RANCANG BANGUN SISTEM MONITORING PERKEMBANGAN BERAT TERNAK BERBASIS WEB DARI TIMBANGAN TERNAK BERBASIS IOT

Rizqi Raffy Imam Malik, Raden Sumiharto*

[1]*Elektronika dan Instrumentasi, Universitas Gadjah Mada*
*rizqiraffyimammalik@mail.ugm.ac.id, r_sumiharto@ugm.ac.id**

Received: 2026-02-27 | Accepted: 2026-03-17 | Published: 2026-03-19

Abstrak

Penelitian ini bertujuan merancang dan mengimplementasikan sistem monitoring berat ternak berbasis web yang terintegrasi dengan perangkat timbangan berbasis Internet of Things (IoT). Sistem menggunakan arsitektur web–cloud–IoT dengan basis data terpusat untuk menjaga integritas dan konsistensi data. Perangkat berbasis ESP32 membaca identitas ternak melalui RFID dan nilai berat menggunakan sensor load cell, kemudian mengirimkan data dalam format JSON ke server untuk divalidasi sebelum disimpan pada PostgreSQL (Supabase). Aplikasi dikembangkan menggunakan Next.js dengan pendekatan full-stack yang mengintegrasikan autentikasi, manajemen sesi, pengelolaan data, dan mekanisme ingest dalam satu sistem. Pengendalian akses diterapkan melalui Supabase Auth dan kebijakan Row Level Security (RLS) untuk membatasi akses data berdasarkan kepemilikan pengguna. Endpoint ingest dilengkapi dengan verifikasi API key, validasi struktur data, serta pemeriksaan status perangkat sebelum pencatatan histori dilakukan. Pengujian meliputi uji fungsional, integrasi perangkat, uji beban pada rentang 40–95 request per second (RPS), serta evaluasi keamanan aplikasi. Hasil menunjukkan sistem beroperasi stabil hingga 90 RPS tanpa ditemukan kerentanan tingkat tinggi. Sistem ini berpotensi mendukung digitalisasi pengelolaan data pada sektor peternakan.

Kata kunci: Internet of Things (IoT); Monitoring Berat Ternak; Arsitektur Web–Cloud; Row Level Security (RLS); ESP32.

Abstract

This study aims to design and implement a web-based livestock weight monitoring system integrated with an Internet of Things (IoT) weighing device. The system adopts a web–cloud–IoT architecture with a centralized database to ensure data integrity and consistency. An ESP32-based device reads livestock identification via RFID and measures weight using a load cell sensor, then transmits data in JSON format to a server for validation before storage in a PostgreSQL (Supabase) database. The application is developed using a full-stack Next.js framework that integrates authentication, session management, data handling, and ingestion mechanisms within a single platform. Access control is implemented using Supabase Auth and Row Level Security (RLS) policies to restrict data access based on user ownership. The ingest endpoint performs API key verification, data structure validation, and device status checking prior to recording historical data. System evaluation includes functional testing, device integration testing, load testing at 40–95 requests per second (RPS), and security assessment. The results indicate stable performance up to 90 RPS without high-severity vulnerabilities. The proposed system has potential to support digital transformation in livestock data management.

Keywords: Internet of Things (IoT); Smart Livestock Monitoring; RFID-Based Identification; Database Security; Full-Stack Web Application.

1. Pendahuluan

Sektor peternakan memiliki peran strategis dalam mendukung ketahanan pangan dan perekonomian nasional. Di Kabupaten Sleman, Daerah Istimewa Yogyakarta, subsektor peternakan domba menunjukkan perkembangan yang signifikan, dengan populasi mencapai sekitar 36.953 ekor per Mei 2024 [1]. Peningkatan populasi tersebut diikuti dengan bertambahnya jumlah peternak, termasuk dari kalangan generasi muda, yang melihat usaha peternakan sebagai peluang ekonomi yang menjanjikan. Kondisi ini mendorong kebutuhan terhadap sistem pengelolaan data ternak yang lebih sistematis dan terdokumentasi secara baik.

Perkembangan berat ternak merupakan indikator kuantitatif utama dalam evaluasi pertumbuhan, penentuan waktu jual, serta estimasi nilai ekonomi ternak. Namun, pada praktik peternakan rakyat, pencatatan hasil penimbangan masih banyak dilakukan secara manual menggunakan timbangan konvensional dan buku tulis. Metode tersebut rawan kesalahan pencatatan, kehilangan data, serta tidak mendukung pengelolaan histori secara terstruktur. Beberapa penelitian telah mengembangkan sistem penimbangan ternak berbasis *Internet of Things* (IoT) untuk otomatisasi pencatatan berat dan transmisi data ke server [2], [3]. Sistem tersebut menunjukkan bahwa integrasi sensor berat, mikrokontroler, dan platform digital mampu meningkatkan efisiensi pencatatan dibandingkan metode konvensional.

Selain aspek akuisisi data, penelitian lain menekankan pentingnya visualisasi dan monitoring berbasis web untuk mendukung pemantauan perkembangan ternak secara individual [4]. Arsitektur sistem berbasis IoT dengan integrasi load cell dan RFID memungkinkan pencatatan otomatis yang terdokumentasi secara historis [4], [5]. Implementasi load cell dengan modul HX711 terbukti layak untuk pengukuran berat ternak skala kecil dengan tingkat kesalahan yang rendah [6], sementara penggunaan RFID RC522 dinilai ekonomis dan memadai untuk identifikasi jarak dekat pada sistem IoT [7].

