai dengan standar industri untuk sistem autentikasi dan komunikasi data. Berdasarkan hasil perhitungan ini, dapat disimpulkan bahwa *secret key* memiliki tingkat kerandoman yang tinggi dan tidak rentan terhadap pendekatan statistik maupun prediktif. Selain pengukuran entropi, dilakukan pula simulasi terhadap ketahanan *secret* terhadap serangan *brute force* yakni skenario di mana penyerang mencoba melakukan enumerasi untuk menebak nilai *secret* dengan asumsi telah memiliki akses terhadap token JWT. Jumlah kemungkinan kombinasi secret dihitung dengan rumus seperti pada Persamaan (2.4):

|  |  |  |
| --- | --- | --- |
|  | ............................................................................ | (2.4) |

Jika diasumsikan penyerang menggunakan perangkat keras yang mampu melakukan satu miliar atau 10⁹ tebakan per detik (R), maka estimasi waktu untuk menjangkau seluruh kemungkinan kombinasi *secret* dapat dihitung dengan Persamaan (2.5):

|  |  |  |
| --- | --- | --- |
|  | ................................................................... | (2.5) |

Hasil estimasi tersebut menunjukkan bahwa pendekatan *brute force* terhadap *secret* dengan panjang 32 karakter heksadesimal berada di luar jangkauan kemampuan komputasi konvensional maupun superkomputer dalam waktu yang realistis.

Berdasarkan kedua metode evaluasi yang dilakukan, yaitu analisis entropi dan simulasi *brute force*, dapat disimpulkan bahwa penggunaan *secret key* JWT sepanjang 32 karakter heksadesimal yang dihasilkan secara acak oleh *server* telah memenuhi aspek dasar keamanan sistem autentikasi. Nilai entropi sebesar 128 bit memberikan perlindungan yang memadai terhadap serangan tebakan acak (*guessing attack*), sementara simulasi *brute force* memperlihatkan bahwa kompleksitas kunci sangat tinggi dan tidak mudah dieksploitasi. Pengujian ini memperkuat keyakinan bahwa sistem yang dibangun memiliki mekanisme keamanan yang solid dalam menangani autentikasi perangkat maupun pengguna, serta mampu menjaga integritas data yang dikirimkan melalui jaringan.

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 4.2.

1. Hasil pengujian notifikasi

| **No.** | **Deskripsi Pengujian** | **Gambar** |
| --- | --- | --- |
| 1. | Menampilkan pesan notifikasi pada *browser* saat ada alarm yang terpicu atau ada peringatan perangkat yang *offline* |  |
| 2. | Menampilkan pesan notifikasi pada nomor WhatsAppsaat ada alarm yang terpicu atau ada peringatan perangkat yang *offline* |  |

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.

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 penggunaan CPU dan memori, selama aplikasi dijalankan. Hasil pengujian tersebut disajikan dengan grafik pada Gambar 4.5.

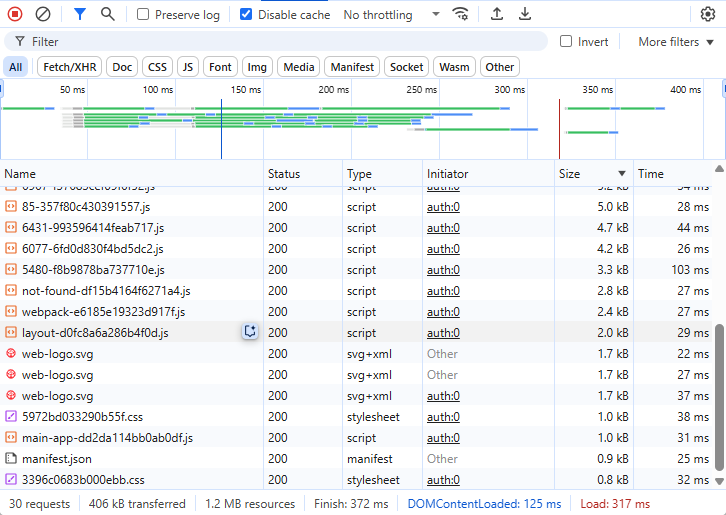
1. Hasil pengujian performa *website*

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil pengujian performa aplikasi website yang ditampilkan pada grafik, sebagaimana ditunjukkan pada Gambar 4.39. 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” ditunjukkan pada Gambar 4.40.



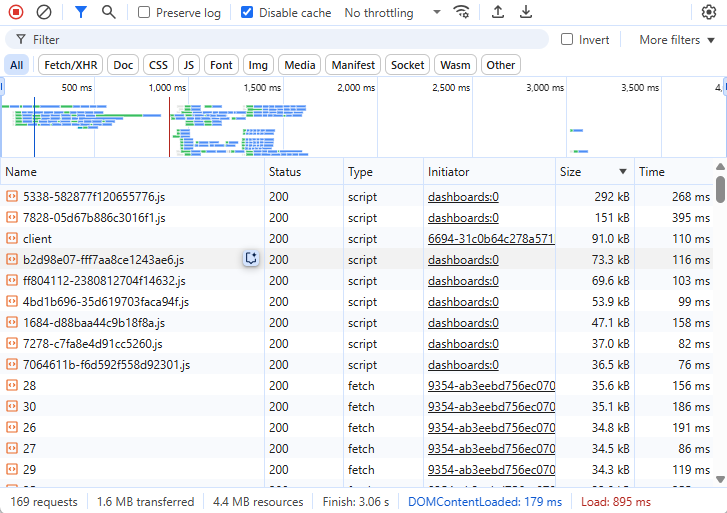
1. Hasil dari “Network Monitor” halaman *login*

(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. Berdasarkan pengujian waktu muat, halaman Login memiliki performa paling optimal dengan waktu *DOMContentLoaded* sebesar 125 ms, *Load* sebesar 317 ms, dan *Finish* sebesar 372 ms. Hal ini menunjukkan bahwa halaman *Login* relatif ringan dan tidak memiliki proses dinamis tambahan setelah halaman dimuat secara visual.

* + 1. Halaman *Dashboards*

Halaman ini menampilkan grafik data dari parameter-parameter yang telah ditambahkan oleh pengguna. Hasil dari “Network Monitor” ditunjukkan pada Gambar 4.41.



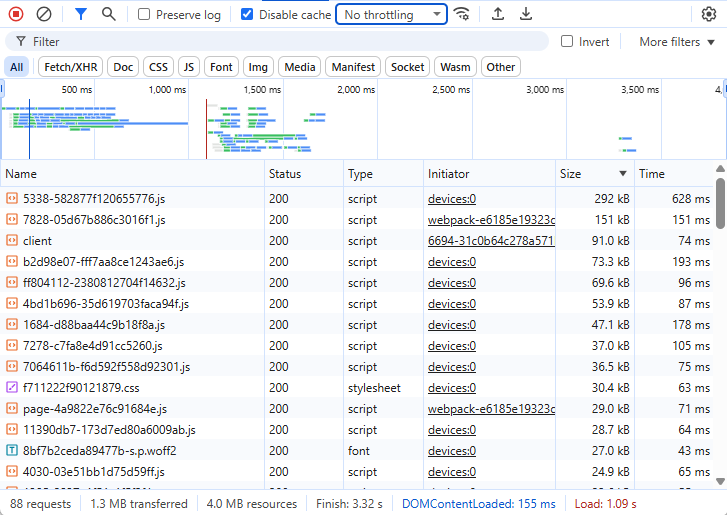
1. Hasil dari “*Network Monitor*” halaman *dashboards*

(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. Namun demikian, pengujian performa menunjukkan bahwa waktu *DOMContentLoaded* adalah 179 ms, *Load* sebesar 895 ms, dan *Finish* mencapai 3060 ms. Perbedaan signifikan antara waktu *load* dan *finish* menunjukkan adanya proses pemuatan data dinamis yang intensif setelah halaman dimuat secara visual, terutama dari permintaan API yang memuat grafik parameter.

* + 1. Halaman *Devices*

Halaman ini menyajikan data dalam bentuk tabel mengenai perangkat IoT yang telah ditambahkan oleh pengguna. Hasil dari “Network Monitor” ditunjukkan pada Gambar 4.42.



