

BAB 3

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama program magang, peserta ditempatkan pada Tim Teknologi Informasi yang berada di bawah naungan Direktur Teknik. Direktur Teknik memiliki tanggung jawab dalam memastikan seluruh operasional teknologi, sistem, serta infrastruktur digital di lingkungan instansi berjalan dengan baik dan mendukung kebutuhan kerja setiap divisi. Sebagai bagian dari Tim Teknologi Informasi, unit ini berperan dalam pengelolaan infrastruktur teknologi, pengembangan dan pemeliharaan sistem, serta memastikan kelancaran operasional perangkat lunak dan perangkat keras.

Dalam pelaksanaan tugas sehari-hari, peserta magang mendapatkan bimbingan langsung dari Bapak Rahmat, yang bertugas sebagai koordinator tim serta *supervisor* kegiatan magang. Beliau memberikan arahan terkait pelaksanaan pekerjaan, memantau progres harian, dan memastikan setiap tugas sesuai dengan kebutuhan teknis. Selain itu, terdapat rapat evaluasi mingguan yang dilakukan bersama tim untuk melaporkan perkembangan pekerjaan, membahas kendala, dan menentukan langkah kerja selanjutnya.

Proses koordinasi harian juga dilakukan melalui grup WhatsApp tim, sehingga komunikasi dapat berjalan dengan cepat dan efektif. Dengan pola bimbingan dan koordinasi tersebut, peserta magang dapat melaksanakan tugas dengan baik serta memahami alur kerja pada lingkungan teknologi informasi.

3.2 Tugas yang Dilakukan

Selama mengikuti program magang, peserta mendapatkan penugasan utama dalam Mengembangkan *Frontend Website* Haades, yaitu sebuah aplikasi berbasis *website* yang digunakan untuk mendukung proses pengelolaan data yang digunakan untuk mencatat penerbangan *overflying* yang melintas di wilayah udara FIR Jakarta dan Ujung pandang [4]. *Website* Haades dikembangkan untuk mempermudah akses informasi, mengelola data layanan, serta membantu tim dalam melakukan monitoring pekerjaan secara lebih terstruktur. Adapun rincian tugas yang dikerjakan adalah sebagai berikut:

- Analisis Kebutuhan Sistem Haades

Pada tahap analisis kebutuhan, dilakukan pemahaman alur kerja dan fitur yang akan dikembangkan pada sisi *frontend Website* Haades. Analisis ini dilakukan melalui diskusi dengan Tim Teknologi Informasi untuk memetakan kebutuhan pengguna serta menentukan fitur-fitur yang harus tersedia pada tampilan antarmuka. Fitur tersebut meliputi pembuatan halaman *Dashboard*, fitur *Create FPS 2*, fitur *Single FPS*, serta berbagai fitur tampilan dan pengelolaan data lainnya. Proses analisis ini bertujuan agar seluruh komponen *frontend* yang dibangun sesuai dengan kebutuhan operasional tim dan mampu menampilkan data secara jelas, interaktif, dan mudah digunakan.

- Perancangan Tampilan dan Struktur Website

Tahap perancangan diawali dengan pembuatan *mockup Website* Haades sebagai rancangan awal tampilan antarmuka. *Mockup* ini digunakan untuk memvisualisasikan susunan halaman, posisi elemen, serta alur navigasi yang akan digunakan pada *website*. Setelah *mockup* disetujui, proses dilanjutkan dengan merancang antarmuka dan struktur halaman secara lebih detail dengan memperhatikan aspek kemudahan penggunaan (*user experience*) dan konsistensi visual. Kegiatan perancangan mencakup pembuatan *layout* halaman, pengaturan navigasi, penentuan skema warna, serta penyusunan komponen visual agar *website* mudah dipahami oleh pengguna dalam mengelola dan menampilkan data pada sistem.

- Pengembangan Website Haades

Pada tahap ini dilakukan pengembangan antarmuka (*frontend*) *website* Haades, yang meliputi penyusunan tampilan utama, pengaturan struktur UI, serta penyesuaian elemen visual agar selaras dengan kebutuhan sistem. Selain membangun *layout* dan komponen dasar, proses ini juga mencakup penambahan berbagai fitur pada sisi *frontend*, seperti implementasi fitur *create FPS 2*, *single FPS*, dan fitur pendukung lainnya. Seluruh pengembangan difokuskan untuk memastikan tampilan antarmuka berjalan responsif, mudah digunakan, dan siap terhubung dengan *backend* dalam keseluruhan alur sistem.

- Implementasi Fitur dan Fungsionalitas

Menerapkan fitur berdasarkan kebutuhan yang telah dianalisis, seperti:

- penyimpanan dan pengolahan data,

- *form input* untuk kebutuhan operasional sistem,
- tampilan tabel yang mudah dibaca,
- fitur pencarian atau *filter* data,
- penyesuaian alur kerja tampilan sesuai kebutuhan tim.

Setiap fitur diuji agar dapat mendukung operasional kerja dengan baik.

- **Pengujian dan Validasi Sistem**

Melakukan uji coba sistem secara menyeluruh untuk memastikan bahwa semua fitur berjalan dengan baik dan sesuai kebutuhan pengguna, serta mengidentifikasi dan memperbaiki potensi kesalahan sebelum sistem digunakan lebih lanjut.

3.3 Uraian Pelaksanaan Magang

Sebagai *Frontend Developer* pada Tim Teknologi Informasi, saya bertugas mengembangkan antarmuka *Website* Haades yang digunakan untuk mendukung pengelolaan data operasional instansi. Selama proses pengembangan, saya dibimbing oleh Bapak Rahmat selaku koordinator tim dan *supervisor* magang. Pengembangan *frontend* dilakukan menggunakan Nuxt.js dengan fokus pada penyusunan struktur halaman, pembuatan komponen UI, pengaturan *layout*, serta penambahan fitur seperti Single FPS, Create FPS 2, dan fitur tampilan data lainnya. Seluruh tampilan dirancang responsif dan disesuaikan dengan kebutuhan tim agar sistem mudah digunakan dan dapat terintegrasi dengan *backend* secara optimal.

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

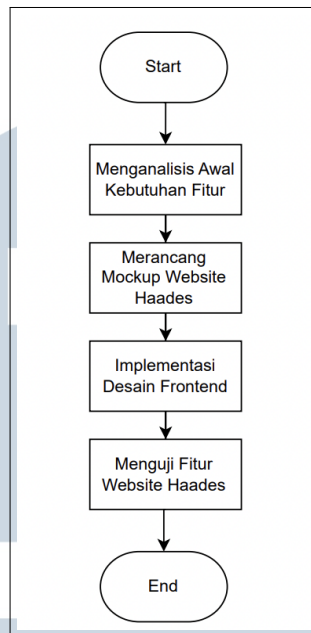
U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang.

Minggu Ke-	Pekerjaan yang Dilakukan
1	Pengenalan sistem Haades, struktur alur kerja tim, serta analisis awal kebutuhan fitur pada sisi <i>frontend</i> .
2–4	Membuat <i>mockup</i> dan desain awal antarmuka (UI) <i>Website</i> Haades sebagai acuan pengembangan.
5–6	Mempelajari dasar-dasar penggunaan Nuxt.js sebagai <i>framework</i> utama dalam pengembangan <i>frontend</i> .
7–9	Merancang struktur navigasi, <i>layout</i> halaman, dan komponen visual <i>frontend</i> berdasarkan <i>mockup</i> yang telah dibuat.
10–12	Mulai mengembangkan <i>frontend</i> menggunakan Nuxt.js, menyusun struktur proyek, serta membuat halaman dan komponen dasar.
13–15	Mengimplementasikan fitur dan komponen UI pada <i>frontend</i> sesuai kebutuhan operasional.
16	Mengintegrasikan tampilan dengan API <i>backend</i> untuk menampilkan data pada tabel, detail halaman, dan elemen UI lainnya.
17	Mengembangkan fitur tambahan seperti pencarian, <i>filter</i> data, serta perbaikan tampilan tabel agar lebih mudah dibaca.
18	Melakukan penyesuaian UI/UX, optimalisasi <i>layout</i> , dan penyempurnaan elemen visual agar lebih responsif dan konsisten.
19	Melakukan pengujian tampilan dan interaksi pengguna untuk memastikan semua fitur <i>frontend</i> berjalan dengan baik.
20	Melakukan <i>debugging</i> , memperbaiki <i>error</i> tampilan, serta melakukan perbaikan alur UI berdasarkan masukan dari tim.
21	Presentasi hasil pengembangan, perbaikan sesuai <i>feedback</i> , dan finalisasi <i>frontend Website</i> Haades.

3.3.1 Flowchart Alur Kerja Pengembangan Website HAADES

Pengembangan *website* HAADES dilakukan melalui serangkaian tahapan yang tersusun secara sistematis untuk memastikan sistem yang dibangun mampu memenuhi kebutuhan pengguna serta berfungsi secara optimal. Flowchart alur pengembangan *website* HAADES dapat dilihat pada Gambar 3.1.



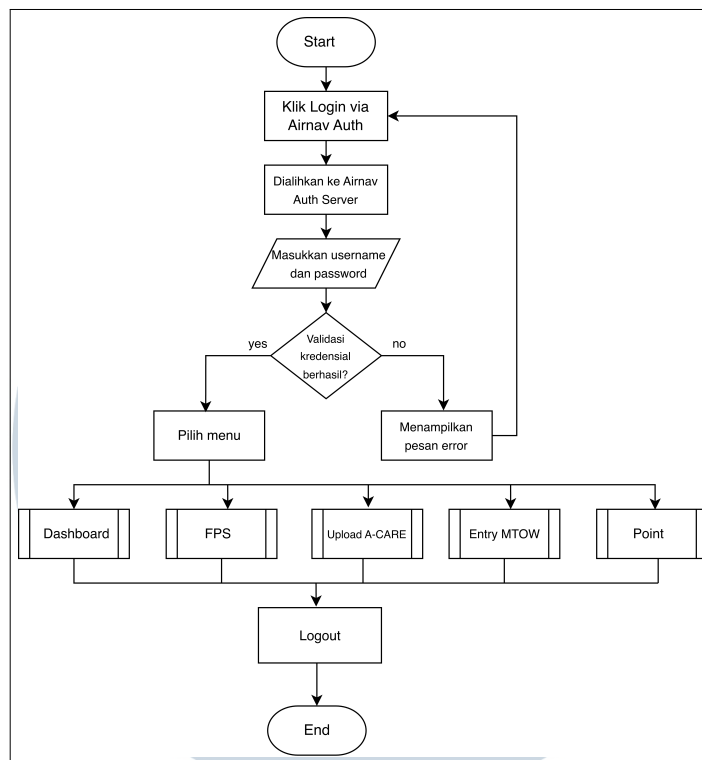
Gambar 3.1. Flowchart Alur Kerja Pengembangan Website HAADES.