Pada sisi komunikasi data, protokol HTTP dinilai praktis dan stabil untuk sistem IoT skala kecil dibandingkan pendekatan yang memerlukan broker tambahan [8], [9]. Sementara itu, penggunaan platform cloud untuk penyimpanan data meningkatkan aksesibilitas dan efisiensi monitoring pertanian [10]. Tantangan implementasi IoT pada sektor peternakan meliputi keterbatasan infrastruktur, biaya komponen, dan literasi teknologi pengguna [11], sehingga sistem yang dikembangkan perlu mempertimbangkan aspek kesederhanaan, keterjangkauan, dan kemudahan adopsi.

Meskipun berbagai penelitian tersebut telah membuktikan kelayakan sistem penimbangan berbasis IoT, sebagian besar berfokus pada sisi perangkat dan akuisisi data. Penguatan lapisan aplikasi sebagai pusat pengelolaan histori, konsistensi relasional data, validasi ingest berlapis, serta kontrol akses berbasis kepemilikan masih relatif terbatas. Tanpa mekanisme pengelolaan yang terstruktur, data dari perangkat IoT berpotensi hanya menjadi arsip digital tanpa memberikan nilai analitik yang optimal.

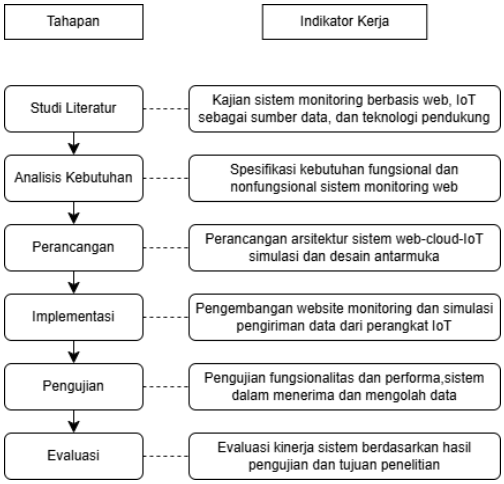
Berdasarkan kesenjangan tersebut, penelitian ini merancang dan mengimplementasikan sistem monitoring perkembangan berat ternak berbasis web yang terintegrasi dengan timbangan ternak berbasis IoT dan identifikasi RFID menggunakan arsitektur web–cloud–IoT. Sistem menempatkan aplikasi web sebagai pusat pengelolaan data relasional, menerapkan mekanisme validasi berlapis pada endpoint ingest, serta membatasi akses data berdasarkan kepemilikan pengguna. Pendekatan ini tidak hanya mengotomatisasi pencatatan berat ternak, tetapi juga memastikan integritas data, keterkaitan relasional antara ternak dan perangkat, serta kesiapan operasional untuk skala peternakan rakyat.

Kontribusi penelitian ini terletak pada integrasi lapisan perangkat, aplikasi, dan basis data dalam satu sistem monitoring terpadu yang menekankan konsistensi data dan kontrol akses, sehingga menghasilkan prototipe sistem yang sederhana, terjangkau, dan relevan untuk implementasi pada peternakan rakyat.

2. Metode Penelitian *(Methodology / Research Method)*

Penelitian ini menggunakan pendekatan rekayasa (*engineering research*) yang berfokus pada perancangan, implementasi, dan evaluasi sistem monitoring berbasis web yang terintegrasi dengan perangkat IoT. Pendekatan ini menempatkan sistem sebagai objek utama yang dirancang untuk memenuhi kebutuhan pengelolaan histori penimbangan ternak, menjaga keterkaitan data secara relasional, menerapkan kontrol akses berbasis kepemilikan, serta memastikan stabilitas integrasi perangkat IoT. Alur penelitian meliputi tahap perancangan arsitektur, implementasi sistem, dan pengujian teknis. Flowchart tahapan penelitian ditunjukkan pada Gambar 1.