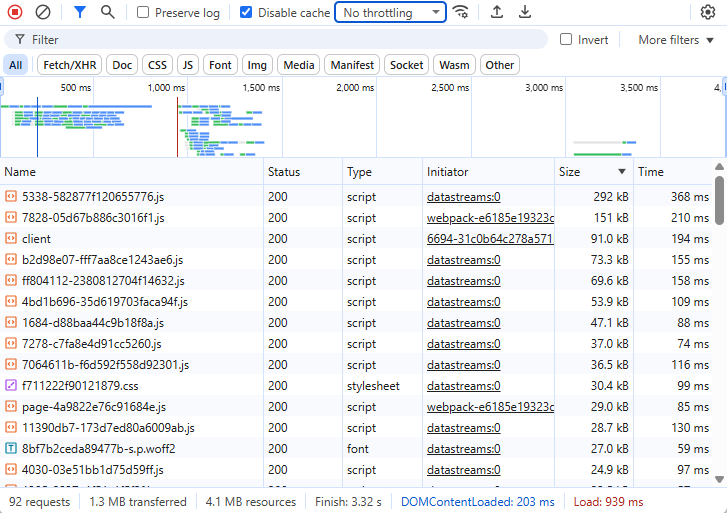
1. Hasil dari “Network Monitor” halaman *devices*

(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. Sementara itu, hasil pengujian performa menunjukkan bahwa waktu *DOMContentLoaded* adalah 155 ms, *Load* sebesar 1090 ms, dan *Finish* sebesar 3320 ms. Seperti pada halaman *Dashboards*, waktu *finish* yang tinggi mengindikasikan adanya aktivitas jaringan tambahan, yang kemungkinan disebabkan oleh pemrosesan data tabel perangkat secara dinamis.

* + 1. Halaman *Datastreams*

Halaman ini menampilkan data tabel dari parameter-parameter (datastream) perangkat IoT yang telah didaftarkan. Hasil dari “Network Monitor” ditunjukkan pada Gambar 4.43



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. Namun, hasil pengujian performa menunjukkan bahwa waktu *DOMContentLoaded* adalah 203 ms, *Load* sebesar 939 ms, dan *Finish* mencapai 3320 ms. Hal ini mengindikasikan bahwa meskipun halaman dimuat secara visual dalam waktu yang wajar, proses pengambilan data dan *rendering* komponen *datastream* memerlukan waktu tambahan yang signifikan.

* + 1. Halaman *Alarms*

Halaman ini menampilkan data tabel mengenai alarm, yaitu pengaturan ambang batas terhadap parameter dari perangkat IoT. Hasil dari “Network Monitor” ditunjukkan pada Gambar 4.44.



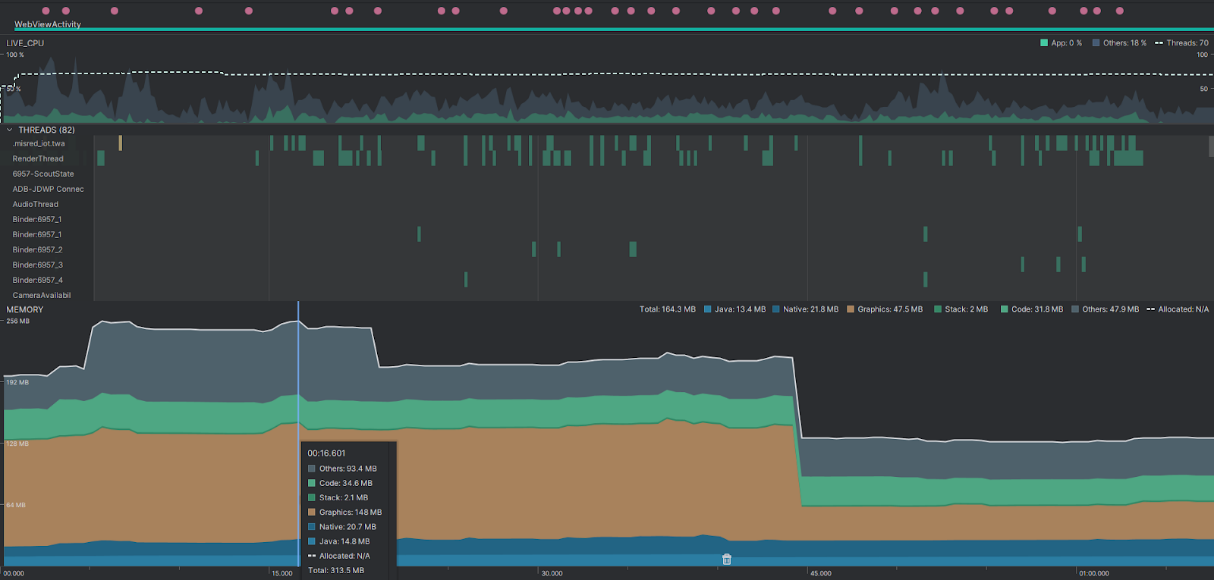
1. Hasil dari “Network Monitor” halaman *alarms*

(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. Pada sisi performa, halaman ini memiliki waktu *DOMContentLoaded* sebesar 165 ms, *Load* sebesar 819 ms, dan *Finish* sebesar 2960 ms. Meski waktu awal pemuatan tergolong cepat, lamanya waktu *finish* menandakan bahwa proses pengambilan dan pemrosesan data alarm berjalan secara bertahap dan tidak langsung ditampilkan pada saat awal.

Secara keseluruhan berdasarkan Tabel 2.2, hasil pengujian menunjukkan bahwa waktu *DOMContentLoaded* di seluruh halaman tergolong cepat (<250 ms), yang berarti struktur HTML dapat di*render* dengan baik oleh browser dan termasuk dalam kategori “Sangat baik”. Namun, waktu *Finish* pada beberapa halaman utama seperti *Dashboards*, *Devices*, dan *Datastreams* cukup tinggi (>3000 ms) dan termasuk dalam kategori “Cukup”.Hal ini menunjukkan adanya beban proses dinamis seperti pemanggilan API, pemrosesan *JavaScript*, dan visualisasi data grafik atau tabel.

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. *User Acceptance Testing* (UAT)

Pengujian User Acceptance Testing (UAT) dilakukan untuk mengevaluasi sejauh mana pengguna menerima dan merasa puas terhadap antarmuka serta fungsionalitas sistem berbasis web yang telah dikembangkan. Metode pengujian ini dilakukan dengan menyebarkan kuesioner kepada sejumlah responden yang telah menggunakan sistem, menggunakan skala Likert 4 poin, yaitu: Sangat Tidak Setuju (1), Tidak Setuju (2), Setuju (3), dan Sangat Setuju (4). Kuesioner terdiri atas 34 pernyataan yang dikelompokkan ke dalam aspek-aspek UI/UX seperti tampilan antarmuka, kemudahan navigasi, kejelasan informasi, performa, responsivitas, kenyamanan penggunaan, dan fungsionalitas fitur. Data hasil kuesioner kemudian dianalisis dengan menghitung nilai rata-rata dari setiap pernyataan, serta mengkonversinya ke dalam bentuk persentase sesuai dengan Persamaan (2.1).

1. Hasil rata-rata tiap pernyataan

(Sumber: Dokumentasi pribadi)

Berdasarkan hasil rekapitulasi nilai rata-rata, diperoleh bahwa sebagian besar pernyataan memperoleh skor di atas 3, yang menunjukkan bahwa mayoritas pengguna menyatakan “Setuju” hingga “Sangat Setuju” terhadap kualitas antarmuka dan fungsionalitas sistem. Nilai tertinggi diperoleh pada pernyataan nomor 2 dalam aspek Visualisasi dan Antarmuka, nomor 11 dalam aspek Fungsionalitas dan Interaksi, serta nomor 34 dalam aspek Notifikasi dan Respons Sistem, sebagaimana tercantum dalam Tabel 3.7. Ketiga pernyataan tersebut memiliki rata-rata skor sebesar 3,81 atau setara dengan 95,25%, yang menurut interpretasi pada Tabel 2.3 termasuk dalam kategori “Sangat Setuju”.  
Sementara itu, pernyataan dengan skor terendah adalah pernyataan nomor 12, yaitu “Sistem memberikan umpan balik ketika terjadi kesalahan,” yang juga termasuk dalam aspek Fungsionalitas dan Interaksi sebagaimana dijelaskan pada Tabel 3.7. Pernyataan ini memperoleh rata-rata skor sebesar 3,63 atau setara dengan 90,75%, yang tetap berada dalam kategori “Setuju” berdasarkan Tabel 2.3.