Gambar 3.1 menunjukkan *flowchart* alur kerja pengembangan *website* HAADES. Proses dimulai dari analisis kebutuhan fitur berdasarkan masukan *stakeholder* dan evaluasi sistem sebelumnya. Selanjutnya dilakukan perancangan *mockup* sebagai gambaran awal antarmuka dan alur navigasi *website*.

Tahap berikutnya adalah implementasi desain menggunakan Nuxt.js, TypeScript, dan Tailwind CSS, yang diikuti dengan pengujian fitur untuk memastikan sistem berjalan sesuai kebutuhan dan bebas dari *error*. Proses pengembangan dilakukan secara iteratif hingga sistem siap digunakan.

3.3.2 Flowchart Website HAADES

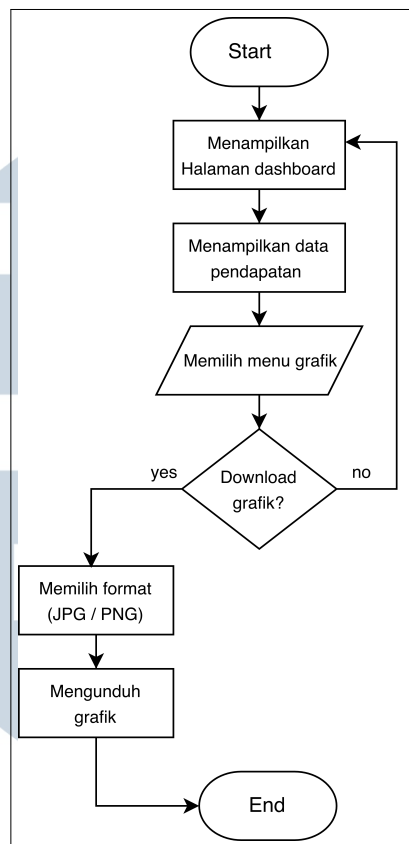
Flowchart digunakan untuk menggambarkan alur proses dari fitur-fitur utama yang terdapat pada *website* HAADES. Setiap *flowchart* divisualisasikan secara sistematis untuk menjelaskan urutan aktivitas, proses, serta keputusan yang terjadi di dalam sistem. Penyajian *flowchart* ini bertujuan untuk memberikan pemahaman yang jelas mengenai logika kerja sistem dan interaksi pengguna pada setiap fitur utama yang tersedia.



Gambar 3.2. Flowchart keseluruhan Website HAADES.

Gambar 3.2 menunjukkan *flowchart* proses autentikasi pengguna pada *website* HAADES yang menerapkan sistem *Single Sign-On* (SSO) melalui AirNav Auth sebagai layanan autentikasi terpusat milik AirNav Indonesia. Proses dimulai ketika pengguna menekan tombol “*Login via AirNav Auth*” pada *website* HAADES, kemudian sistem mengalihkan pengguna ke *server* AirNav Auth untuk memasukkan kredensial berupa *username* dan *password* serta melakukan proses validasi.

Apabila validasi kredensial tidak berhasil, sistem akan menampilkan pesan *error* dan mengarahkan pengguna kembali ke halaman *login*. Sebaliknya, jika autentikasi berhasil, pengguna diberikan akses ke dalam sistem HAADES untuk memilih menu yang tersedia, seperti *Dashboard*, *FPS*, *Upload A-CARE*, *Entry MTOW*, dan *Point*. Setelah selesai menggunakan sistem, pengguna dapat melakukan proses *logout* untuk mengakhiri sesi penggunaan.

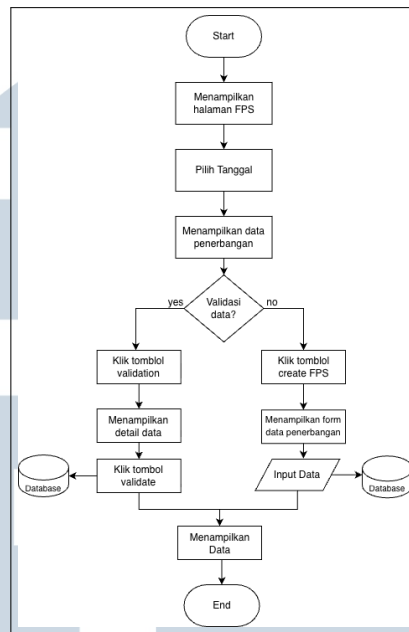


Gambar 3.3. Flowchart Halaman Dashboard Website HAADES.

Gambar 3.3 menggambarkan alur proses pada halaman Dashboard *website* HAADES versi sebelumnya yang tidak mengalami perubahan dalam pengembangan versi terbaru. Proses dimulai ketika *user* mengakses halaman dashboard setelah berhasil login. Sistem menampilkan halaman dashboard yang berisi ringkasan dan statistik data penerbangan. Selanjutnya, sistem menampilkan data pendapatan yang menunjukkan informasi terkait *revenue* atau penghasilan dari operasional penerbangan. Setelah data pendapatan ditampilkan, *user* dihadapkan pada pilihan untuk memilih menu grafik, dan sistem menampilkan opsi visualisasi data dalam bentuk grafik.

Pada tahap selanjutnya, *user* memiliki keputusan apakah akan mengunduh grafik tersebut atau tidak (*Download* grafik?). Jika *user* memilih "Yes", sistem menampilkan pilihan format file (JPG atau PNG), kemudian mengunduh grafik sesuai format yang dipilih. Jika *user* memilih "No", sistem akan kembali ke halaman dashboard (*loop back*), memungkinkan *user* untuk melihat informasi lain atau melakukan aktivitas lainnya. Proses berakhir ketika *user* selesai menggunakan

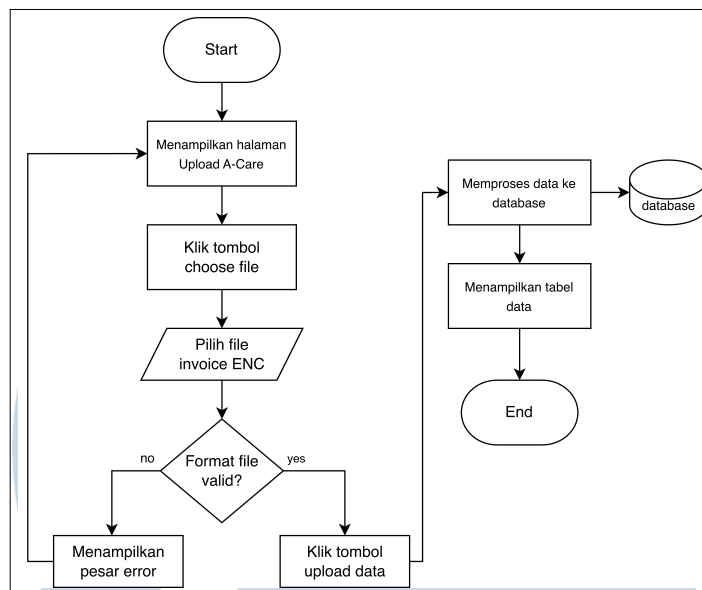
fitur dashboard dan memutuskan untuk keluar atau berpindah ke menu lain.



Gambar 3.4. Flowchart Halaman FPS Website HAADES.

Gambar 3.4 menggambarkan alur proses validasi dan input data pada halaman FPS (Flight Progress Strip). Proses dimulai ketika *user* mengakses halaman FPS dan memilih tanggal untuk menampilkan data penerbangan. Sistem kemudian menampilkan tabel data penerbangan berdasarkan tanggal yang dipilih. Setelah data ditampilkan, *user* dihadapkan pada keputusan apakah data yang ditampilkan sudah valid atau perlu menambahkan data penerbangan baru. Jika data sudah valid (Yes), *user* mengklik tombol *Validation* dan sistem menampilkan detail data penerbangan untuk diverifikasi. Setelah *user* melakukan verifikasi, *user* mengklik tombol *Validate* untuk menyimpan status validasi data ke *database*. Data yang telah divalidasi akan ditandai dengan indikator perubahan status dalam sistem.

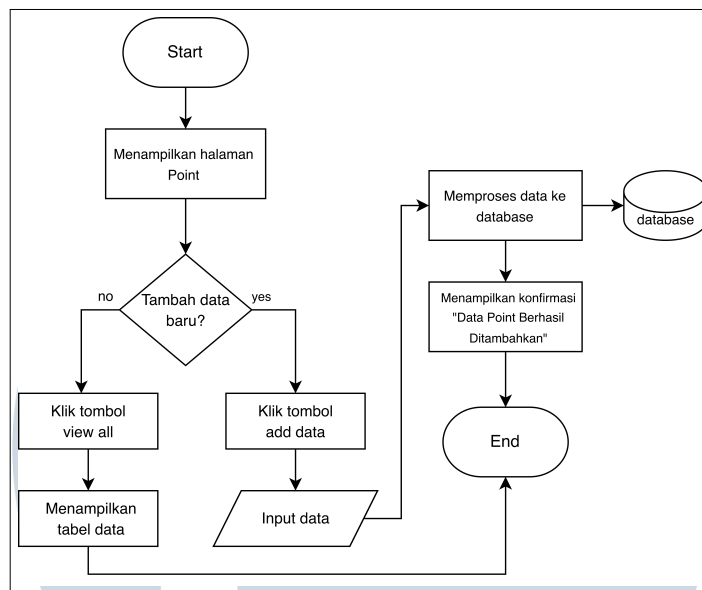
Jika data belum valid atau *user* perlu menambahkan data baru (No), *user* mengklik tombol *Create FPS*. Sistem menampilkan *form* data penerbangan yang perlu diisi oleh *user*. Setelah *user* menginput data penerbangan secara lengkap, sistem menyimpan data tersebut ke *database*. Setelah data berhasil tersimpan ke *database* dari kedua jalur (validasi atau input baru), sistem menampilkan data yang telah diproses pada halaman FPS. Proses berakhir setelah *user* selesai memproses data penerbangan yang diperlukan.