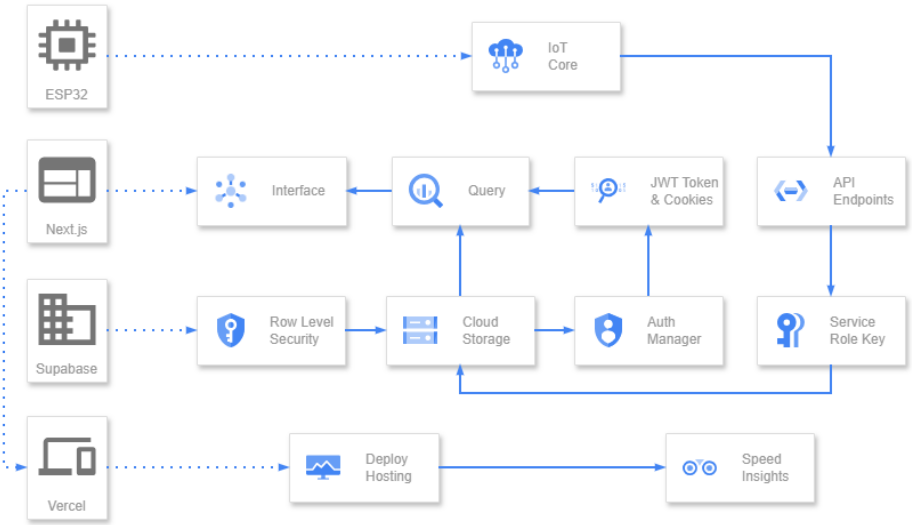
ISSN: 2252-4517
EISSN: 2723-7249



Gambar 1. Flowchart tahapan penelitian

2.1 Arsitektur Sistem

Arsitektur sistem menggunakan pendekatan web–*cloud*–IoT dengan pemisahan peran antara perangkat sebagai sumber data, lapisan aplikasi sebagai pengelola logika dan kontrol, serta basis data sebagai penyimpanan terpusat. Diagram arsitektur sistem ditunjukkan pada Gambar 1.



Gambar 1. Diagram arsitektur sistem web-cloud-IoT

Perangkat IoT mengirimkan data penimbangan melalui protokol HTTP ke lapisan aplikasi. Seluruh proses validasi dan pengendalian akses dilakukan pada lapisan aplikasi sebelum data disimpan ke basis data relasional. Penyimpanan menggunakan PostgreSQL untuk menjaga konsistensi hubungan antara entitas pengguna, ternak, perangkat, dan histori penimbangan. Interaksi data tidak dilakukan langsung dari klien ke database, melainkan melalui fungsi *server-side* dan API Route.

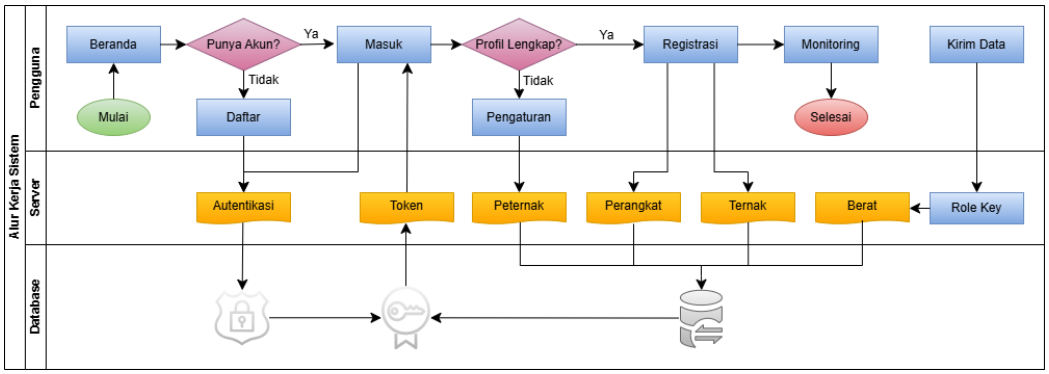
2.2 Website Monitoring

Website monitoring dikembangkan sebagai *full-stack web application* menggunakan Next.js dengan pendekatan App Router. Pada arsitektur ini, komponen frontend dan backend berada dalam satu aplikasi yang sama, sehingga halaman, API Route, dan logika server terintegrasi dalam satu lingkungan.

Antarmuka dibangun menggunakan komponen React dan CSS untuk menjaga konsistensi tampilan serta responsivitas. Akses data dilakukan melalui fungsi *server-side* sehingga klien tidak memiliki akses langsung ke basis data. Dengan pendekatan ini, website berfungsi sebagai pusat interaksi pengguna sekaligus pengendali akses terhadap data *backend*.

2.2.1 Alur Interaksi Pengguna

Alur interaksi pengguna dirancang agar setiap data penimbangan memiliki keterkaitan relasional yang jelas sejak awal penggunaan sistem. Diagram alur ditunjukkan pada Gambar 2.

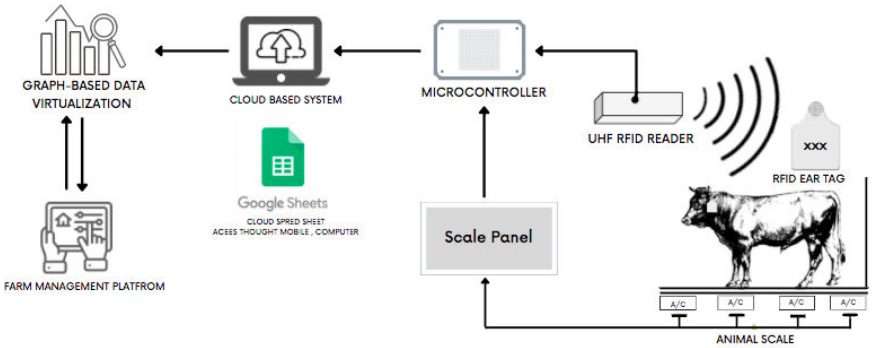


Gambar 2 Diagram alur interaksi pengguna dan sistem

Proses dimulai dari autentikasi pengguna. Setelah sesi terbentuk, sistem memeriksa kelengkapan profil peternak sebagai dasar kepemilikan data. Pengguna kemudian melakukan registrasi perangkat dan registrasi ternak menggunakan RFID. Kedua entitas tersebut menjadi prasyarat agar data penimbangan dapat diterima dan dikaitkan secara konsisten. Data yang tersimpan ditampilkan pada dashboard, log data, dan halaman detail ternak. Urutan penggunaan sistem berlangsung secara berurutan: autentikasi → kelengkapan profil → registrasi entitas → monitoring.