Jika dihitung secara keseluruhan, nilai rata-rata dari seluruh pernyataan adalah 3,73, yang setara dengan 93,25%. Hasil ini mengindikasikan bahwa sistem telah secara umum diterima dengan sangat baik oleh pengguna dari sisi kenyamanan, kemudahan penggunaan, serta kepuasan terhadap fitur dan antarmuka sistem. Dengan demikian, dapat disimpulkan bahwa sistem yang telah dikembangkan telah berhasil memenuhi ekspektasi pengguna dan layak untuk digunakan dalam konteks operasional.

1. Evaluasi

Berdasarkan hasil pengujian yang telah dilakukan, sistem pemantauan limbah cair industri terintegrasi IoT dan JWT berbasis *website* dan Android yang dikembangkan menunjukkan keunggulan signifikan dibandingkan dengan penelitian-penelitian sebelumnya. Evaluasi ini mengacu pada perbandingan dengan penelitian yang telah dirangkum dalam Tabel 2.1 pada Bab II.

1. Komparasi hasil penelitian

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Aspek Evaluasi** | **Tugas Akhir** | **Febriana & Farros (2024)** | **Salem dkk. (2022)** | **Bogdan dkk. (2023)** | **Raman & Martin (2024)** | **Shodiq dkk. (2021)** |
| **Protokol Komunikasi** | HTTP, MQTT \* | HTTP | HTTP | HTTP | MQTT | - |
| **Keamanan Data** | JWT, AES-128, HTTPS \* | - | - | - | - | JWT, XXTEA |
| **Entropi JWT Secret** | 128 bit \* | - | - | - | - | ~64 bit |
| **Resistensi Brute Force** | ~10²¹ tahun \* | - | - | - | - | ~10¹⁰ tahun |
| **Platform Akses** | *Website*, Android \* | *Website* | *Website* | Android | *Website* | *Website* |
| **Sistem Notifikasi** | *Browser* (Bawaan), WhatsApp \* | - | SMS | Android (Bawaan) | - | - |
| **Fungsionalitas** | *Black Box* (100% ) | *-* | *Black Box* (100% ) | *Black Box* (100% ) | *Black Box* (100% ) | *-* |
| **Performa Website** | DOMContentLoaded <250ms \* | - | - | - | - | - |
| **Performa Android** | CPU <10%, RAM 20–31MB \* | - | - | - | - | - |
| **User Acceptance Testing** | 93,25% \* | - | - | - | - | - |

(\* *lebih unggul*)

Pertama, dari aspek interoperabilitas protokol komunikasi, sistem yang dikembangkan mampu mendukung dua protokol komunikasi sekaligus, yaitu HTTP dan MQTT. Hal ini memberikan fleksibilitas yang lebih tinggi dibandingkan dengan penelitian Febriana & Farros (2024), Salem dkk. (2022), dan Bogdan dkk. (2023) yang hanya mendukung protokol HTTP, serta penelitian Raman & Martin (2024) yang hanya menggunakan protokol MQTT. Kemampuan ini memungkinkan pengguna untuk mengintegrasikan berbagai jenis perangkat IoT dengan protokol komunikasi yang berbeda dalam satu ekosistem pemantauan.

Kedua, dari segi keamanan data, sistem ini mengimplementasikan mekanisme keamanan berlapis yang terdiri dari JWT untuk autentikasi dan otorisasi, serta HTTPS untuk enkripsi komunikasi. Implementasi ini merupakan peningkatan signifikan dibandingkan dengan sebagian besar penelitian sebelumnya yang tidak menerapkan sistem keamanan sama sekali, seperti penelitian Febriana & Farros (2024), Raman & Martin (2024), Salem dkk. (2022), Bogdan dkk. (2023), dan Shakdher dkk. (2019). Meskipun penelitian Shodiq dkk. (2021) telah menerapkan JWT dan enkripsi XXTEA, implementasi pada sistem ini lebih komprehensif dengan menggabungkan HTTPS yang memberikan perlindungan ujung ke ujung.

Ketiga, dari aspek aksesibilitas platform, sistem yang dikembangkan menyediakan dua antarmuka yang dapat digunakan pengguna, yaitu *website* dan aplikasi Android. Keunggulan ini melebihi penelitian Febriana & Farros (2024), Raman & Martin (2024), Salem dkk. (2022), Shakdher dkk. (2019), dan Shodiq dkk. (2021) yang hanya menyediakan antarmuka *website*. Meskipun penelitian Bogdan dkk. (2023) telah menyediakan aplikasi Android, sistem tersebut tidak dilengkapi dengan platform website yang memungkinkan akses dari berbagai perangkat.

Keempat, dari segi fitur notifikasi, sistem ini menyediakan dua jenis notifikasi, yaitu notifikasi bawaan pada *browser*, serta notifikasi melalui WhatsApp. Hal ini memberikan respons yang lebih cepat dan mudah dibandingkan dengan penelitian Salem dkk. (2022) yang hanya menggunakan SMS, dan penelitian Bogdan dkk. (2023) yang hanya mengandalkan notifikasi bawaan aplikasi. Penelitian lainnya sama sekali tidak menyediakan fitur notifikasi.

Kelima, berdasarkan hasil pengujian fungsionalitas menggunakan metode *black box testing*, sistem mencapai tingkat keberhasilan 100% untuk seluruh 16 skenario pengujian yang dilakukan. Hasil pengujian sistem keamanan menunjukkan bahwa mekanisme JWT dan HTTPS berhasil melindungi integritas data dan mencegah akses tidak sah. Pengujian notifikasi membuktikan bahwa sistem dapat memberikan peringatan secara real-time melalui dua media komunikasi yang berbeda.

Keenam, dari aspek ketahanan terhadap serangan *brute force*, sistem ini memiliki entropi JWT *secret* sebesar 128 bit yang secara teoritis membutuhkan waktu sekitar 10²¹ tahun untuk berhasil dipecahkan dengan asumsi laju serangan satu juta percobaan per detik. Nilai ini jauh lebih tinggi dibandingkan dengan penelitian Shodiq dkk. (2021) yang hanya memiliki estimasi ketahanan sekitar 10¹⁰ tahun dengan entropi ~64 bit. Peningkatan ini menunjukkan bahwa rancangan kunci rahasia pada sistem yang dikembangkan memiliki kompleksitas yang jauh lebih tinggi dan secara signifikan memperkuat perlindungan terhadap serangan enumerasi kunci.

Ketujuh, dari aspek performa, hasil pengujian menunjukkan bahwa aplikasi *website* memiliki waktu DOMContentLoaded yang sangat baik (<250 ms) pada seluruh halaman, meskipun waktu *finish* pada halaman dengan konten dinamis mencapai 3000 ms. Sementara itu, aplikasi Android menunjukkan penggunaan CPU yang stabil di bawah 10% dan konsumsi memori yang efisien sekitar 20-31 MB.

Kedelapan, hasil UAT dengan melibatkan pengguna nyata menunjukkan tingkat penerimaan sebesar 93,25%, yang mengindikasikan bahwa sistem telah memenuhi ekspektasi pengguna dari segi antarmuka, fungsionalitas, dan pengalaman penggunaan. Evaluasi UAT ini merupakan aspek yang tidak dilakukan pada penelitian-penelitian sebelumnya, sehingga memberikan validasi tambahan terhadap kualitas sistem yang dikembangkan.

BAB V  
PENUTUP

1. Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem yang telah dilakukan, maka dapat disimpulkan bahwa:

1. Sistem pemantauan limbah cair industri berbasis IoT dan JWT yang dirancang dalam tugas akhir ini mampu melakukan pemantauan parameter limbah cair, yaitu suhu, debit, kekeruhan, pH, NH₃-N, dan COD, serta menyediakan fitur pengelolaan perangkat (*device*), parameter (*datastream*), *dashboard* pemantauan, dan sistem alarm yang terintegrasi.
2. Pengujian fungsionalitas dengan metode *black box* menunjukkan bahwa seluruh 16 skenario pengujian berhasil dijalankan dengan tingkat keberhasilan 100%. Hal ini menunjukkan bahwa sistem berjalan sesuai dengan kebutuhan dan fungsionalitas yang telah ditentukan.
3. Hasil pengujian performa menunjukkan bahwa aplikasi *website* memiliki waktu pemuatan halaman (DOMContentLoaded) yang cepat di bawah 250 ms pada seluruh halaman utama, sementara aplikasi Android menunjukkan performa yang efisien dengan konsumsi CPU di bawah 10% dan penggunaan memori berkisar antara 20–31 MB.
4. Mekanisme keamanan sistem telah diterapkan secara berlapis melalui kombinasi JWT untuk autentikasi dan otorisasi, AES-128 untuk enkripsi data pada tingkat aplikasi, serta HTTPS untuk enkripsi komunikasi end-to-end. Implementasi ini memastikan integritas, kerahasiaan, dan keaslian data tetap terjaga, sekaligus meningkatkan ketahanan sistem terhadap serangan MITM maupun *brute force*. Dengan entropi kunci rahasia sebesar 128 bit, sistem memiliki ketahanan teoretis terhadap serangan *brute force* hingga sekitar 10²¹ tahun pada asumsi kecepatan satu juta tebakan per detik, yang jauh lebih tinggi dibandingkan penelitian terdahulu.
5. Sistem pemantauan juga menyediakan fitur notifikasi yang komprehensif melalui notifikasi internal (browser dan aplikasi) serta WhatsApp, sehingga meningkatkan respons pengguna terhadap kondisi parameter yang melampaui ambang batas.
6. Hasil pengujian User Acceptance Testing (UAT) menunjukkan tingkat penerimaan pengguna sebesar 93,25%, yang menandakan bahwa sistem telah memenuhi ekspektasi pengguna dari sisi antarmuka pengguna (UI), pengalaman pengguna (UX), serta fungsionalitas sistem secara keseluruhan.
7. Saran

Berdasarkan hasil penelitian dan keterbatasan yang ditemukan, maka disarankan beberapa hal untuk pengembangan selanjutnya, antara lain:

1. Menambahkan fitur analitik *real-time* berbasis *machine learning* untuk mendukung pengambilan keputusan berbasis data.
2. Mengintegrasikan autentikasi dua faktor (*Two-Factor Authentication*/2FA) untuk peningkatan keamanan akun pengguna.
3. Mengembangkan versi aplikasi iOS agar sistem dapat digunakan oleh lebih banyak pengguna lintas platform.
4. Menyediakan sistem manajemen pengguna berbasis peran (*Role-Based Access Control*) untuk meningkatkan fleksibilitas pengelolaan hak akses.
5. Mengoptimalkan tampilan UI/UX pada perangkat dengan resolusi layar rendah dan menambahkan fitur aksesibilitas bagi pengguna dengan kebutuhan khusus.

DAFTAR PUSTAKA

Ahmad, S., Jeon, G., & Park, M. (2022). Assessing the linguistic quality of REST APIs for IoT applications. *Journal of Systems and Software*, *189*, 111294. <https://doi.org/10.1016/j.jss.2022.111294>

Ahmed, S., & Mahmood, Q. (2019). An authentication based scheme for applications using JSON web token. *2019 22nd International Multitopic Conference (INMIC)*, 1-6. <https://doi.org/10.1109/INMIC48123.2019.9022766>

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>

Aladwani, A. M. (2020). Website performance and bounce rate: An empirical investigation. *International Journal of Information Management*, *54*, 102154. <https://doi.org/10.1016/j.ijinfomgt.2020.102154>

Avancha, S., Goel, O., & Pandian, P. K. G. (2024). Agile project planning and execution in large-scale IT projects. *Deleted Journal*, *12*(3), 239–252. <https://doi.org/10.36676/dira.v12.i3.80>

Ayuningtyas, R., Putri, D. A., & Sari, M. (2023). Performance and functional testing with the black box testing method. *International Journal of Progressive Sciences and Technologies*, *39*(2), 234-241. <https://doi.org/10.52155/ijpsat.v39.2.5471>

Biznetgio. (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>

Bourhis, P., Reutter, J. L., Suárez, F., & Vrgoč, D. (2020). JSON: Data model and query languages. *Information Systems*, *89*, 101478. <https://doi.org/10.1016/j.is.2019.101478>

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>

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

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.

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>

Google. (2023). Core Web Vitals: What developers should know. *Google Developers*. <https://web.dev/vitals/>

Hosenkhan, M. R., & Pattanayak, B. K. (2021). A framework for secure communication on Internet of Things (IoT). In *Advances in Intelligent Systems and Computing* (pp. 599–605). Springer. <https://doi.org/10.1007/978-981-33-4299-6_49>

Jartarghar, H. A., Salanke, G. R., A.R, A. K., G.S, S., & Dalali, S. (2022). React apps with server-side rendering: Next.js. *Journal of Telecommunication, Electronic and Computer Engineering*, *14*(4). <https://doi.org/10.54554/jtec.2022.14.04.005>

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>

Joshi, A., Kale, S., Chandel, S., & Pal, D. K. (2015). Likert scale: Explored and explained. *British Journal of Applied Science & Technology*, *7*(4), 396–403. <https://doi.org/10.9734/BJAST/2015/14975>

JSON. (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>

Lee, J. E.-Y. (2022). *Structured Query Language (SQL)* (pp. 889–893). <https://doi.org/10.1007/978-3-319-32010-6_460>

Loring, M. C., Marron, M., & Leijen, D. (2015**).** "Static Analysis of Event-Driven Node.js JavaScript Applications." *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA '15)*. ACM. <https://doi.org/10.1145/2858965.2814272>

Machuca Yaguana, J. (2023). Tratamiento y representación de datos provenientes de escalas tipo Likert. *Ciencia Latina*, *7*(4). <https://doi.org/10.37811/cl_rcm.v7i4.6905>

Marnewick, C. (2016). Benefits of information system projects: The tale of two countries. *International Journal of Project Management*, *34*(4), 748–760. <https://doi.org/10.1016/j.ijproman.2016.03.010>

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

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>

Mudassir, M., & Mushtaq, M. (2024). The role of APIs in modern software development. *World Journal of Advanced Engineering Technology and Sciences*, *13*(1), 1045–1047. <https://doi.org/10.30574/wjaets.2024.13.1.0515>

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing* (3rd ed.). John Wiley & Sons.

Next.js. (2024). The React framework for the web. *Next.js*. <https://nextjs.org/>

Nurbaya, S. (2019). *Peraturan Menteri Lingkungan Hidup dan Kehutanan*. Kementerian Lingkungan Hidup dan Kehutanan. <https://ppkl.menlhk.go.id/website/filebox/885/200306144608%20Permen%20LHK%20tentang%20SPARING.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>

Pavlović, A., Zdravković, M., Misic, V., & Sladic, G. (2021). Performance impact of optimization methods on MySQL document-based and relational databases. *Applied Sciences*, *11*(15), 6794. <https://doi.org/10.3390/app11156794>

Pramukantoro, E. K. (2020). *Internet of Things dengan Python, ESP32, dan Raspberry PI: Teori dan praktik*. Universitas Brawijaya Press.

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)*, 1–6. <https://doi.org/10.1109/ADICS58448.2024.10533607>

Sabherwal, R., Chan, Y. E., & Hirschheim, R. (2019). Recent developments in research on the management of information technology: An overview of leading journals. *Journal of Information Technology*, *34*(2), 97–112. <https://doi.org/10.1177/0268396219831986>

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>

Saputra, W. Y., Sugiarti, S., Junianto, H., & Suhartono, D. (2025). Password strength study using the zxcvbn algorithm and brute-force time estimation to strengthen cybersecurity. *Journal of Computing and Information System*, *21*(1), 52–59. <https://doi.org/10.33480/pilar.v21i1.6119>

Sari, D. R. (2024). Analisis keamanan sistem informasi dalam era Internet of Things (IoT). *Technologia Journal: Jurnal Informatika*, *1*(2), 1-10.

Sawitri, D. (2023). Internet of Things memasuki era society 5.0. *KITEKTRO: Jurnal Komputer, Informasi Teknologi, dan Elektro*, *8*(1), 31-35.