Gambar 3.5. *Flowchart* Upload A-CARE Website HAADES.

Gambar 3.5 menggambarkan alur proses upload data A-CARE pada *website* HAADES versi sebelumnya yang tidak mengalami perubahan dalam pengembangan versi terbaru. Proses dimulai ketika *user* mengakses halaman Upload A-CARE dan mengklik tombol *Choose File* untuk memilih file invoice ENC dalam format Excel. Setelah *user* memilih file, sistem melakukan validasi untuk memastikan format file yang diupload sudah sesuai dengan ketentuan yang berlaku.

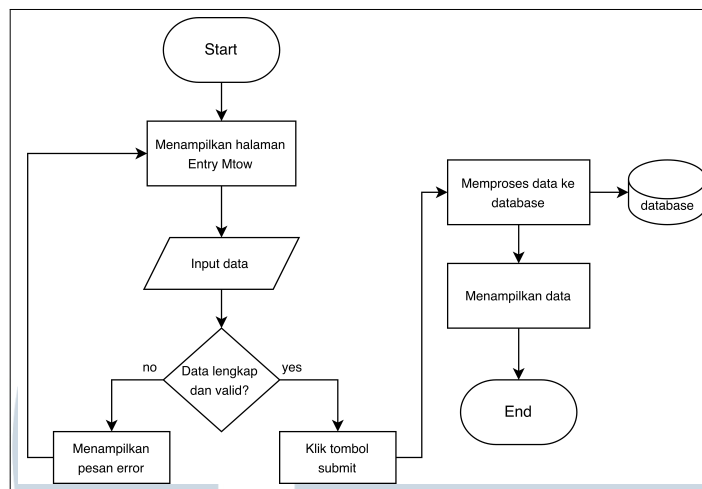
Jika format file tidak valid (No), sistem menampilkan pesan error dan *user* harus kembali ke tahap pemilihan file untuk memilih ulang file dengan format yang benar. Jika format file valid (Yes), *user* mengklik tombol *Upload Data* dan sistem memproses data tersebut untuk disimpan ke *database*. Setelah data berhasil tersimpan, sistem menampilkan tabel data yang telah diupload sebagai konfirmasi bahwa proses upload telah selesai dilakukan.



Gambar 3.6. *Flowchart* Halaman Point Website HAADES.

Gambar 3.6 menggambarkan alur proses pengelolaan data pada halaman Point pada *website* HAADES versi sebelumnya yang tidak mengalami perubahan dalam pengembangan versi terbaru. Proses dimulai ketika *user* mengakses halaman Point, kemudian *user* dihadapkan pada keputusan apakah akan menambahkan data baru atau hanya melihat data yang telah tersedia. Jika *user* memilih untuk tidak menambahkan data baru (No), maka *user* mengklik tombol *View All* dan sistem menampilkan tabel data Point yang berisi seluruh data yang telah tersimpan. Setelah tabel data ditampilkan, proses berakhir.

Jika *user* memilih untuk menambahkan data baru (Yes), maka *user* mengklik tombol *Add Data*. Sistem akan menampilkan *form* input data Point yang harus diisi oleh *user*. Setelah *user* menginput data, sistem memproses dan menyimpan data tersebut ke dalam *database*. Setelah data berhasil disimpan, sistem menampilkan konfirmasi "Data Point Berhasil Ditambahkan" kepada *user* dan proses pengelolaan data pada halaman Point selesai.



Gambar 3.7. *Flowchart* Entry MTOW Website HAADES.

Gambar 3.7 menggambarkan alur proses entry data MTOW (Maximum Take-Off Weight) pada *website* HAADES versi sebelumnya yang tidak mengalami perubahan dalam pengembangan versi terbaru. Proses dimulai ketika *user* mengakses halaman Entry MTOW dan menginput data pesawat yang diperlukan. Setelah *user* menginput data, sistem melakukan validasi untuk memastikan data yang diinput sudah lengkap dan valid.

Jika data tidak lengkap atau tidak valid (No), sistem menampilkan pesan error dan *user* harus kembali ke tahap input data untuk melengkapi atau memperbaiki data yang salah. Jika data lengkap dan valid (Yes), *user* mengklik tombol *Submit* dan sistem memproses serta menyimpan data MTOW ke *database*. Setelah data berhasil tersimpan, sistem menampilkan data yang telah ditambahkan sebagai konfirmasi kepada *user*, dan proses entry MTOW selesai.

3.3.3 Perancangan Mockup Website HAADES

Berdasarkan hasil analisis kebutuhan yang telah dilakukan, proses perancangan *website* HAADES dimulai dengan evaluasi terhadap desain aplikasi yang sudah ada sebelumnya. Evaluasi ini bertujuan untuk mengidentifikasi kekurangan dan area yang perlu diperbaiki dalam pengembangan sistem HAADES yang baru, khususnya dari aspek tampilan antarmuka pengguna dan pengalaman pengguna secara keseluruhan.

A Evaluasi Desain Sebelumnya

Tahap evaluasi desain sebelumnya dilakukan untuk mengidentifikasi permasalahan, kekurangan, serta aspek antarmuka pengguna yang belum optimal pada *website* HAADES. Evaluasi ini bertujuan untuk mengetahui sejauh mana desain yang telah diterapkan mampu mendukung kemudahan penggunaan (*usability*), kejelasan navigasi, serta kenyamanan pengguna dalam berinteraksi dengan sistem.

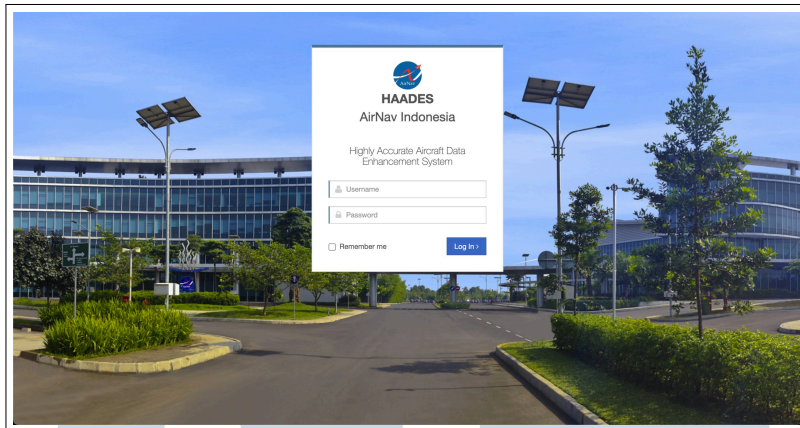
A.1 Evaluasi Desain Halaman Login Sebelumnya

Halaman *login* pada desain sebelumnya yang ditampilkan pada Gambar 3.8 langsung menampilkan *form input username* dan *password* beserta *checkbox* "Remember me" dan tombol "Log In" pada halaman awal tanpa adanya pengenalan atau sambutan kepada pengguna. Meskipun pendekatan ini efisien dari segi teknis, namun kurang memberikan pengalaman yang ramah pengguna terutama bagi pengguna baru yang pertama kali mengakses sistem HAADES.

Pada Gambar 3.8 terlihat bahwa *background* halaman menggunakan foto gedung AirNav Indonesia yang ditampilkan secara langsung di belakang *form login*, namun tidak terdapat informasi tambahan mengenai sistem atau petunjuk navigasi yang jelas. *Form login* menggunakan *card* putih dengan *layout* yang cukup sederhana, menampilkan logo HAADES di bagian atas, keterangan "Highly Accurate Aircraft Data Enhancement System", dan langsung diikuti dengan *field* input tanpa adanya tahapan atau *flow* yang membimbing pengguna.

Pendekatan *direct-to-form* ini dapat membuat pengguna yang baru pertama kali mengakses sistem merasa kurang mendapat orientasi atau pemahaman tentang apa yang akan mereka akses.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



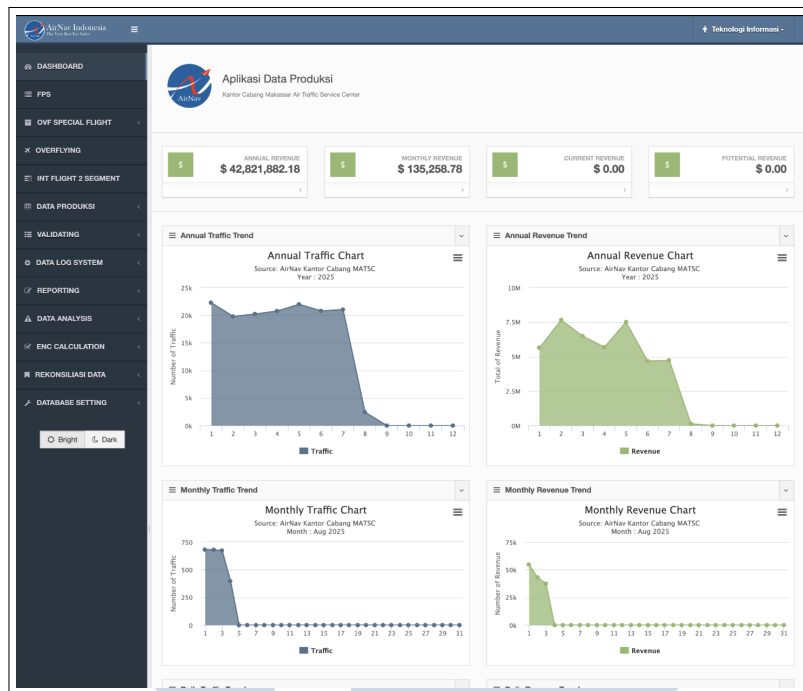
Gambar 3.8. Desain halaman login sebelumnya.

A.2 Evaluasi Desain Halaman Dashboard Sebelumnya

Desain *dashboard* sebelumnya yang terlihat pada Gambar 3.9 menampilkan beberapa kelemahan dalam hal antarmuka pengguna. Penggunaan *sidebar* dengan *text label* yang panjang membuat navigasi terkesan penuh dan kurang efisien. Visualisasi data menggunakan warna yang monoton tanpa sistem *color coding* yang jelas, sehingga pengguna kesulitan membedakan kategori data dengan cepat.