2.2.2 Mekanisme Ingest Data IoT

Mekanisme ingest dirancang sebagai satu-satunya jalur masuk data penimbangan melalui *endpoint* /api/iot/weights yang diimplementasikan sebagai API Route pada Next.js. Perangkat ESP32 mengirimkan payload JSON berisi RFID, nilai berat, identitas perangkat, dan waktu pencatatan melalui HTTP. Skema komunikasi ditunjukkan pada Gambar 3.



Gambar 3. Skema pengiriman data IoT ke server [5]

Pada *server-side*, *request* diproses melalui tahapan validasi berurutan. Tahap awal adalah verifikasi API key perangkat, validasi struktur dan tipe payload, pemeriksaan status perangkat, serta verifikasi relasi kepemilikan antara perangkat dan ternak. Jika salah satu tahap tidak terpenuhi, proses dihentikan dan data tidak disimpan. Jika seluruh prasyarat valid, data disimpan pada database menggunakan akses server. Pendekatan ini memastikan integritas relasional dan kontrol kepemilikan diterapkan sebelum interaksi dengan database.

2.2.3 Mekanisme Ingest Data IoT

Autentikasi berfungsi sebagai dasar pengendalian akses sekaligus pengikatan kepemilikan data antara identitas pengguna dan entitas domain sistem. Identitas akun dikelola menggunakan Supabase Auth pada skema auth.users, sedangkan data domain aplikasi disimpan terpisah pada skema *public*.

Metode autentikasi menggunakan email dan kata sandi dengan manajemen sesi berbasis *JSON Web Token (JWT)*. Token disimpan dalam bentuk *HTTP-only cookie* untuk mengurangi risiko pencurian melalui skrip klien. Setiap akses ke halaman terlindungi diverifikasi berdasarkan sesi aktif sebelum *query* dijalankan, sehingga validasi terjadi pada lapisan server dan tidak bergantung pada logika *frontend*.

2.2.4 Akses Data dan Pemrosesan Server-Side

Seluruh pengambilan data dilakukan melalui fungsi *server-side*. *Query* basis data dijalankan setelah sesi pengguna terverifikasi. Pola query disesuaikan dengan konteks akses, seperti mode pemilik berdasarkan user.id atau mode publik berdasarkan atribut is_public. Contoh *query* ditunjukkan pada Gambar 4.

```
export async function getWeightLogs(rfid) {
    const { supabase, user } = await getUserContext()
    if (!user) return []

    const { data } = await supabase
    .from("weights")
    .select(`id, rfid, weight, created_at, livestocks!inner (name, user_id)`)
    .eq("rfid", rfid)
    .eq("livestocks.user_id", user.id)
    .order("created_at", { ascending: false })

    return data ?? []
}
```

Gambar 4. Query server-side dengan join dan filter kepemilikan

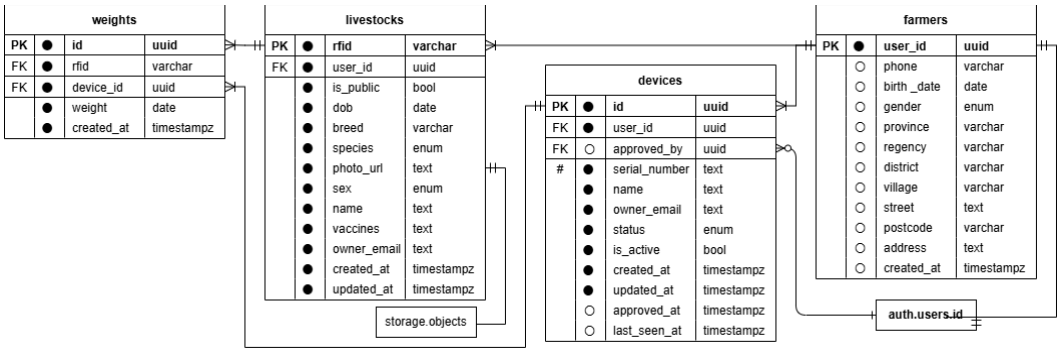
Hasil *query* diproses pada sisi server sebelum dikirim ke klien. Pemrosesan meliputi perhitungan umur ternak, klasifikasi tahap umur, serta evaluasi perubahan berat berdasarkan dua penimbangan terakhir. Pemrosesan terpusat memastikan konsistensi perhitungan dan menghindari duplikasi logika pada berbagai halaman.

2.3 Implementasi Cloud Database

Sistem menggunakan Supabase berbasis PostgreSQL sebagai *cloud database* relasional. Pemisahan antara data autentikasi dan data domain dilakukan untuk menjaga struktur sistem tetap terorganisasi dan aman.