Shah, R., & Correia, S. (2021). *Encryption of Data over HTTP (Hypertext Transfer Protocol)/HTTPS (Hypertext Transfer Protocol Secure) Requests for Secure Data transfers over the Internet*. <https://doi.org/10.1109/RTEICT52294.2021.9573978>

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)*, 70–74. <https://doi.org/10.1109/ICICyTA53712.2021.9689126>

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

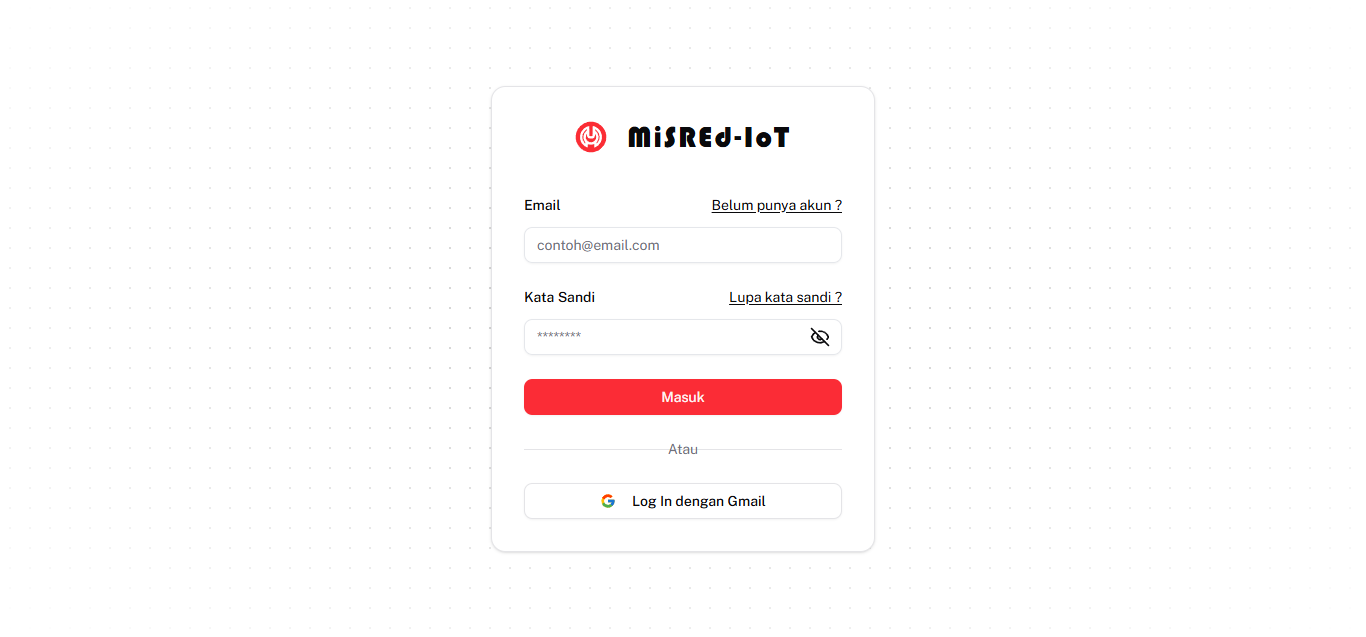
Zhang, L., Jayashankar, U., & Babu, C. A. (2022). Hardware based entropy calculation in crypto applications. *2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 1-6. <https://doi.org/10.1109/CONECCT52877.2021.9622546>