Pada Gambar 3.9 dapat diamati bahwa *layout* yang kurang terstruktur dan minimnya *white space* membuat tampilan terkesan padat dan sulit dibaca. Grafik yang ditampilkan juga kurang modern dengan informasi yang tidak terorganisir dengan baik. Hal ini menyebabkan pengguna memerlukan waktu lebih lama untuk memahami informasi yang disajikan pada halaman *dashboard*.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 3.9. Desain halaman *dashboard* sebelumnya.

A.3 Evaluasi Desain Halaman Menu FPS Sebelumnya

Halaman menu FPS pada desain sebelumnya yang ditunjukkan pada Gambar 3.10 memiliki beberapa kelemahan signifikan dalam aspek desain antarmuka. Bagian *header* menggunakan kombinasi warna biru gelap dengan gradien yang kurang memberikan kontras visual yang baik, sementara logo AirNav dan judul HAADES ditempatkan kurang proporsional sehingga mengurangi area efektif untuk informasi utama.

Bagian "*Query Data*" pada Gambar 3.10 memiliki *spacing* yang tidak konsisten, dengan empat tombol aksi (*View*, *Create FPS*, *Create FPS 2*, dan *Single FPS*) yang tidak memiliki hierarki visual yang jelas untuk membedakan tingkat kepentingan fungsi masing-masing tombol. Penempatan tombol "*Single FPS*" yang terpisah dari kelompok tombol lainnya menimbulkan kebingungan dalam pengelompokan fungsi yang saling terkait.

Tabel data yang terlihat pada Gambar 3.10 menggunakan *header* berwarna biru terang yang terlalu kontras dan menyilaukan mata, membuat teks sulit dibaca terutama pada kolom sempit. Kolom "*ACTION*" menampilkan tombol "*VALIDATION*" dengan warna oranye yang terlalu dominan dan mengganggu fokus

pada data utama. Bagian ”*DATA SOURCE*” menggunakan empat *badge* berwarna berbeda yang ditampilkan horizontal, membuat kolom sangat lebar dan tidak efisien dalam penggunaan ruang layar.

Sistem paginasi menggunakan desain standar tanpa *visual feedback* yang cukup untuk menunjukkan halaman aktif. Secara keseluruhan, desain halaman FPS sebelumnya memiliki masalah dalam konsistensi warna, hierarki visual, dan penggunaan ruang yang tidak efisien, menyebabkan pengguna memerlukan waktu lebih lama untuk memahami dan mengoperasikan halaman tersebut.

ACTION	FLIGHT NUMBER	ADIR	ADIR	DOF	AC REG	AC TYPE	POINT IN	TIME IN	POINT OUT	TIME OUT	DATA SOURCE
VALIDATION	AART51	RKSI	WSSS	2025-08-13	HL3383	A359	UKUS	2025-08-13 12:22	NIXEB	2025-08-13 12:47	FPS AFTN AIDC ATIS
VALIDATION	AIC347	WSSS	VOMM	2025-08-13	VTPPM	A321	PUGER	2025-08-13 02:40	ANSAX	2025-08-13 03:17	FPS AFTN AIDC ATIS
VALIDATION	AIC348	WSSS	VOMM	2025-08-13	VTPPI	A321	PUGER	2025-08-13 05:55	ANSAX	2025-08-13 06:32	FPS AFTN AIDC ATIS
VALIDATION	ALK302	VCBI	WSSS	2025-08-13	4RABO	A320	NIXUL	2025-08-13 03:53	SALAX	2025-08-13 05:09	FPS AFTN AIDC ATIS
VALIDATION	ALK314	VCBI	WAKK	2025-08-13	4RANF	A321N	NIXUL	2025-08-13 04:05	PUGER	2025-08-13 05:10	FPS AFTN AIDC ATIS

Gambar 3.10. Desain halaman menu FPS sebelumnya.

A.4 Evaluasi Desain Halaman Entry MTOW Sebelumnya

Halaman Entry MTOW pada desain sebelumnya yang ditampilkan pada Gambar 3.11 memiliki beberapa kelemahan dalam aspek desain antarmuka pengguna. Bagian *header* masih menggunakan desain yang sama dengan halaman FPS, dengan kombinasi warna biru gelap dan logo yang kurang proporsional.

Pada Gambar 3.11 terlihat bahwa *form input* MTOW menggunakan tata letak dua kolom dengan formulir input di sisi kiri dan bagian *upload file* di sisi kanan, namun proporsi pembagian ruang tidak seimbang. *Field* input seperti ”AC REG”, ”Airlines”, ”Type Aircraft”, ”MTOW”, dan ”Faktor Berat” memiliki *spacing* vertikal yang tidak konsisten dan terlalu renggang, membuat formulir terkesan panjang dan membuang banyak ruang kosong.

Bagian ”*MTOW File Upload*” menggunakan desain standar tanpa

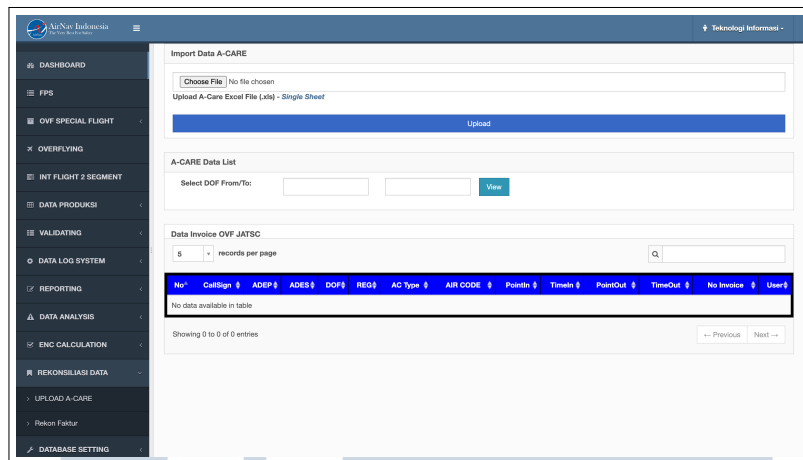
kustomisasi visual, dengan tombol *"EMPTY"* dan *"UPLOAD"* yang ditempatkan horizontal. Tombol *"Submit"* ditempatkan di bagian bawah dengan posisi rata kanan. Secara keseluruhan, desain halaman Entry MTOW sebelumnya memiliki masalah dalam efisiensi penggunaan ruang, konsistensi *spacing*, dan kurangnya *visual feedback* yang membantu pengguna.

Gambar 3.11. Desain halaman Entry MTOW sebelumnya.

A.5 Evaluasi Desain Halaman Upload A-CARE Sebelumnya

Halaman Upload A-CARE pada desain sebelumnya yang ditampilkan pada Gambar 3.12 menunjukkan beberapa kelemahan dalam aspek desain antarmuka. Bagian *"Import Data A-CARE"* menggunakan tombol *"Choose File"* dengan desain standar *browser* tanpa kustomisasi visual yang modern. Teks keterangan *"Upload A-Care Excel File (.xls) - Single Sheet"* ditampilkan dalam ukuran kecil dan kurang menonjol, sehingga pengguna dapat melewati informasi penting mengenai format *file*. Tombol *"Upload"* memanjang penuh secara horizontal dengan desain yang kurang proporsional dan terkesan monoton.

Bagian *"A-CARE Data List"* pada Gambar 3.12 memiliki dua *input field* untuk filter *"Select DOF From/To"* tanpa *label* yang jelas dan *placeholder* yang informatif. Tabel *"Data Invoice OVF JATSC"* menampilkan pesan *"No data available in table"* tanpa *visual feedback* atau ilustrasi yang membantu pengguna memahami bahwa mereka perlu mengunggah data terlebih dahulu. Secara keseluruhan, desain halaman ini terkesan kurang modern dengan kurangnya elemen visual yang menarik dan informasi panduan yang jelas.



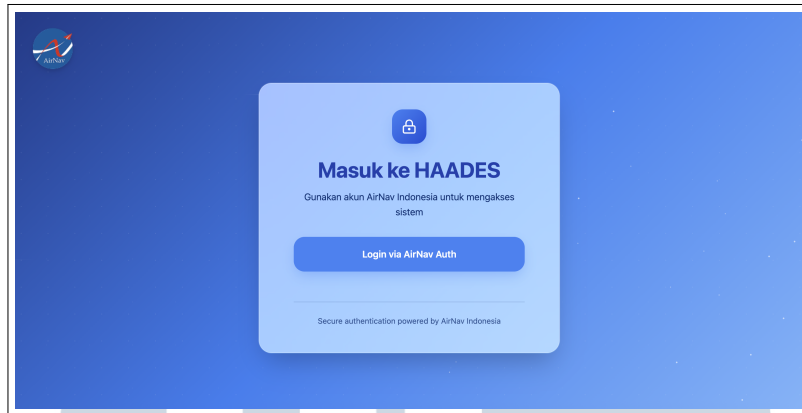
Gambar 3.12. Desain halaman Upload A-CARE sebelumnya.

B Perancangan Ulang Desain

Berdasarkan hasil evaluasi terhadap desain sebelumnya, dilakukan perancangan ulang antarmuka pengguna pada *website* HAADES untuk meningkatkan kualitas pengalaman pengguna. Perancangan ulang ini bertujuan untuk memperbaiki kelemahan pada desain lama dengan meningkatkan kejelasan alur penggunaan (*user flow*) serta tampilan visual yang lebih modern dan profesional.

B.1 Perancangan Ulang Halaman Login

Berdasarkan evaluasi terhadap desain *login* sebelumnya, dilakukan perancangan ulang untuk meningkatkan aspek visual dan pengalaman pengguna. Desain baru yang ditampilkan pada Gambar 3.13 menggunakan *card* berwarna biru muda dengan *rounded corners* dan *shadow effect* yang diletakkan di tengah halaman. Halaman dilengkapi dengan *icon lock* berwarna biru sebagai *visual indicator* keamanan, judul "Masuk ke HAADES", dan keterangan "Gunakan akun AirNav Indonesia untuk mengakses sistem". Tombol "Login via AirNav Auth" berwarna biru tua diposisikan secara *prominent* untuk memudahkan pengguna melakukan autentikasi. *Background* menggunakan *gradient* warna biru dengan efek partikel bintang yang memberikan kesan teknologi dan profesional. Di bagian bawah *card* terdapat teks "Secure authentication powered by AirNav Indonesia" untuk memberikan informasi keamanan kepada pengguna.

