ASSETFLOW

Sistem Manajemen Aset & Reservasi

DOKUMENTASI TEKNIS RESMI

Versi 1.0.0

2026

1. Pendahuluan

AssetFlow adalah sistem manajemen aset dan reservasi berbasis web yang dirancang untuk memudahkan pengelolaan inventaris dan proses peminjaman barang secara efisien dan transparan. Sistem ini dibangun menggunakan teknologi React modern dan dikonsumsi melalui RESTful API dengan otentikasi berbasis JSON Web Token (JWT).

Dokumen ini merupakan panduan teknis resmi yang mencakup arsitektur sistem, alur kerja lengkap, panduan instalasi, referensi API, dan kebijakan keamanan. Dokumen ditujukan untuk developer, administrator teknis, dan pemangku kepentingan proyek.

1.1 Tujuan Sistem

- Menyediakan inventarisasi aset digital yang terpusat dan dapat diandalkan.
- Mengotomatisasi proses Check-out dan Check-in aset antar departemen.
- Memberikan visibilitas real-time atas lokasi, kondisi, dan ketersediaan aset.
- Menghasilkan laporan dan analitik untuk mendukung pengambilan keputusan manajemen.

1.2 Ruang Lingkup

Sistem AssetFlow mencakup dua peran pengguna utama: Administrator dengan akses penuh terhadap seluruh fitur manajemen, dan User/Staff yang dapat melakukan pencarian aset dan pengajuan reservasi. Sistem tidak mencakup manajemen keuangan atau integrasi ERP secara langsung.

2. Arsitektur Sistem

AssetFlow mengadopsi arsitektur Single Page Application (SPA) yang berkomunikasi dengan backend melalui RESTful API. Pemisahan yang jelas antara lapisan presentasi (frontend) dan logika bisnis (backend) memudahkan skalabilitas dan pemeliharaan independen.

2.1 Komponen Arsitektur

Frontend Layer	:	React 18 SPA (dibangun dengan Rsbuild)
API Layer	:	RESTful API dengan autentikasi JWT
Data Layer	:	Database relasional via API endpoint
Build & Deploy	:	Rsbuild bundler + hosting statis

2.2 Struktur Direktori Proyek

```
src/
├── assets/          # Gambar, ikon, dan file statis
├── component/       # Komponen UI reusable
│   ├── Sidebar.jsx
│   ├── Navbar.jsx
│   └── [komponen lain]
├── hooks/           # Custom React hooks
│   ├── useAuth.js
│   └── useAssets.js
├── pages/           # Halaman utama aplikasi
│   ├── Dashboard/
│   ├── Assets/
│   ├── Reservations/
│   └── Users/
├── utils/           # Fungsi utilitas & Route Guards
│   ├── axiosInstance.js
│   └── PrivateRoute.jsx
└── App.jsx          # Konfigurasi rute utama
```

2.3 Stack Teknologi

Teknologi	Versi	Peran dalam Sistem
React 18	18.x	Framework UI utama — komponen, state, dan routing
Rsbuild	Latest	Build tool berbasis Rust, pengganti Vite untuk performa maksimal
GSAP	3.x	Animasi micro-interaction dan transisi halaman yang halus

Teknologi	Versi	Peran dalam Sistem
Chart.js	4.x	Visualisasi data statistik pada Dashboard Analytics
Axios	1.x	HTTP client dengan interceptors untuk otentikasi JWT
Lucide React	Latest	Sistem ikon SVG yang konsisten dan ringan
SweetAlert2	11.x	Dialog konfirmasi dan notifikasi interaktif
Biome	Latest	Linting dan formatting kode ultra-cepat (pengganti ESLint)

3. Alur Kerja Sistem (Application Flow)

Bagian ini menjelaskan siklus hidup lengkap operasi AssetFlow, dari autentikasi pengguna hingga analitik laporan. Setiap fase dirancang untuk menjamin integritas data dan transparansi proses.

3.1 Fase 1 — Autentikasi & Otorisasi

Setiap sesi pengguna dimulai dari proses autentikasi yang ketat. Sistem menggunakan JWT (JSON Web Token) sebagai mekanisme keamanan stateless yang memungkinkan verifikasi identitas tanpa menyimpan sesi di server.

> Detail Teknis: Token JWT disimpan di localStorage/sessionStorage browser dan disertakan secara otomatis pada setiap request API via Axios interceptors. Token kadaluarsa akan memicu redirect ke halaman login secara otomatis.