2.3.1 Skema Relasional

Struktur relasi antar tabel ditunjukkan pada Gambar 4. Relasi antara auth.users dan farmers bersifat satu-ke-satu (1:1). Tabel farmers menjadi acuan kepemilikan bagi tabel livestocks dan devices (1:N). Tabel weights menyimpan histori penimbangan dan terhubung ke livestocks melalui rfid serta ke devices melalui device_id. Desain ini memastikan setiap data berat dapat ditelusuri ke ternak, perangkat, dan pemilik melalui constraint relasional.



Gambar 5. Entity relationship diagram sistem monitoring

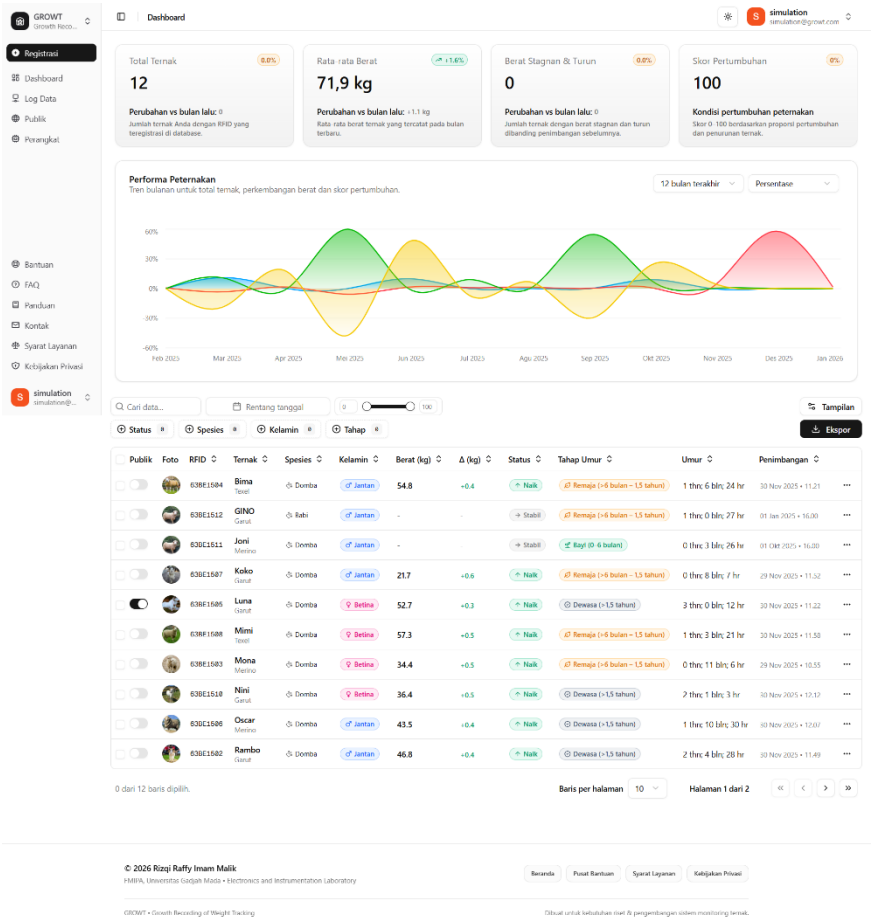
2.3.2 Implementasi Cloud Database

Row Level Security (RLS) diterapkan pada tabel domain untuk membatasi akses berdasarkan identitas sesi auth.uid(). Pengguna hanya dapat mengakses data yang terikat pada identitasnya. Akses publik

dimungkinkan untuk entitas tertentu tanpa mengubah struktur kepemilikan. Operasi *insert* pada tabel weights tidak disediakan melalui antarmuka web, melainkan hanya melalui *endpoint ingest* IoT menggunakan akses server. Validasi kepemilikan tetap dilakukan pada lapisan aplikasi sebelum proses penyimpanan.

2.4 Dashboard Monitoring

Setelah autentikasi berhasil, pengguna diarahkan ke dashboard sebagai titik masuk utama monitoring. Dashboard menampilkan indikator agregat, tren bulanan, dan daftar ternak. Perhitungan statistik dilakukan pada sisi server dengan mengambil catatan penimbangan terakhir tiap ternak dalam periode tertentu. Tampilan antarmuka dashboard ditunjukkan pada Gambar 5



Gambar 6. Tampilan antarmuka dashboard monitoring

Sistem juga menyediakan halaman publik, detail ternak per RFID, dan log penimbangan. Perbedaan antar halaman terletak pada ruang lingkup data yang ditampilkan, sedangkan mekanisme akses dan pemrosesan tetap mengikuti struktur sistem yang sama. Modul kontrol perangkat digunakan untuk mengelola status perangkat (*pending*, *active*, *inactive*, *revoked*). Status ini menjadi prasyarat operasional pada *endpoint ingest* sehingga perangkat yang tidak valid akan ditolak sebelum proses penyimpanan.

2.5 Deployment pada Lingkungan Produksi

Website monitoring di-*deploy* menggunakan Vercel yang mendukung Next.js dan eksekusi API Route sebagai *serverless function*. Proses build berjalan otomatis setiap terjadi pembaruan repositori dan hasilnya didistribusikan melalui *Content Delivery Network* (CDN).