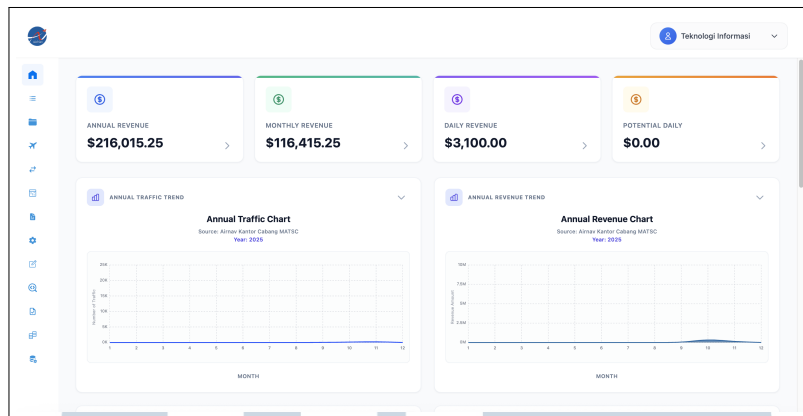


Gambar 3.13. Desain Halaman Login Setelahnya

B.2 Perancangan Ulang Halaman Dashboard

Berdasarkan evaluasi terhadap desain *dashboard* sebelumnya, dilakukan perancangan ulang dengan pendekatan yang lebih modern dan *user-friendly*. Hasil perancangan *dashboard* baru dapat dilihat pada Gambar 3.14.

Setiap *card revenue* pada Gambar 3.14 dibedakan dengan warna yang berbeda yaitu biru untuk *Annual Revenue*, hijau untuk *Monthly Revenue*, ungu untuk *Daily Revenue*, dan orange untuk *Potential Daily*. Navigasi disederhanakan melalui penggunaan *icon-based sidebar* yang lebih *clean* dan minimalis. Visualisasi grafik menggunakan *line chart* yang *smooth* untuk menampilkan *Annual Traffic Trend* dan *Annual Revenue Trend* secara berdampingan, memudahkan perbandingan data. Desain baru juga menerapkan penggunaan *white space* yang optimal, *rounded corners*, dan *typography* yang modern, menciptakan pengalaman visual yang lebih profesional.



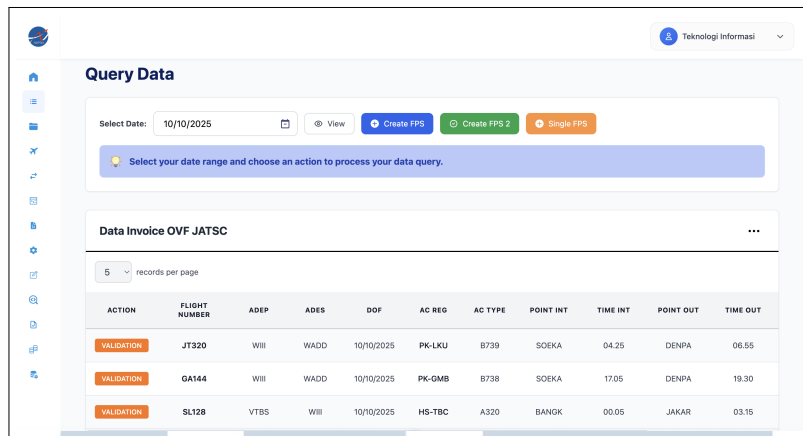
Gambar 3.14. Desain halaman *dashboard* setelahnya.

B.3 Perancangan Ulang Halaman Menu FPS

Berdasarkan evaluasi terhadap desain menu FPS sebelumnya, dilakukan perancangan ulang dengan pendekatan yang lebih modern dan *user-friendly*. Hasil perancangan menu FPS baru dapat dilihat pada Gambar 3.15. Desain baru menggunakan *card* dengan *white background* dan *rounded corners* yang lebih *clean*.

Bagian "*Query Data*" pada Gambar 3.15 ditampilkan dengan *spacing* yang konsisten, dilengkapi *date picker* dengan ikon kalender dan empat tombol aksi yang dikelompokkan dengan jarak proporsional. Penambahan *info box* berwarna biru muda memberikan panduan kepada pengguna. Tabel "*Data Invoice OVF JATSC*" menggunakan *header* dengan *typography* yang lebih *readable*, sementara *sidebar* menggunakan *icon-based navigation* yang minimalis. Desain baru ini menciptakan *visual hierarchy* yang jelas dan penggunaan ruang yang lebih optimal.

UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.15. Desain halaman menu FPS setelahnya.

B.4 Perancangan Ulang Halaman Entry MTOW

Berdasarkan evaluasi terhadap desain Entry MTOW sebelumnya, dilakukan perancangan ulang dengan fokus pada efisiensi penggunaan ruang dan kejelasan informasi. Hasil perancangan Entry MTOW baru dapat dilihat pada Gambar 3.16. Desain baru menggunakan *single column layout* yang lebih *streamlined*.

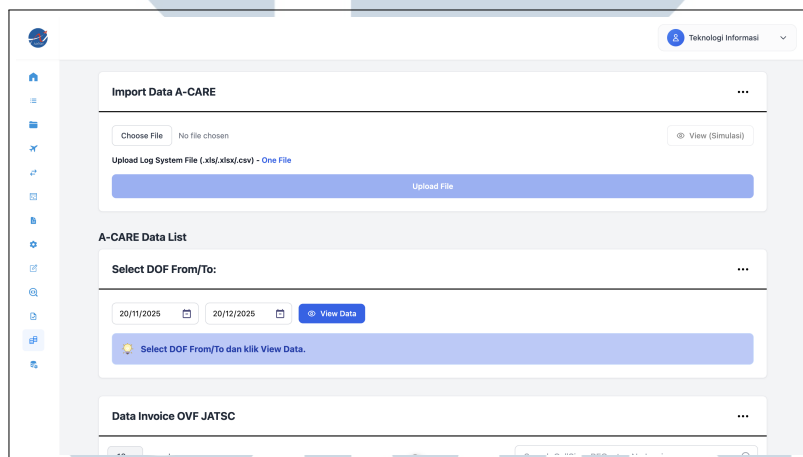
Pada Gambar 3.16 terlihat bahwa setiap *field* input dilengkapi dengan *label* yang jelas dan *placeholder text* yang informatif. *Spacing* vertikal antar *field* dibuat lebih konsisten, menghilangkan ruang kosong yang berlebihan. *Field* wajib ditandai dengan *asterisk* (*) berwarna merah, memberikan *visual cue* yang jelas. Tombol "Submit" ditempatkan di bagian kanan bawah dengan desain modern menggunakan *rounded corners*.

Gambar 3.16. Desain halaman Entry MTOW setelahnya.

B.5 Perancangan Ulang Halaman Upload A-CARE

Berdasarkan evaluasi terhadap desain Upload A-CARE sebelumnya, dilakukan perancangan ulang dengan meningkatkan kejelasan informasi dan *visual feedback*. Hasil perancangan Upload A-CARE baru dapat dilihat pada Gambar 3.17. Desain baru menggunakan *card-based layout* yang lebih terstruktur dengan pemisahan yang jelas antara bagian *upload file*, filter data, dan tampilan tabel.

Bagian "Import Data A-CARE" pada Gambar 3.17 menggunakan area *upload* yang lebih modern dengan tombol "Choose File" yang dikustomisasi. Penambahan fitur "View (Simulasi)" memberikan opsi untuk melihat simulasi data sebelum melakukan *upload*. Bagian "A-CARE Data List" menggunakan *input field* dengan ikon kalender yang jelas untuk filter "Select DOF From/To". Penambahan *info box* berwarna biru muda memberikan panduan yang eksplisit kepada pengguna. Desain baru ini meningkatkan pemahaman pengguna terhadap *workflow upload* data dengan *visual guidance* yang lebih baik.



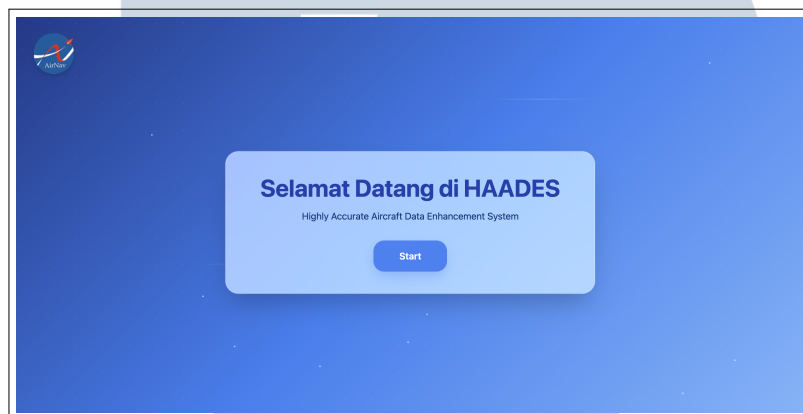
Gambar 3.17. Desain halaman Upload A-CARE setelahnya.

3.3.4 Hasil Implementasi Sistem Website Haades

Bagian ini menjelaskan hasil implementasi sistem *website* HAADES yang telah dikembangkan, meliputi tampilan antarmuka pengguna, fitur-fitur utama, serta implementasi kode program yang digunakan dalam sistem.

A Halaman Login

Halaman *login* merupakan pintu masuk utama bagi pengguna untuk mengakses sistem HAADES. Implementasi halaman *login* menggunakan mekanisme *Single Sign-On* (SSO) melalui AirNav Auth untuk autentikasi terpusat menggunakan kredensial AirNav Indonesia. Proses *login* dimulai dari halaman *welcome* yang dapat dilihat pada Gambar 3.18.