LAMPIRAN

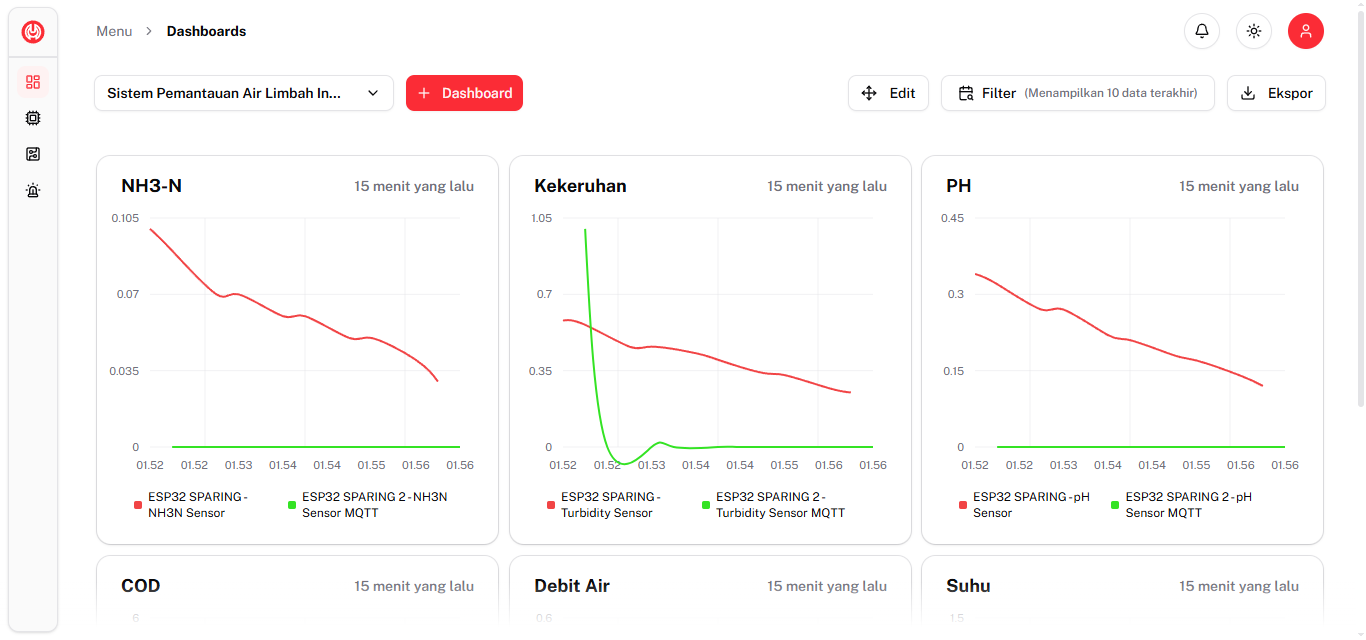
1. Halaman *Landing Page*



1. Halaman *Login*



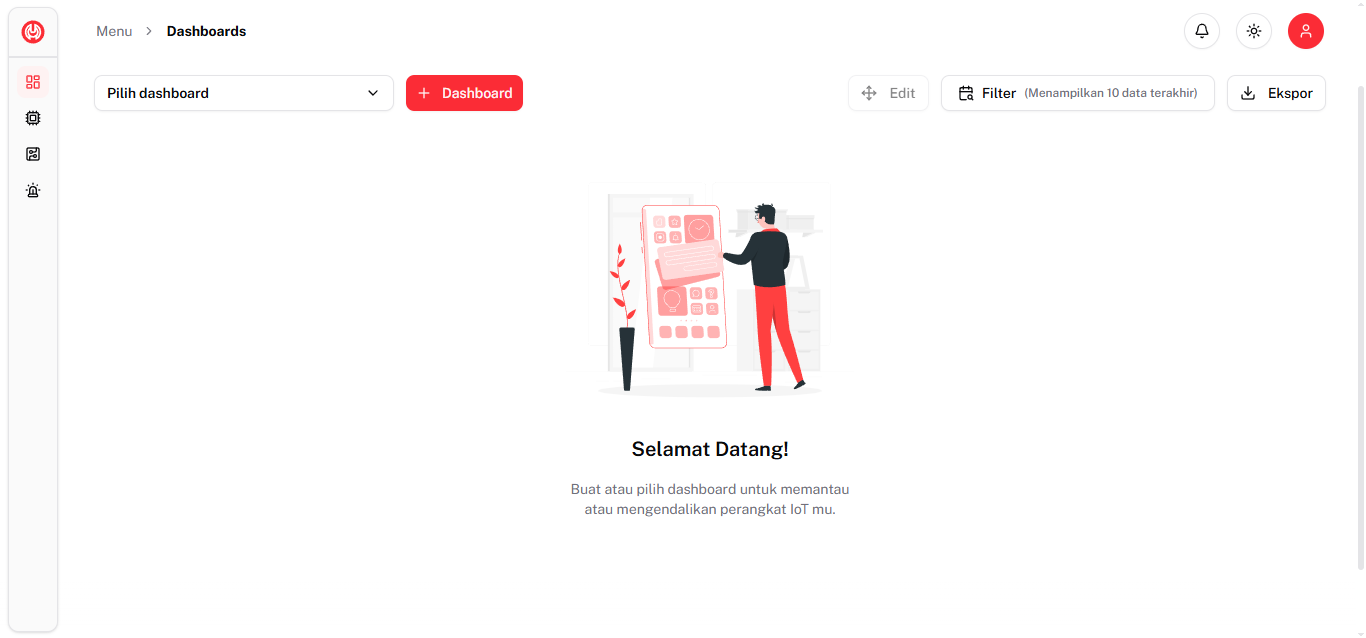
1. Halaman *Dashboards*



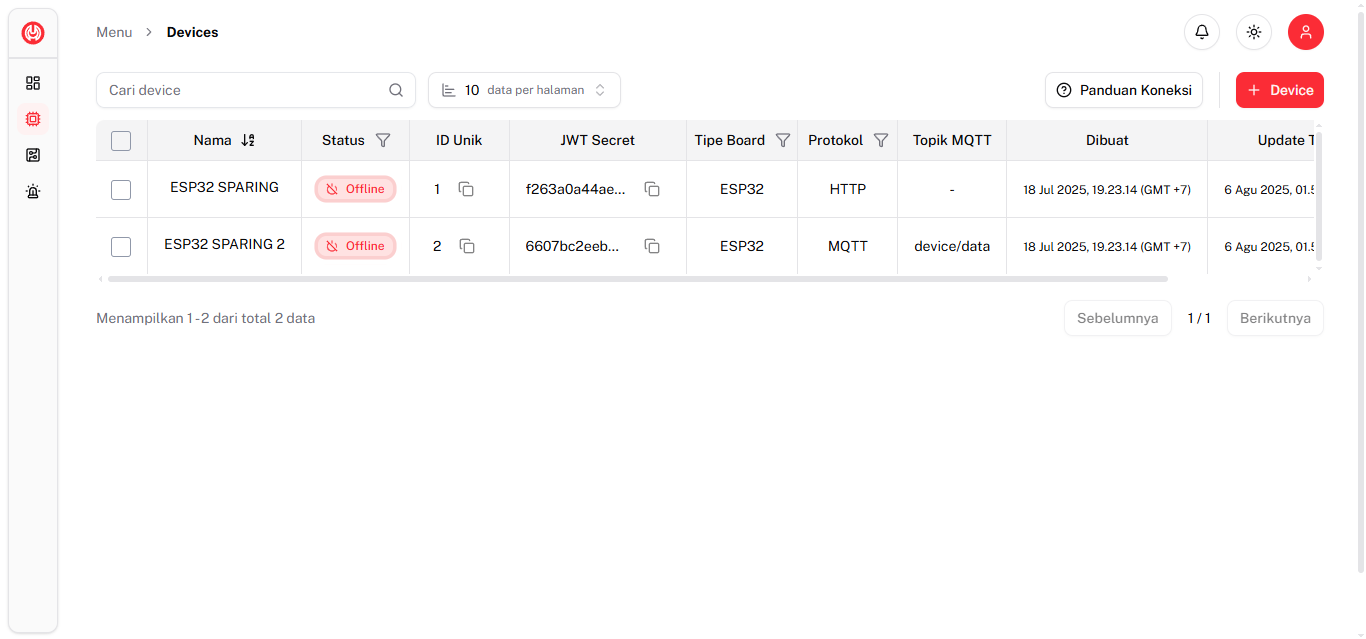
1. Halaman *Dashboards* Mode *Edit*



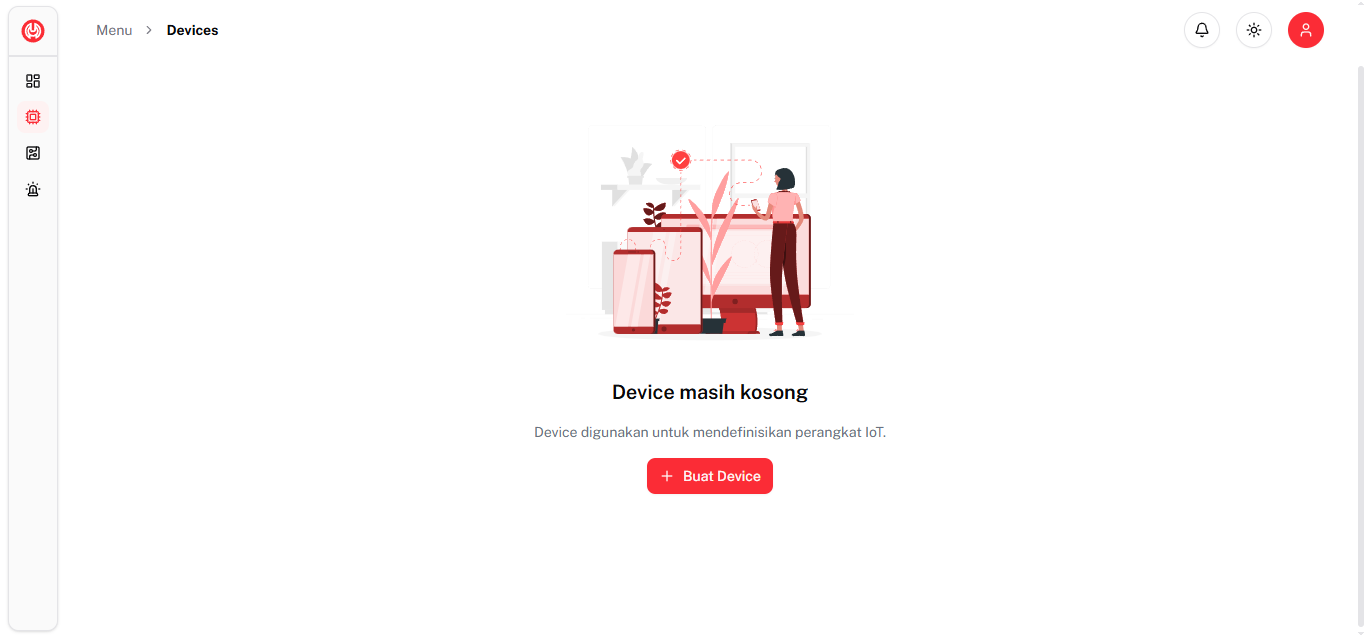
1. Halaman *Dashboards* Kosong



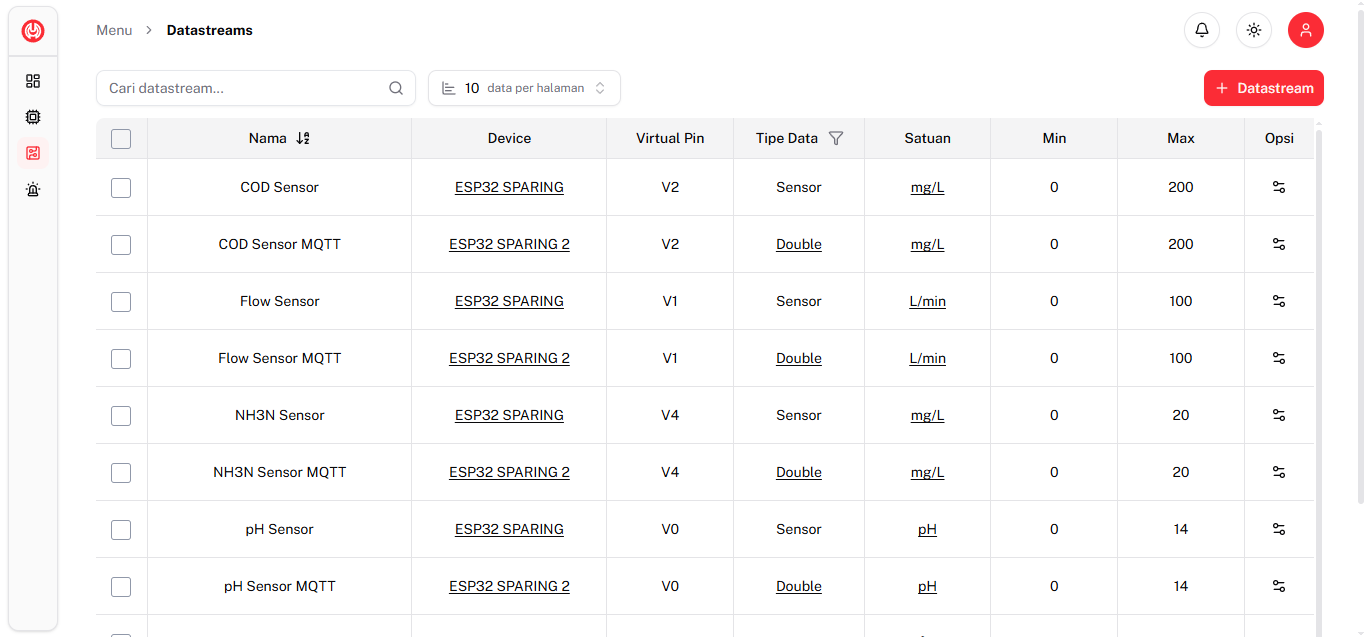