1. Pengguna membuka aplikasi dan diarahkan ke halaman Login.
2. Sistem mengirim kredensial (email + password) ke endpoint POST /api/auth/login.
3. Backend memvalidasi kredensial dan mengembalikan JWT token beserta data profil pengguna.
4. Axios interceptor menyimpan token dan menyertakannya pada header Authorization: Bearer <token> untuk semua request berikutnya.
5. Sistem membaca field role dari payload JWT untuk menentukan wewenang: Admin atau User/Staff.
6. Pengguna diarahkan ke Dashboard yang sesuai dengan perannya.
7. Route Guard (PrivateRoute) memproteksi setiap halaman. Akses tanpa token atau dengan token kadaluarsa akan dialihkan ke Login.

3.2 Fase 2 — Setup Data Master (Admin)

Sebelum aset dapat diinput, Administrator harus menyiapkan data referensi yang menjadi fondasi konsistensi data di seluruh sistem. Data master ini bersifat terpusat dan digunakan sebagai pilihan dropdown pada form-form lain.

- Kategori Aset: Klasifikasi aset (contoh: Elektronik, Furnitur, Kendaraan). Digunakan untuk filtering dan pelaporan.
- Lokasi: Gedung, lantai, atau ruangan tempat aset ditempatkan. Mendukung tracking lokasi fisik aset.
- Vendor: Data pemasok atau mitra servis. Digunakan pada pencatatan pengadaan dan riwayat servis aset.

Setiap entitas data master memiliki operasi CRUD penuh yang hanya dapat diakses oleh Administrator. Perubahan pada data master tidak bersifat retroaktif — aset yang sudah terdaftar menggunakan nilai lama tidak akan terpengaruh secara otomatis.

3.3 Fase 3 — Registrasi & Inventarisasi Aset

Proses inventarisasi adalah inti dari AssetFlow. Setiap unit fisik yang masuk ke dalam sistem menerima identitas digital unik yang memungkinkan pelacakan individual.

8. Admin membuka menu Assets > Tambah Aset Baru.
9. Formulir diisi dengan detail: Nama Aset, Kategori, Lokasi, Vendor, Merk, Nomor Seri, Harga Perolehan, dan Tanggal Pengadaan.
10. Sistem backend menghasilkan Unique ID (UUID) untuk setiap aset yang terdaftar.
11. Status aset secara default diset ke Tersedia (Available).
12. Data aset tersimpan dan langsung terlihat di halaman daftar inventaris.

Penting: Nomor Seri bersifat unik per aset. Sistem akan menolak duplikasi Nomor Seri untuk menjaga integritas data inventaris.

3.4 Fase 4 — Proses Reservasi (Check-out & Check-in)

Alur reservasi adalah jantung operasional harian AssetFlow. Proses ini melibatkan dua aktor: User/Staff yang membuat permintaan dan Administrator yang melakukan otorisasi.

Alur Pengajuan (User/Staff):

13. User mencari aset yang dibutuhkan melalui fitur Discovery & Search (filter berdasarkan kategori, nama, atau lokasi).
14. User memilih aset dengan status Tersedia dan mengklik Ajukan Peminjaman.
15. User mengisi formulir reservasi: tanggal mulai, tanggal selesai, dan tujuan penggunaan.
16. Sistem mengirim notifikasi permintaan kepada Administrator.
17. Status aset berubah menjadi Menunggu Persetujuan (Pending).

Alur Otorisasi (Administrator):

18. Admin menerima notifikasi permintaan baru di dashboard.
19. Admin meninjau detail permintaan: pemohon, aset, dan periode pemakaian.
20. Admin dapat Menyetujui (Approve) atau Menolak (Reject) permintaan.
21. Jika disetujui: status aset berubah ke Dipinjam, log Check-out dibuat, dan notifikasi dikirim ke User.
22. Jika ditolak: status kembali ke Tersedia, dan alasan penolakan dikirimkan ke User.

Proses Pengembalian (Check-in):

23. User menyelesaikan penggunaan dan melaporkan pengembalian aset.
24. User memperbarui kondisi aset pasca penggunaan (Baik / Perlu Servis / Rusak).
25. Admin mengkonfirmasi penerimaan fisik dan menutup siklus reservasi.
26. Status aset kembali ke Tersedia dan riwayat transaksi tersimpan di log.

3.5 Fase 5 — Manajemen Kondisi & Pemeliharaan

Sistem menyediakan mekanisme untuk melacak kondisi fisik setiap aset sepanjang masa pakainya, memastikan aset selalu dalam kondisi layak operasional.