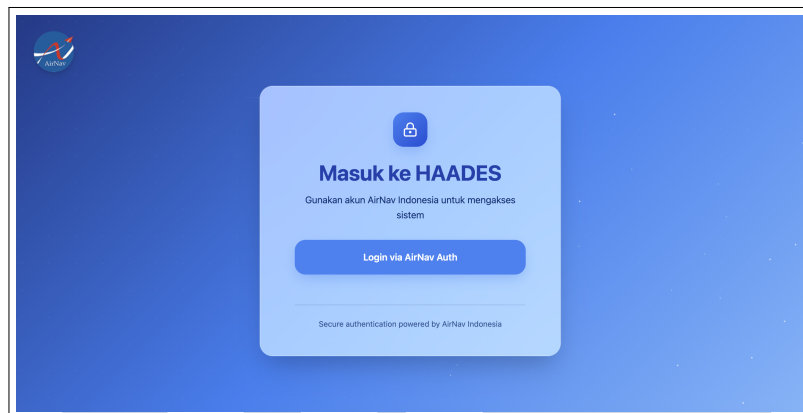


Gambar 3.18. Tampilan Halaman Welcome HAADES

Berdasarkan Gambar 3.18, halaman *welcome* menampilkan ucapan "Selamat Datang di HAADES" dengan deskripsi "*Highly Accurate Aircraft Data Enhancement System*". Halaman menggunakan *card* berwarna biru muda dengan *rounded corners* di tengah layar dan tombol "*Start*" berwarna biru tua sebagai *call-to-action*. *Background* menggunakan *gradient* biru dengan efek partikel bintang untuk memberikan kesan teknologi dan profesional.

Ketika pengguna mengklik tombol "*Start*", sistem mengarahkan ke halaman "*Masuk ke HAADES*" seperti yang ditampilkan pada Gambar 3.19.

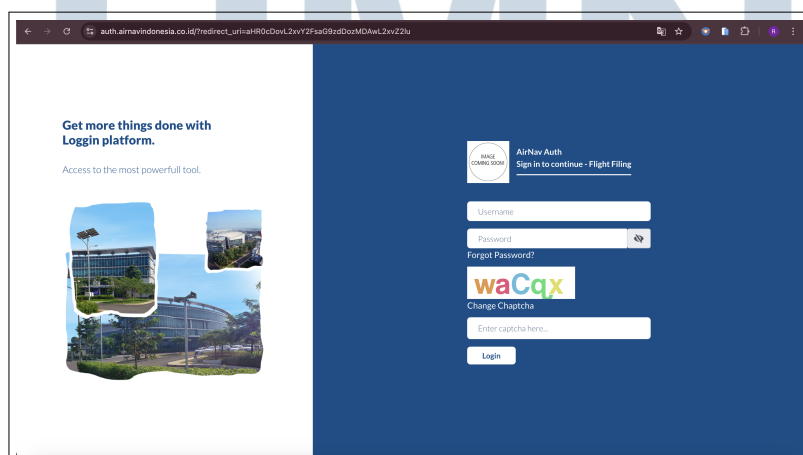
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.19. Tampilan Halaman Masuk ke HAADES

Berdasarkan Gambar 3.19, halaman ini menampilkan *icon lock* sebagai *visual indicator* keamanan, judul "Masuk ke HAADES", dan keterangan "Gunakan akun AirNav Indonesia untuk mengakses sistem". Tombol "Login via AirNav Auth" berwarna biru tua diposisikan secara *prominent* untuk memulai proses autentikasi SSO. Di bagian bawah terdapat teks "Secure authentication powered by AirNav Indonesia" yang menginformasikan bahwa autentikasi menggunakan sistem keamanan AirNav Indonesia.

Setelah pengguna mengklik tombol "Login via AirNav Auth", sistem mengalihkan pengguna ke server AirNav Auth untuk melakukan autentikasi seperti yang terlihat pada Gambar 3.20.



Gambar 3.20. Tampilan Halaman Login AirNav Auth

Berdasarkan Gambar 3.20, halaman *login* AirNav Auth memiliki beberapa

komponen utama, yaitu *input field* untuk *username*, *input field* untuk *password* dengan fitur *show/hide password*, fitur "*Forgot Password*", *captcha* sebagai keamanan tambahan, dan tombol "*Login*". Setelah autentikasi berhasil, sistem mengalihkan pengguna kembali ke *website* HAADES dengan membawa *access token* dan *refresh token*.

Implementasi proses autentikasi pada sisi klien menggunakan fungsi `setTokens` yang bertugas memvalidasi dan menyimpan *token* yang diterima dari AirNav Auth. Implementasi fungsi tersebut dapat dilihat pada Kode 3.1.

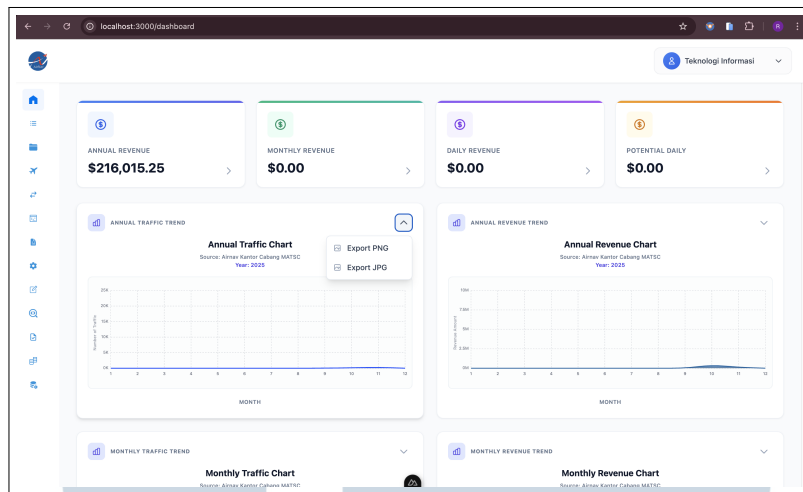
```
1 // Fungsi untuk set token setelah redirect dari AirNav SSO
2 const setTokens = async (access: string, refresh?: string) => {
3   const valid = await verifyToken(access);
4
5   if (valid) {
6     accessToken.value = access;
7     if (refresh) refreshToken.value = refresh;
8     await navigateTo("/dashboard", { replace: true });
9   } else {
10    accessToken.value = null;
11    refreshToken.value = null;
12    await navigateTo("/login?error=invalid_token", { replace: true
13    });
14  }
15 };
```

Kode 3.1: Fungsi `setTokens` untuk Validasi dan Penyimpanan Token

Berdasarkan Kode 3.1, fungsi `setTokens` menerima dua parameter yaitu *access* (*access token*) dan *refresh* (*refresh token* opsional). Fungsi ini melakukan verifikasi *token* menggunakan `verifyToken`, jika valid maka *token* disimpan ke *state management* dan pengguna diarahkan ke halaman *dashboard*, jika tidak valid maka *token* dihapus dan pengguna diarahkan kembali ke halaman *login* dengan pesan *error*. Mekanisme ini memastikan hanya pengguna terautentikasi yang dapat mengakses sistem HAADES.

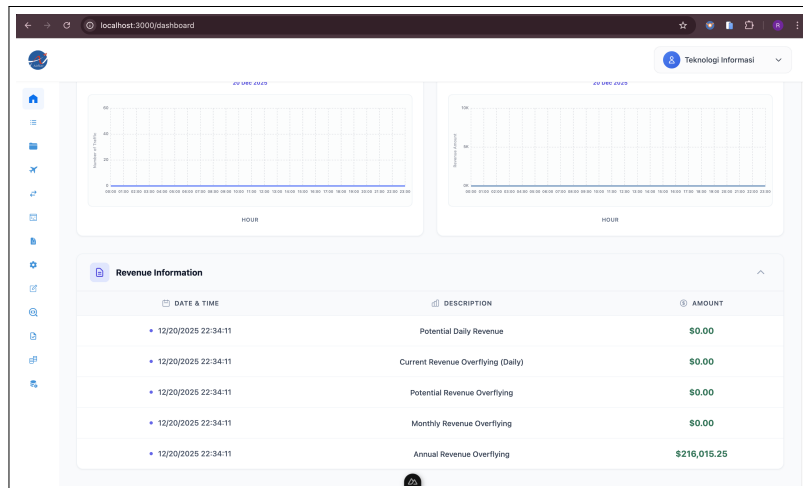
B Halaman Dashboard

Halaman *dashboard* merupakan halaman utama sistem HAADES yang menampilkan informasi *traffic overflying* dan pendapatan secara periodik dalam bentuk kartu informasi dan grafik. Tampilan halaman *dashboard* dapat dilihat pada Gambar 3.21 dan Gambar 3.22.



Gambar 3.21. Tampilan Kartu Informasi dan Grafik Tren Tahunan pada Dashboard

Berdasarkan Gambar 3.21, bagian atas *dashboard* menampilkan empat kartu informasi yaitu *Annual Revenue* (\$216,015.25), *Monthly Revenue* (\$0.00), *Daily Revenue* (\$0.00), dan *Potential Daily* (\$0.00). Di bawahnya terdapat *Annual Traffic Chart* dan *Annual Revenue Chart* untuk tahun 2025 dari sumber Airtax Kantor Cabang MATSC. Setiap grafik dilengkapi *dropdown menu* untuk mengunduh dalam format PNG atau JPG.



Gambar 3.22. Tampilan Grafik Harian dan Tabel Revenue Information pada Dashboard

Berdasarkan Gambar 3.22, bagian bawah *dashboard* menampilkan dua grafik harian untuk tanggal 20 Desember 2025 yang menampilkan *traffic* dan pendapatan per jam. Grafik dilengkapi *dropdown menu* untuk mengunduh

visualisasi. Di bawahnya terdapat tabel *Revenue Information* yang dapat dibuka atau ditutup, menampilkan tiga kolom yaitu *Date & Time*, *Description*, dan *Amount* dengan *timestamp* 12/20/2025 22:34:11. Bagian paling bawah terdapat *Monthly Traffic Chart* dan *Monthly Revenue Chart* dengan fitur *dropdown* untuk mengunduh grafik.