Integrasi Supabase dan pengamanan endpoint IoT dikonfigurasi melalui *environment variable*. Variabel dengan prefiks NEXT_PUBLIC_ tersedia pada sisi klien, sedangkan kredensial sensitif seperti SUPABASE_SERVICE_ROLE_KEY dan IOT_API_KEY hanya tersedia pada sisi server untuk mencegah eksposur kunci akses pada browser.

3. Hasil dan Pembahasan (Results and Discussion)

Pengujian dilakukan untuk memastikan sistem bekerja sesuai rancangan pada aspek fungsional, integritas data, stabilitas endpoint ingest, performa produksi, serta konfigurasi keamanan. Evaluasi disusun mengikuti lapisan arsitektur web–cloud–IoT sehingga setiap mekanisme dapat dianalisis berdasarkan perannya masing-masing.

3.1 Metode Pengujian

Metode pengujian dibagi berdasarkan jenis mekanisme yang diuji, yaitu API dan kontrol akses, integrasi perangkat IoT, kapasitas endpoint ingest, performa frontend pada produksi, serta konfigurasi keamanan. Pendekatan ini digunakan agar setiap hasil pengujian dapat diinterpretasikan secara terpisah dan tidak saling tumpang tindih. Ringkasan metode pengujian disajikan pada Tabel 1.

Tabel 1 Metode pengujian

Metode	Target	Tujuan
Postman	Endpoint API	Validasi autentikasi dan RLS
ESP32	Endpoint ingest	Verifikasi integrasi perangkat
k6 by Grafana	Endpoint ingest	Uji beban dan stabilitas
Lighthouse & Speed Insights	Domain produksi	Evaluasi performa frontend
OWASP ZAP	Domain produksi	Audit konfigurasi keamanan

Setiap metode digunakan sesuai karakteristik lapisan sistem. Pengujian tidak hanya memeriksa keberhasilan respons, tetapi juga konsistensi validasi, pembatasan akses, serta ketahanan layanan pada kondisi beban tertentu.

3.2 Pengujian API

Pengujian API dilakukan menggunakan Postman sebanyak 11.900 iterasi. Skenario dirancang untuk menguji autentikasi, operasi CRUD pada tabel domain, kontrol akses berbasis peran, serta mekanisme ingest IoT dalam kondisi valid dan tidak valid. Ringkasan hasil pengujian API disajikan pada Tabel 2.

Tabel 2. Hasil pengujian API

Komponen Uji	Iterasi	Hasil
Autentikasi	1000	Stabil
REST farmers	1000	Lulus
REST devices	1000	Lulus
REST livestocks	1000	Lulus
Role-based Access	1000	Konsisten
CRUD Admin	900	Lulus
Ingest IoT	7000	99,9% berhasil
Total	**11.900**	**Lulus Pengujian**

Autentikasi berhasil membentuk dan mengakhiri sesi sesuai konteks token. Kebijakan *Row Level Security* membatasi akses lintas akun sehingga data tidak dapat diakses tanpa kepemilikan yang sah, meskipun respons HTTP tetap 200 dalam bentuk hasil kosong. Constraint unik pada serial_number dan rfid menghasilkan respons konflik (409) saat terjadi duplikasi. Pada *endpoint ingest*, seluruh skenario kegagalan menghasilkan kode respons deterministik (401 atau 400), sedangkan satu kejadian 500 disebabkan *timeout* sementara dan tidak berulang pada pengujian berikutnya.

3.3 Pengujian Integrasi IoT

Integrasi diuji menggunakan lima perangkat ESP32 dengan variasi status operasional untuk mensimulasikan kondisi lapangan. Konfigurasi perangkat yang digunakan dalam pengujian ditunjukkan pada Gambar 7. Dua perangkat berada pada status aktif, sedangkan tiga lainnya berstatus *pending*, *inactive*, dan *revoked*. Ringkasan pengujian perangkat aktif disajikan pada Tabel 3.

Gambar 7. Perangkat ESP32 untuk pengujian integrasi IoT

Tabel 3 Hasil pengujian integrasi IoT

Skenario	Status	Hasil
Valid Input	201	Data tersimpan
API Key tidak valid	401	Ditolak autentikasi
Format/Skema tidak valid	400	Ditolak validasi
RFID tidak ditemukan	400	Ditolak relasi
Kepemilikan tidak sesuai	400	Ditolak verifikasi

Perangkat aktif hanya dapat menyimpan data ketika autentikasi API key valid, struktur *payload* sesuai, status perangkat aktif, dan relasi kepemilikan terpenuhi. Respons penolakan selalu muncul pada tahap validasi yang relevan sehingga alur pemeriksaan berjalan berurutan sesuai desain. Perangkat dengan status nonaktif tidak pernah mencapai tahap penyimpanan data. Validasi status dilakukan sebelum eksekusi operasi tulis sehingga tidak ditemukan skenario perangkat tidak aktif berhasil mengirim data ke sistem.

3.4 Pengujian Beban Endpoint IoT

Pengujian beban dilakukan menggunakan k6 dengan skenario *constant-arrival-rate* pada rentang 40–95 RPS selama ±20 menit. Parameter yang dianalisis meliputi tingkat keberhasilan dan distribusi latensi (p50, p90, p95) untuk menentukan batas stabilitas operasional. Ringkasan hasil pengujian beban disajikan pada Tabel 4.