1. Halaman *Devices*



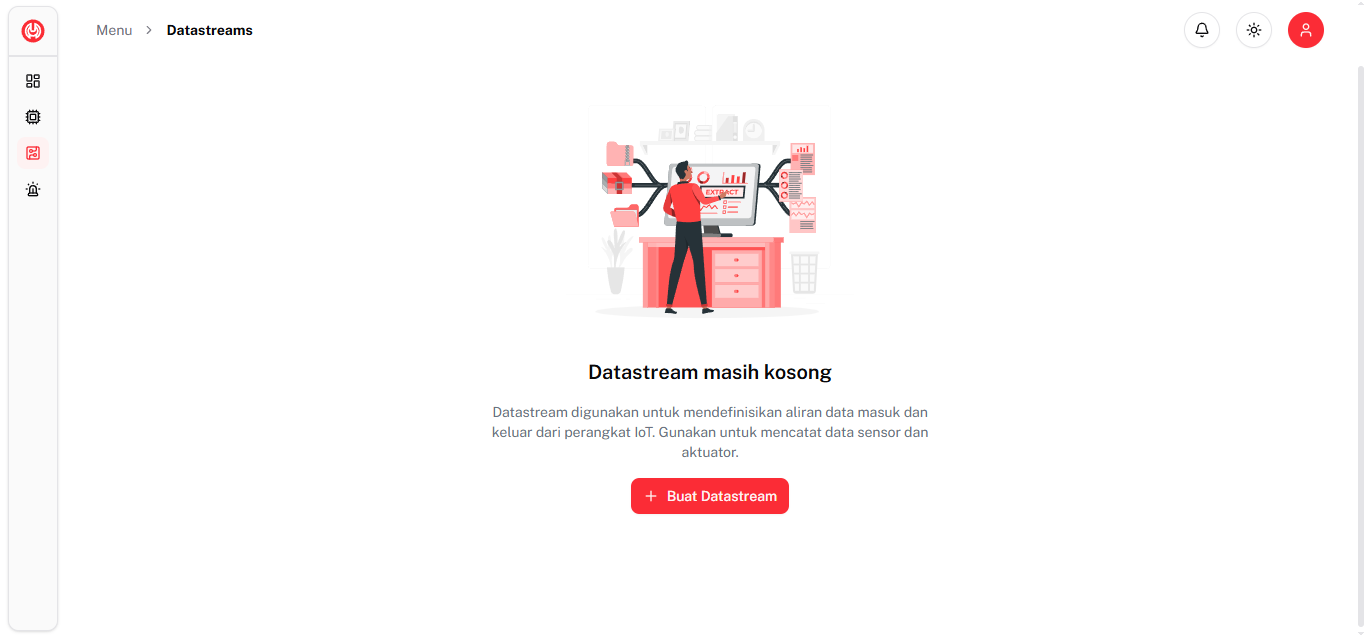
1. Halaman *Devices* Kosong



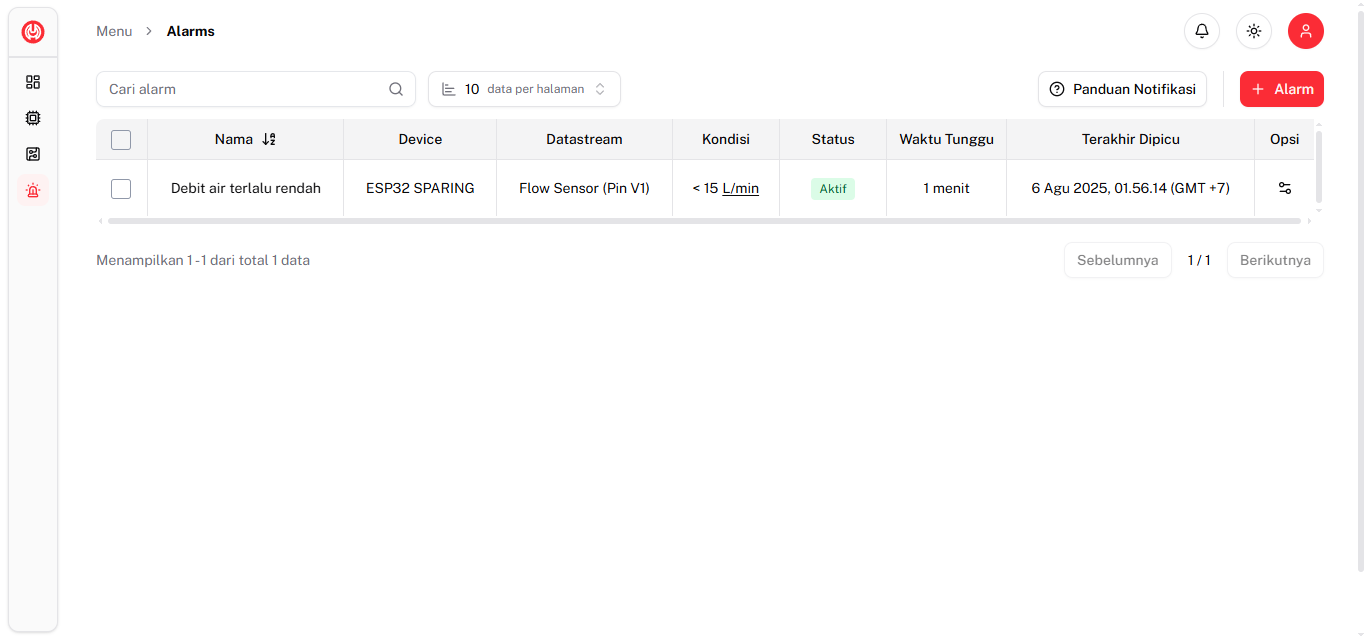
1. Halaman *Datastreams*



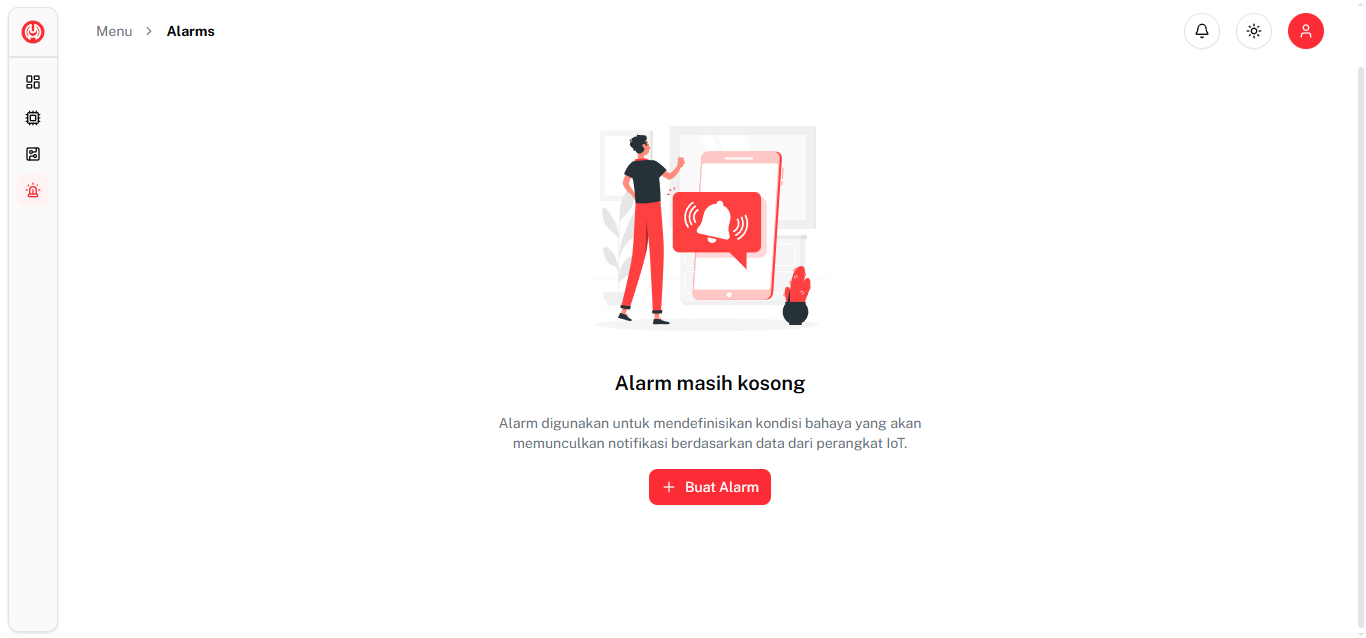
1. Halaman *Datastreams* Kosong



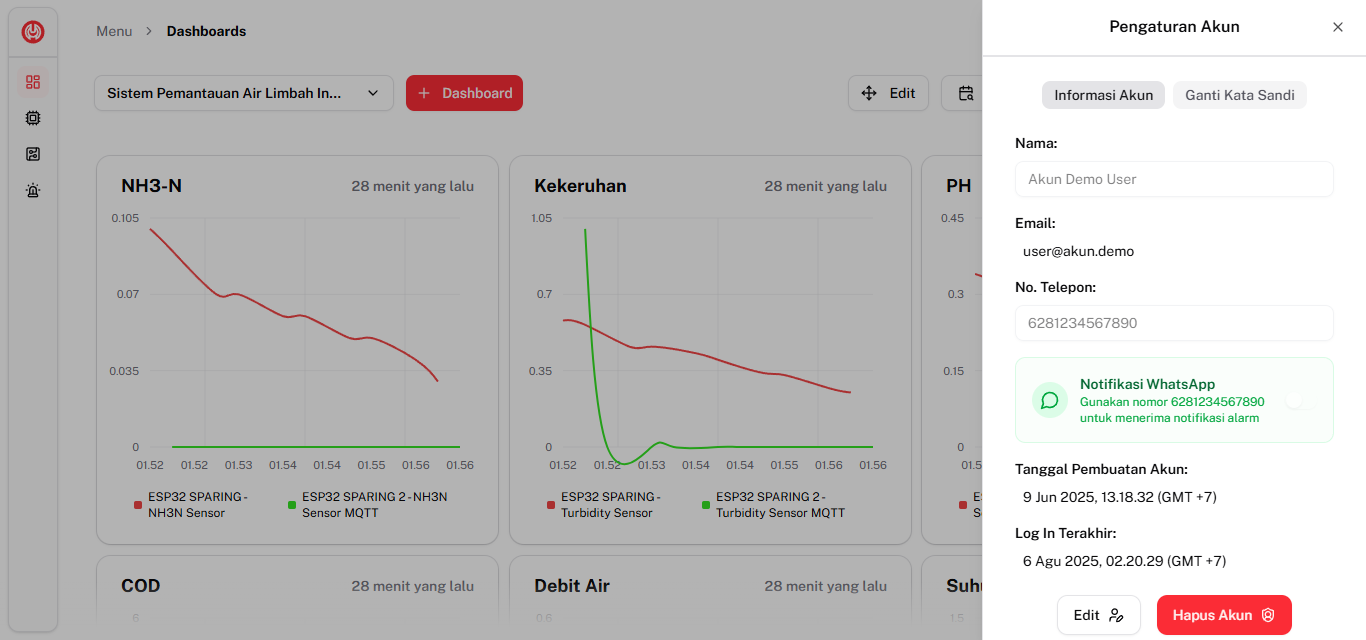
1. Halaman *Alarms*



1. Halaman *Alarms* Kosong



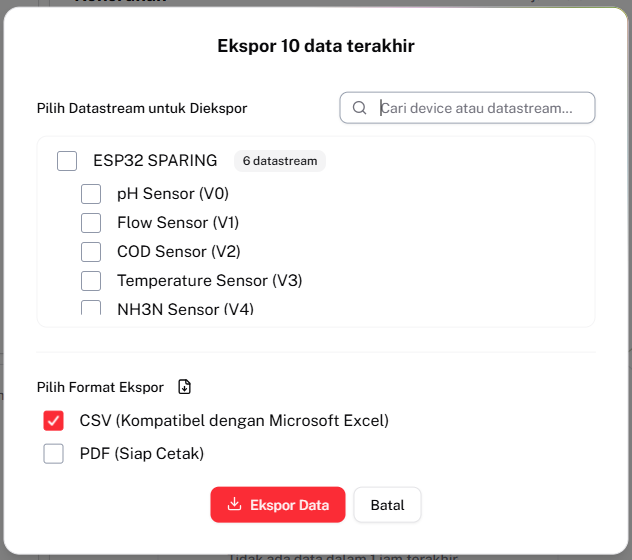