Implementasi pengambilan data *dashboard* dimulai dengan mendefinisikan *interface* TypeScript untuk parameter. *Interface* tersebut dapat dilihat pada Kode 3.2.

```
1 interface DashboardParams {  
2   period?: "yearly" | "monthly" | "daily";  
3   year?: number;  
4   month?: number;  
5 }
```

Kode 3.2: Interface DashboardParams untuk Parameter Dashboard

Berdasarkan Kode 3.2, *interface* *DashboardParams* mendefinisikan parameter *period* ("yearly", "monthly", atau "daily"), *year*, dan *month* yang semuanya bersifat opsional.

Implementasi fungsi utama untuk mengambil data *dashboard* dapat dilihat pada Kode 3.3.

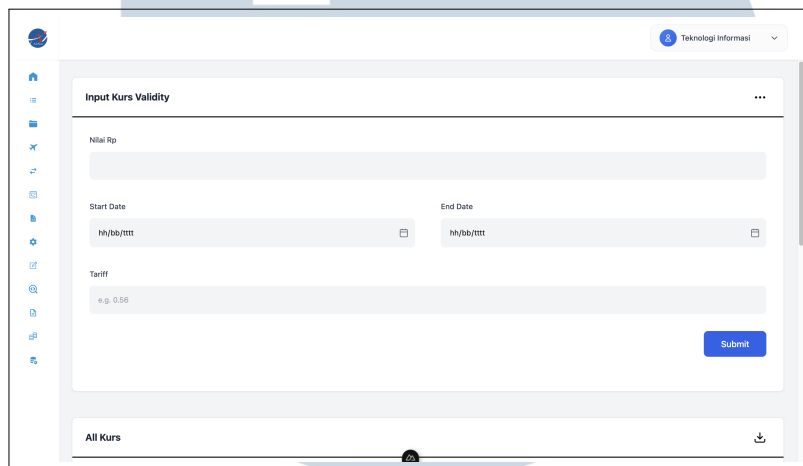
```
1 // Fungsi untuk mengambil data dashboard  
2 const getDashboardData = async (params?: DashboardParams) => {  
3   const config = useRuntimeConfig();  
4   const baseUrl = config.public.apiBase || "http://localhost:4000/  
   api";  
5  
6   const response = await fetch(`${baseUrl}/dashboard`, {  
7     method: "GET",  
8     headers: { "Content-Type": "application/json" },  
9   });  
10  
11   const data = await response.json();  
12   return { cards: data.revenue, charts: data.trends };  
13 };
```

Kode 3.3: Fungsi getDashboardData untuk Mengambil Data Dashboard

Berdasarkan Kode 3.3, fungsi *getDashboardData* melakukan konfigurasi *base URL API*, melakukan HTTP GET ke *endpoint* /dashboard, kemudian mengembalikan objek berisi *cards* untuk data pendapatan dan *charts* untuk data tren grafik.

C Halaman Kurs

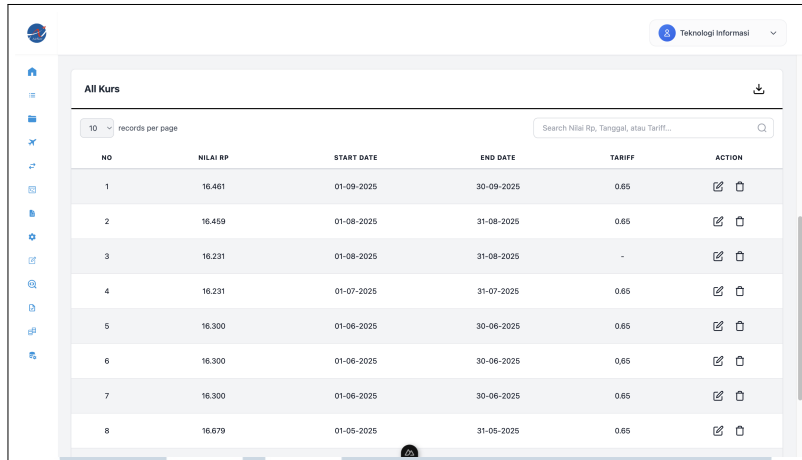
Halaman *kurs* digunakan untuk melakukan sinkronisasi data penerbangan dengan *update* nilai tukar mata uang sesuai periode waktu tertentu. Menu ini dapat diakses melalui *dropdown* profil pengguna di pojok kanan atas dengan memilih opsi "Kurs". Tampilan halaman *kurs* dapat dilihat pada Gambar 3.23, Gambar 3.24, dan Gambar 3.25.



Gambar 3.23. Tampilan Form Input Kurs Validity

Berdasarkan Gambar 3.23, form "Input Kurs Validity" digunakan untuk menambahkan data kurs baru. Form memiliki empat *input field* yaitu *Nilai Rp* untuk memasukkan nilai tukar rupiah, *Start Date* untuk tanggal mulai berlaku, *End Date* untuk tanggal berakhir, dan *Tariff* untuk nilai tarif dengan contoh format 0.56. Di pojok kanan bawah terdapat tombol "Submit" berwarna biru untuk menyimpan data. Form dilengkapi dengan *date picker* untuk memudahkan pemilihan tanggal.

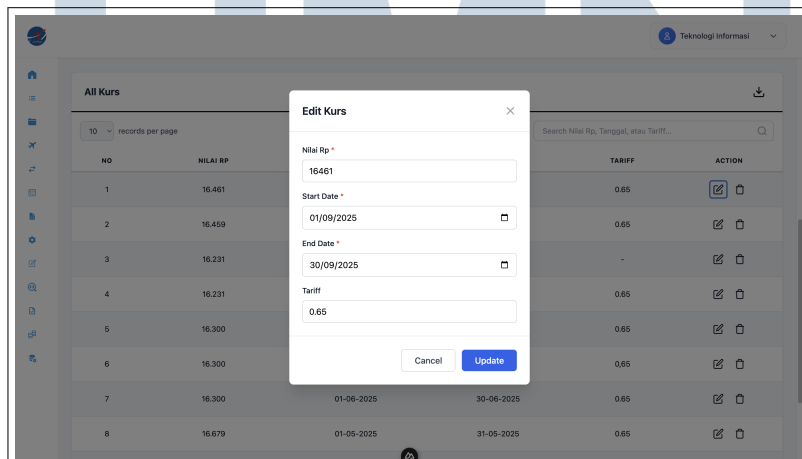
UNIVERSITAS
MULTIMEDIA
NUSANTARA



NO	NILAI RP	START DATE	END DATE	TARIFF	ACTION
1	16.461	01-09-2025	30-09-2025	0.65	[Edit] [Delete]
2	16.459	01-08-2025	31-08-2025	0.65	[Edit] [Delete]
3	16.231	01-08-2025	31-08-2025	-	[Edit] [Delete]
4	16.231	01-07-2025	31-07-2025	0.65	[Edit] [Delete]
5	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
6	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
7	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
8	16.679	01-05-2025	31-05-2025	0.65	[Edit] [Delete]

Gambar 3.24. Tampilan Tabel Data Kurs

Berdasarkan Gambar 3.24, di bawah form input terdapat tabel "All Kurs" yang menampilkan data histori nilai kurs periode sebelumnya. Tabel memiliki enam kolom yaitu *NO* untuk nomor urut, *NILAI RP* untuk nilai tukar rupiah, *START DATE* untuk tanggal mulai berlaku, *END DATE* untuk tanggal berakhir, *TARIFF* untuk nilai tarif, dan *ACTION* yang berisi tombol edit dan hapus. Di bagian atas tabel terdapat pengaturan jumlah data per halaman dan fitur pencarian untuk memfilter data berdasarkan nilai Rp, tanggal, atau tarif. Bagian kanan atas tabel terdapat tombol *download* untuk mengeksport data.



Teknologi Informasi

All Kurs

10 records per page

Search Nilai Rp, Tanggal, atau Tariff...

NO	NILAI RP	START DATE	END DATE	TARIFF	ACTION
1	16.461	01-09-2025	30-09-2025	0.65	[Edit] [Delete]
2	16.459	01-08-2025	31-08-2025	0.65	[Edit] [Delete]
3	16.231	01-08-2025	31-08-2025	-	[Edit] [Delete]
4	16.231	01-07-2025	31-07-2025	0.65	[Edit] [Delete]
5	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
6	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
7	16.300	01-06-2025	30-06-2025	0.65	[Edit] [Delete]
8	16.679	01-05-2025	31-05-2025	0.65	[Edit] [Delete]

Nilai Rp *

16461

Start Date *

01/09/2025

End Date *

30/09/2025

Tariff

0.65

Cancel

Update

Gambar 3.25. Tampilan Modal Edit Kurs

Berdasarkan Gambar 3.25, ketika pengguna mengklik tombol edit pada kolom *ACTION* di tabel, sistem menampilkan modal "Edit Kurs" yang berisi form

dengan data kurs yang sudah terisi sebelumnya. Modal ini memiliki struktur yang sama dengan form input yaitu *Nilai Rp*, *Start Date*, *End Date*, dan *Tariff*. Di bagian bawah modal terdapat dua tombol yaitu *"Cancel"* untuk membatalkan perubahan dan menutup modal, serta tombol *"Update"* berwarna biru untuk menyimpan perubahan data kurs. Modal ini memungkinkan pengguna untuk mengubah nilai kurs yang sudah ada tanpa perlu menghapus dan membuat data baru.

Implementasi form input kurs menggunakan komponen Vue dengan *two-way data binding*. Struktur form dapat dilihat pada Kode 3.4.

```

1 <template>
2   <form @submit.prevent="handleSubmit">
3     <input v-model="form.nilairp" type="number" placeholder="Nilai
4       Rp" required />
5     <input v-model="form.startdate" type="date" required />
6     <input v-model="form.enddate" type="date" required />
7     <input v-model="form.tariff" type="text" placeholder="Tariff"
8       required />
9     <button type="submit">Submit</button>
10  </form>
11 </template>

```

Kode 3.4: Form Input Kurs dengan Vue Template

Berdasarkan Kode 3.4, form menggunakan direktif `@submit.prevent` untuk mencegah *reload* halaman saat submit dan memanggil fungsi `handleSubmit`. Setiap *input field* menggunakan `v-model` untuk mengikat data dengan properti form dan atribut `required` untuk validasi.