- Baik (Good): Aset dalam kondisi prima, siap digunakan.
- Butuh Servis (Needs Service): Aset masih berfungsi tetapi memerlukan pengecekan rutin atau perbaikan minor. Aset tetap dapat dipinjam.
- Rusak (Damaged): Aset tidak berfungsi dan dikecualikan dari sistem reservasi hingga selesai diperbaiki.
- Dalam Perbaikan (Under Maintenance): Aset sedang dalam proses servis oleh vendor. Tidak tersedia untuk dipinjam.

Setiap perubahan kondisi tercatat dalam log aset beserta timestamp dan identitas pengguna yang melakukan pembaruan, menciptakan jejak audit yang lengkap.

3.6 Fase 6 — Dashboard Analytics & Pelaporan

Dashboard Analytics menyediakan visibilitas menyeluruh atas kondisi inventaris dalam bentuk visualisasi data yang interaktif, membantu manajemen dalam pengambilan keputusan strategis.

- Total Nilai Aset: Kalkulasi agregat dari nilai pengadaan seluruh aset aktif.
- Distribusi Lokasi: Diagram donut atau bar yang menampilkan sebaran aset per lokasi/gedung.
- Status Pemeliharaan: Proporsi aset berdasarkan kondisi (Baik, Servis, Rusak).
- Tren Reservasi: Grafik garis yang menampilkan volume peminjaman per periode.
- Aset Paling Sering Dipinjam: Ranking aset berdasarkan frekuensi penggunaan.
- Permintaan Pending: Jumlah reservasi yang menunggu otorisasi Admin.

4. Manajemen Pengguna & Peran

AssetFlow menerapkan sistem Role-Based Access Control (RBAC) yang membagi wewenang secara tegas antara dua peran: Administrator dan User/Staff.

Fitur	Administrator	User / Staff
Dashboard Analytics	Akses penuh	Terbatas (aset sendiri)
Manajemen Aset (CRUD)	Ya	Tidak
Data Master (Kategori, Lokasi, Vendor)	Ya	Tidak
Request Reservasi (Check-out)	Ya (otorisasi)	Ya (pengajuan)
Update Kondisi Aset	Ya	Ya (laporan kerusakan)
Manajemen Pengguna	Ya	Tidak
Riwayat Peminjaman	Semua pengguna	Milik sendiri
Ekspor Laporan	Ya	Tidak

4.1 Manajemen Akun (Admin)

- Membuat akun staff baru dengan email dan peran yang ditetapkan.
- Mengaktifkan atau menonaktifkan akun yang tidak lagi digunakan.
- Mengubah peran pengguna (promosi User menjadi Admin atau sebaliknya).
- Melihat riwayat aktivitas seluruh pengguna untuk keperluan audit.

4.2 Profil Pengguna (User/Staff)

- Melihat dan memperbarui informasi profil pribadi.
- Mengubah password secara mandiri.
- Melihat riwayat peminjaman pribadi dan status aset yang sedang dipinjam.

5. Instalasi & Konfigurasi

5.1 Prasyarat Sistem

- Node.js versi 18 atau lebih tinggi
- NPM versi 8 atau lebih tinggi (atau Yarn 1.x)
- Akses internet untuk instalasi dependensi
- Backend API aktif dan dapat diakses (lihat konfigurasi environment)

5.2 Langkah Instalasi

27. Clone repositori ke direktori lokal:

```
git clone <url-repository>
cd finalexam-frontend
```

28. Instal seluruh dependensi proyek:

```
npm install
```

29. Buat file konfigurasi environment di root direktori:

```
.env
PUBLIC_API_URL=https://api-reservations-production.up.railway.app
```

30. Jalankan server pengembangan:

```
npm run dev
```

Aplikasi akan berjalan di http://localhost:3000 (atau port yang dikonfigurasi Rsbuild).

5.3 Build untuk Produksi

```
npm run build
```

Output build akan tersimpan di direktori dist/ dan siap untuk dideploy ke server statis atau CDN. Pastikan server dikonfigurasi untuk melakukan redirect semua request ke index.html agar client-side routing berfungsi dengan benar.

5.4 Skrip yang Tersedia

Perintah	Deskripsi
npm run dev	Menjalankan server pengembangan dengan hot reload otomatis
npm run build	Membangun bundle produksi yang dioptimalkan di folder dist/
npm run preview	Menjalankan preview lokal dari hasil build produksi
npm run format	Merapikan seluruh kode menggunakan Biome formatter

Perintah	Deskripsi
npm run test	Menjalankan suite unit test menggunakan Rstest

6. Referensi API

Berikut adalah daftar endpoint API utama yang dikonsumsi oleh AssetFlow. Base URL dikonfigurasi melalui variabel environment PUBLIC_API_URL.