Tabel 4 Hasil pengujian beban endpoint IoT

RPS	Request	Success Rate	P50 (s)	p90 (s)	p95 (s)	Keterangan
40	9.600	99,9%	1,32	1,53	2,08	Stabil
60	14.401	99,7%	1,33	1,47	1,89	Stabil
70	8.401	100%	1,32	1,40	1,48	Stabil
75	9.001	100%	1,32	1,38	1,44	Stabil
80	9.601	99,8%	1,32	1,38	1,42	Stabil
85	10.201	99,9%	1,32	1,55	2,65	Stabil
90	10.801	99,8%	1,32	1,42	1,74	Stabil*
95	11.125	79,4%	1,33	10,0	10,0	Overload

Sistem mempertahankan stabilitas hingga 90 RPS dengan p95 di bawah 2 detik. Pada 95 RPS terjadi lonjakan latensi dan penurunan tingkat keberhasilan akibat antrean dan timeout, yang mengindikasikan batas kapasitas pada konfigurasi saat pengujian. Dengan ambang stabil di sekitar 90 RPS, sistem diperkirakan mampu mendukung ratusan perangkat dengan interval pengiriman periodik. Estimasi kapasitas bergantung pada frekuensi pengiriman data dari setiap perangkat.

3.5 Pengujian Performa Produksi

Evaluasi performa dilakukan pada aplikasi yang telah di-*deploy* di Vercel untuk merepresentasikan kondisi operasional nyata. Pengukuran dilakukan melalui dua pendekatan, yaitu *real user monitoring* menggunakan Speed Insights dan audit sintetis menggunakan Google Lighthouse. Hasil pemantauan Speed Insights menunjukkan:

- *Real Experience Score (RES)* = 97
- *First Contentful Paint (FCP)* ≈ 2 detik
- *Largest Contentful Paint (LCP)* ≈ 2 detik
- *Interaction to Next Paint (INP)* < 100 ms

- *Cumulative Layout Shift (CLS)* = 0,04

Nilai tersebut menunjukkan waktu render awal berada dalam kategori cepat, respons interaksi sangat baik, serta pergeseran layout rendah. Dengan RES sebesar 97, pengalaman pengguna pada domain produksi dapat dikategorikan stabil dan responsif dalam penggunaan normal. Audit teknis menggunakan Lighthouse dilakukan pada mode Mobile dan Desktop. Ringkasan hasil ditunjukkan pada Tabel 5.

Tabel 5 Hasil audit lighthouse

Kelompok Halaman	Mobile	Desktop	Acc	BP	SEO
Halaman statis	79–99	100	90–100	100	100
Halaman dinamis	66	74	95	73	100
Perangkat / Kontrol	66	78	95	73	100
Pengaturan	61	44	66	77	91
Registrasi	45	74	66	77	91

Halaman statis menunjukkan performa sangat tinggi terutama pada Desktop yang mencapai skor 100. Penurunan skor pada halaman dinamis, khususnya pada Mobile, dipengaruhi oleh proses *initial data fetching*, *render* tabel dan grafik, serta proses *hydration* pada sisi klien.

Perbedaan antara hasil Lighthouse dan Speed Insights menunjukkan bahwa secara teknis masih terdapat ruang optimasi pada render awal, namun pengalaman pengguna nyata tetap berada pada kategori baik. Hal ini mengindikasikan bahwa beban dinamis tidak secara signifikan mengganggu penggunaan sistem pada skenario operasional normal.

3.6 Pengujian Keamanan

Pengujian keamanan dilakukan menggunakan OWASP ZAP untuk menilai konfigurasi *header* HTTP dan kebijakan keamanan pada domain produksi. Pemindaian difokuskan pada potensi risiko injeksi skrip, konfigurasi transport layer, dan eksposur endpoint autentikasi. Ringkasan temuan disajikan pada Tabel 6.

Tabel 6. Hasil pengujian keamanan

Kategori	Temuan	Interpretasi
Medium	CSP menggunakan unsafe-inline & unsafe-eval	Potensi risiko XSS
Medium	CSP wildcard directive	Direktif belum spesifik
Medium	Cross-domain resource	Resource dari CDN
Low	HSTS belum diterapkan	Perlu hardening HTTPS
Low	X-Content-Type-Options belum aktif	Pencegahan MIME sniffing
Informational	Endpoint autentikasi terdeteksi	Tidak ada eksposur kredensial

Tidak ditemukan kerentanan dengan tingkat risiko High. Temuan yang ada berkaitan dengan konfigurasi dan dapat diperkuat melalui pengetatan kebijakan keamanan tanpa menunjukkan adanya celah eksploitasi langsung.

3.7 Evaluasi dan Penyempurnaan Produk

Evaluasi dilakukan dengan mengintegrasikan hasil pengujian fungsional, integrasi perangkat, kapasitas layanan, performa produksi, dan keamanan untuk menilai kesiapan sistem secara keseluruhan. Ringkasan hasil disajikan pada Tabel 7.