1. Tampilan Pengaturan Akun



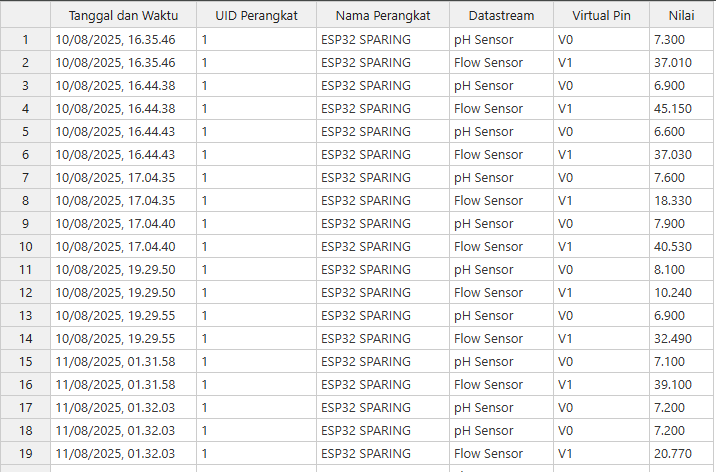
1. Tampilan Riwayat Notifikasi



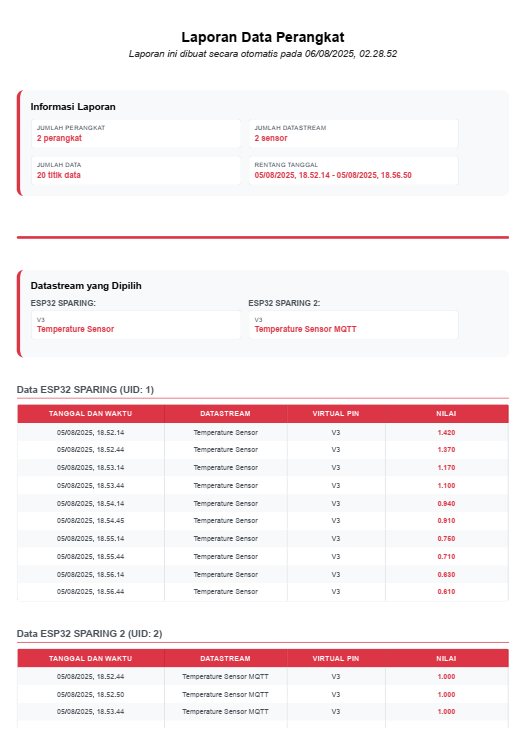
1. Tampilan Fitur Ekspor



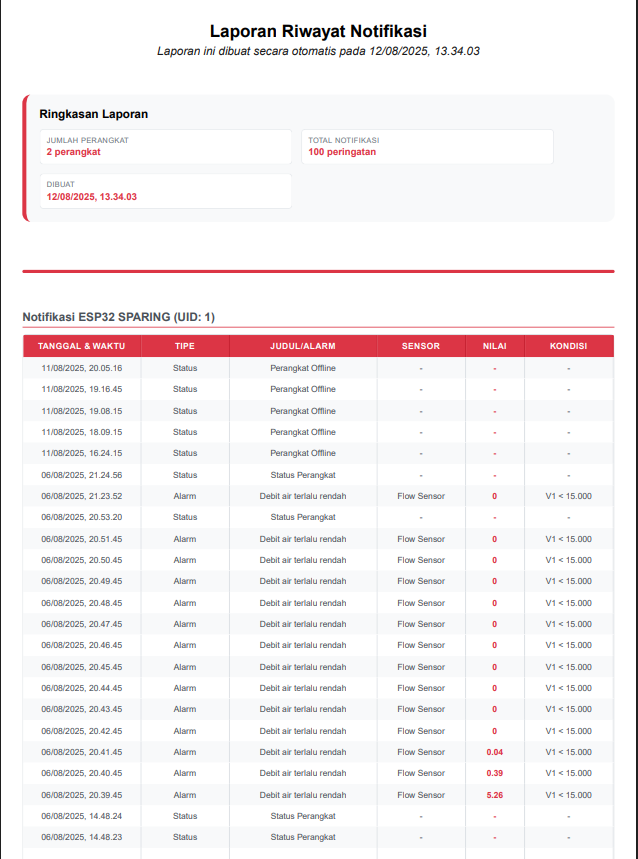
1. Hasil Ekspor CSV Data Perangkat



1. Hasil Ekspor PDF Data Perangkat



1. Hasil Ekspor Riwayat Notifikasi



1. Kode Dekripsi AES