Method	Endpoint	Akses	Deskripsi
POST	/api/auth/login	Publik	Login dan dapatkan JWT token
POST	/api/auth/logout	Semua	Invalidasi sesi aktif
GET	/api/assets	Semua	Daftar semua aset (dengan filter)
POST	/api/assets	Admin	Tambah aset baru ke inventaris
PUT	/api/assets/:id	Admin	Perbarui data aset
DELETE	/api/assets/:id	Admin	Hapus aset dari sistem
GET	/api/categories	Semua	Daftar kategori aset
POST	/api/reservations	User/Admin	Buat permintaan Check-out
PATCH	/api/reservations/:id	Admin	Setujui atau tolak reservasi
GET	/api/dashboard/stats	Admin	Data statistik untuk dashboard
GET	/api/users	Admin	Daftar pengguna terdaftar
PATCH	/api/users/:id/role	Admin	Ubah peran pengguna

6.1 Autentikasi Request

Semua endpoint yang memerlukan autentikasi harus menyertakan JWT token pada header HTTP sebagai berikut:

```
Authorization: Bearer <jwt_token>
```

Axios interceptor pada file utils/axiosInstance.js mengelola penambahan header ini secara otomatis untuk setiap request yang keluar dari aplikasi.

6.2 Format Respons

Seluruh endpoint API mengembalikan respons dalam format JSON dengan struktur standar berikut:

```
{
```

```
  "status": "success" | "error",
  "message": "Pesan deskriptif",
  "data": { ... }  // Payload respons (jika berhasil)
  "errors": [ ... ] // Detail kesalahan (jika gagal)
}
```

7. Keamanan & Optimasi

7.1 Keamanan Autentikasi

- JWT Authentication: Semua request terproteksi menggunakan token JWT yang diterbitkan oleh backend setelah login berhasil.
- Token Expiry: Token memiliki masa berlaku terbatas. Aplikasi mendeteksi respons 401 Unauthorized dan secara otomatis mengalihkan pengguna ke halaman login.
- Axios Interceptors: Implementasi terpusat di axiosInstance.js memastikan konsistensi penanganan autentikasi di seluruh aplikasi.

7.2 Route Guarding

- PrivateRoute: Komponen wrapper yang memverifikasi keberadaan dan validitas token sebelum merender halaman yang diproteksi.
- Role-Based Routing: Halaman tertentu seperti Manajemen Pengguna dan Dashboard Analytics hanya dapat diakses oleh pengguna dengan role Admin.
- Redirect Logic: Pengguna yang tidak terautentikasi akan diarahkan ke /login; pengguna yang tidak memiliki wewenang akan diarahkan ke /unauthorized.

7.3 Optimasi Performa

- Rsbuild (berbasis Rust): Build tool dengan kecepatan kompilasi yang jauh lebih tinggi dibandingkan Webpack atau Vite konvensional.
- Code Splitting: Setiap halaman dimuat secara lazy (React.lazy + Suspense) sehingga ukuran bundle awal diminimalkan.
- Biome Linting: Mendeteksi potensi bug dan masalah performa pada tahap pengembangan.
- Responsive Layout: Antarmuka dioptimalkan untuk tampilan mobile, tablet, dan desktop menggunakan CSS Grid dan Flexbox.

7.4 Penanganan Error

- Global Error Handler via Axios interceptor menangkap error jaringan dan respons HTTP 4xx/5xx secara terpusat.
- SweetAlert2 digunakan untuk menampilkan notifikasi error yang informatif dan dialog konfirmasi sebelum operasi destruktif (hapus, nonaktifkan akun).
- Validasi form sisi klien dilakukan sebelum mengirim request ke API untuk mengurangi beban server.

8. Panduan Kontribusi

8.1 Konvensi Kode

- Gunakan Biome untuk linting dan formatting sebelum setiap commit: npm run format.
- Penamaan komponen menggunakan PascalCase; fungsi dan variabel menggunakan camelCase.
- Setiap komponen baru harus disertai prop-types atau JSDoc untuk dokumentasi.
- Hindari logika bisnis di dalam komponen UI — pisahkan ke custom hooks atau utils.

8.2 Git Workflow

- Gunakan branch terpisah untuk setiap fitur: feature/nama-fitur atau fix/nama-bug.
- Nama commit mengikuti format Conventional Commits: feat:, fix:, docs:, refactor:.
- Pull Request wajib melalui code review minimal satu reviewer sebelum di-merge ke main.