Tabel 7. Hasil evaluasi sistem

Aspek	Metode	Hasil Utama	Status
Fungsional API	Postman	Respons konsisten; RLS & constraint unik berjalan	Lulus
Integrasi IoT	ESP32	Validasi sesuai kondisi	Lulus
Kapasitas Endpoint	k6 (40–95 RPS)	Stabil ≤90 RPS; degradasi pada 95 RPS	Stabil
Performa Produksi	Lighthouse & Speed Insights	Halaman statis optimal; dinamis moderat	Baik
Keamanan Dasar	OWASP ZAP	Tidak ada High severity	Aman

Secara umum, sistem bekerja sesuai rancangan. Kontrol akses berbasis kepemilikan berjalan konsisten, perangkat nonaktif tidak dapat menyimpan data, dan batas kapasitas operasional telah teridentifikasi pada pengujian beban. Performa produksi tergolong baik pada penggunaan nyata, sedangkan aspek keamanan tidak menunjukkan risiko kritis. Perbaikan yang direkomendasikan terbatas pada optimasi halaman dinamis dan penguatan konfigurasi keamanan.

4. Kesimpulan *(Conclusion)*

Penelitian ini menghasilkan sistem monitoring perkembangan berat ternak berbasis web yang terintegrasi dengan perangkat timbangan IoT melalui endpoint ingest berbasis HTTP. Arsitektur memisahkan lapisan perangkat, aplikasi, dan database sehingga validasi data, kontrol akses, dan integritas relasional dapat diterapkan secara berlapis.

Autentikasi terintegrasi dengan Supabase Auth serta kebijakan *Row Level Security* memastikan pembatasan akses berbasis kepemilikan berjalan konsisten. Mekanisme ingest memeriksa autentikasi perangkat, struktur data, status operasional, dan relasi kepemilikan sebelum data disimpan.

Pengujian menunjukkan sistem stabil hingga 90 RPS dengan latensi terkontrol dan mampu mendukung ratusan perangkat pada pengiriman periodik. Performa produksi berada pada kategori baik dan tidak ditemukan kerentanan keamanan tingkat tinggi. Sistem telah memenuhi tujuan penelitian dan layak dikembangkan lebih lanjut untuk skala operasional yang lebih besar.

Daftar Pustaka

[1] Antara, "DP3 Sleman sebut banyak muncul peternak domba dari generasi milenial," *Antaranews Yogyakarta*, 2024. [Online]. Available: https://jogja.antaranews.com/berita/690273/dp3-sleman-sebut-banyak-muncul-peternak-domba-dari-generasi-milenial

[2] T. Kushartadi and M. Asvial, "Design and Implementation of the Smart Weighing Precision Livestock Monitoring Technology Based on the Internet of Things (IoT)," *International Journal on Advanced Science Engineering and Information Technology*, vol. 13, no. 4, p. 1438, 2023. [Online]. Available: https://doi.org/10.18517/ijaseit.13.4.18557

[3] A. U. Rahayu, L. Faridah, N. Hiron, and F. M. S. Nursuwars, "Livestock Weighing System Using the Internet of Things (IoT) for Caribi Marketplace," *Advances in Biological Sciences Research*, pp. 233–243, 2023. [Online]. Available: https://doi.org/10.2991/978-94-6463-180-7_25

[4] C. Trilaksana, E. Akbartama, A. Muttaqin, and O. Setyawati, "Internet of Things-Based Cow Body Weight Recording System," *Jurnal EECCIS*, vol. 17, no. 1, pp. 8–12, 2023. [Online]. Available: https://doi.org/10.21776/jeeccis.v17i1.1632

[5] K. Noinan, S. Wicha, and R. Chaisricharoen, "The IoT-based Weighing System for Growth Monitoring and Evaluation of Fattening Process in Beef Cattle Farm," in *Proc. 2022 Joint Int. Conf. Digit. Arts, Media Technol.*, pp. 384–388, 2022. [Online]. Available: https://doi.org/10.1109/ECTIDAMTNCON53731.2022.9720346

[6] N. Hapsari, T. D. Indraswati, M. Haifan, and D. Maulana, "Digital Automatic Livestock Weighing System Using Single Beam Load Cell," *AIP Conference Proceedings*, 2019. [Online]. Available: https://doi.org/10.1063/1.5112391

[7] K. Ali et al., "Design of Adaptive RFID RC522 on IoT Platform with Different Types Passive Tag Based on Self-Service Library Management System (SSLMS)," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 33, no. 1, pp. 163–174, 2023. [Online]. Available: https://doi.org/10.37934/araset.33.1.163174

[8] E. Fitria, A. G. Putra, and M. Abdurohman, "Analisis perbandingan performansi MQTT dan HTTP pada platform IoT Node-Red," *eProceedings of Engineering*, vol. 6, no. 2, 2019. [Online]. Available: https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/9824

[9] K. Dineva and T. Atanasova, "Cloud Data-Driven Intelligent Monitoring System for Interactive Smart Farming," *Sensors*, vol. 22, no. 17, p. 6566, 2022. [Online]. Available: https://doi.org/10.3390/s22176566

[10] M. S. Farooq, O. O. Sohail, A. Abid, and S. Rasheed, "A Survey on the Role of IoT in Agriculture for the Implementation of Smart Livestock Environment," *IEEE Access*, 2022. [Online]. Available: https://doi.org/10.1109/access.2022.3142848