Proses validasi dan pengiriman data form dilakukan melalui fungsi `handleSubmit`. Implementasi fungsi tersebut dapat dilihat pada Kode 3.5.

```

1 const handleSubmit = async () => {
2   if (new Date(form.enddate) <= new Date(form.startdate)) {
3     kursStore.error = "End Date harus lebih besar dari Start Date!";
4     return;
5   }
6
7   await kursStore.createKurs({
8     nilairp: Number(form.nilairp),
9     startdate: form.startdate,
10    enddate: form.enddate,
11    tariff: String(form.tariff),
12  });

```

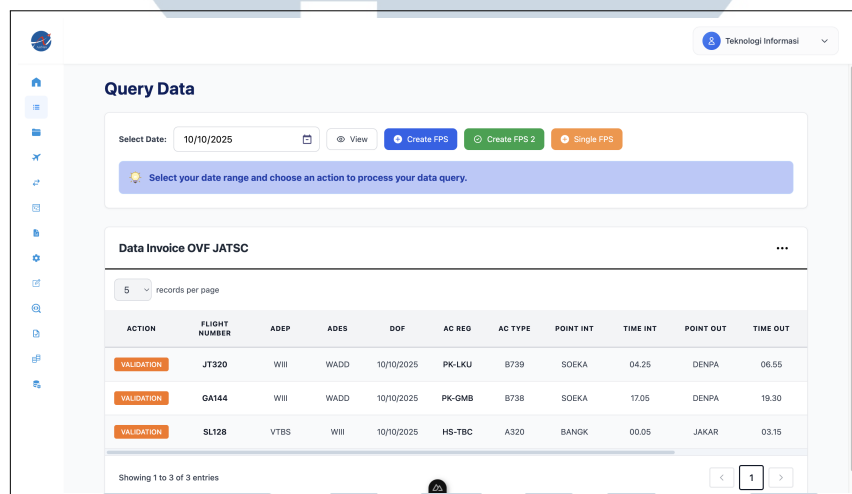
13 };

Kode 3.5: Fungsi handleSubmit untuk Validasi dan Pengiriman Data Kurs

Berdasarkan Kode 3.5, fungsi `handleSubmit` melakukan validasi tanggal dengan memastikan *End Date* lebih besar dari *Start Date*. Jika validasi gagal, pesan error ditampilkan dan proses dihentikan. Jika validasi berhasil, data dikirim ke *store* melalui fungsi `createKurs` dengan mengkonversi nilai `rp` ke tipe `Number` dan `tariff` ke tipe `String`.

D Halaman FPS (Flight Progress Strip)

Halaman FPS (*Flight Progress Strip*) digunakan untuk melakukan validasi dan penginputan data penerbangan lintas (*overflying*) berdasarkan data dukung FPS dan hanya dapat diakses oleh akun cabang JATSC. Menu ini berfungsi sebagai pendukung validasi data penerbangan selain dari sumber AFTN dan AIDC.



Gambar 3.26. Tampilan Halaman Query Data FPS

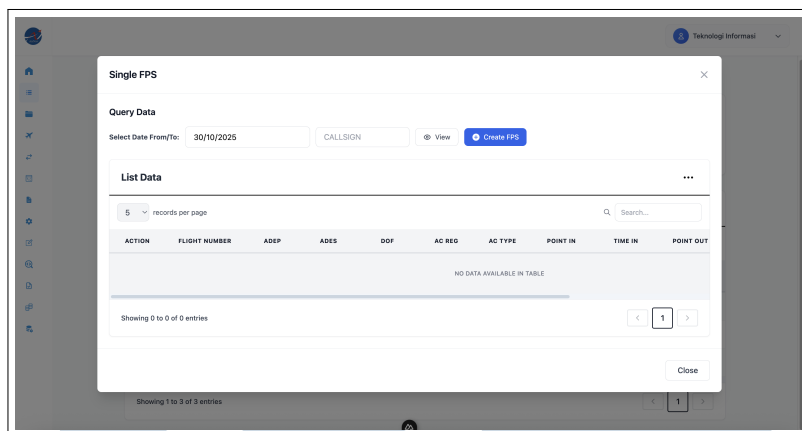
Berdasarkan Gambar 3.26, pengguna dapat memilih tanggal penerbangan melalui fitur *Query Data*. Setelah tanggal dipilih, sistem menampilkan data penerbangan dalam bentuk tabel. Pada bagian ini tersedia tombol *Create FPS*, *Create FPS 2*, dan *Single FPS* yang digunakan untuk melakukan input data penerbangan secara manual.

Gambar 3.27. Tampilan Form Create FPS

Berdasarkan Gambar 3.27, form *Create FPS* digunakan untuk menginput data penerbangan baru seperti *Flight Number*, *ADEP*, *ADES*, *DOF*, rute, serta waktu masuk dan keluar wilayah. Data yang telah diisi dapat disimpan dengan menekan tombol *Create*.

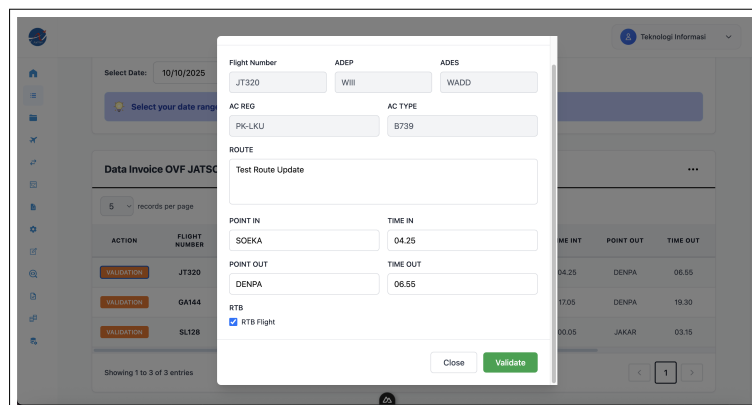
Gambar 3.28. Tampilan Form Create FPS 2

Berdasarkan Gambar 3.28, form *Create FPS 2* memiliki struktur dan fungsi yang sama dengan form *Create FPS*. Form ini digunakan untuk menginput data penerbangan lintas yang tidak tercapture secara otomatis oleh sistem. Field yang tersedia meliputi *Flight Number*, *Date of Flight (DOF)*, *ADEP*, *ADES*, *Flight Type*, *Aircraft Registration*, *Aircraft Type*, *Route*, serta *Point In/Out* dan *Time In/Out*. Setelah data diisi, pengguna dapat menyimpan data dengan menekan tombol *Create*.



Gambar 3.29. Tampilan Single FPS

Berdasarkan Gambar 3.29, *Single FPS* digunakan untuk melakukan *query* dan menampilkan data penerbangan berdasarkan tanggal dan *callsign*. Bagian atas berisi field *Select Date From/To*, *CALLSIGN*, tombol "View" untuk menampilkan data, dan tombol "Create FPS" untuk membuat data baru. Di bawahnya terdapat tabel "List Data" dengan kolom *FLIGHT NUMBER*, *ADEP*, *ADES*, *DOF*, *AC REG*, *AC TYPE*, *POINT IN/OUT*, dan *TIME IN/OUT*. Tabel dilengkapi fitur pencarian, paginasi, dan tombol "Close" di bagian bawah untuk menutup modal.



Gambar 3.30. Tampilan Validasi FPS

Berdasarkan Gambar 3.30, sistem menampilkan detail data penerbangan untuk dilakukan verifikasi. Jika data telah sesuai, pengguna dapat menekan tombol *Validate* sehingga status FPS berubah menjadi valid dan data disimpan ke dalam basis data.

Implementasi logika validasi FPS dilakukan dengan mekanisme *toggle status* untuk mengubah status validasi data.

```
1 const toggleFPSValidation = (index: number) => {  
2   paginatedData.value[index].fpsValidated =  
3   !paginatedData.value[index].fpsValidated;  
4 };
```

Kode 3.6: Logika Toggle Validasi FPS

Proses penyimpanan data FPS hasil input manual dilakukan melalui fungsi `handleCreateFPS`.

```
1 const handleCreateFPS = async () => {  
2   await submitFPSData(fpsFormData.value);  
3   closeModal();  
4 };
```

Kode 3.7: Fungsi Create FPS

E Halaman Entry MTOW

Maximum Take Off Weight (MTOW) adalah berat setinggi-tingginya yang diizinkan bagi pesawat udara untuk bertolak sebagaimana tercantum dalam sertifikat tipe (*Type Certificate*) pesawat udara yang bersangkutan dan tidak tergantung pada pembatasan kekuatan landasan dari bandara yang ditinggalkan maupun yang akan didarati. Halaman *Entry MTOW* digunakan untuk menambahkan data registrasi pesawat beserta informasi MTOW yang belum ada di database lokal HAADES. Tampilan halaman *Entry MTOW* dapat dilihat pada Gambar 3.31 dan Gambar 3.32.

Gambar 3.31. Tampilan Form Input MTOW

Berdasarkan Gambar 3.31, form *"Form Input MTOW"* berisi lima field yaitu *AC REG* untuk registrasi pesawat, *AIRLINES* untuk nama maskapai, *TYPE AIRCRAFT* untuk tipe pesawat, *MTOW* untuk berat maksimum lepas landas sesuai sertifikat tipe pesawat, dan *FAKTOR BERAT* untuk faktor perhitungan berat yang digunakan dalam kalkulasi tarif. Field *AC REG* dan *AIRLINES* ditandai dengan asterisk (*) sebagai field wajib. Tombol *"Submit"* berwarna hijau di pojok kanan bawah digunakan untuk menyimpan data ke database.

Gambar 3.32. Tampilan MTOW File Upload

Berdasarkan Gambar 3.32, di bawah form input terdapat bagian *"MTOW File Upload"* untuk mengunggah data MTOW secara massal melalui file Excel atau CSV. Bagian ini memiliki area *file selector* dengan teks *"No file selected..."* dan tombol *"Browse"* berwarna biru untuk memilih file dari komputer. Setelah file dipilih, pengguna dapat menggunakan tombol *"EMPTY"* berwarna merah untuk membatalkan pilihan file atau tombol *"UPLOAD"* berwarna hijau untuk mengunggah file ke sistem. Fitur *upload* massal ini mempermudah penambahan data MTOW dalam jumlah banyak sekaligus.

Implementasi proses submit data MTOW menggunakan fungsi *handleSubmit* dengan validasi field wajib. Implementasi fungsi tersebut dapat dilihat pada Kode 3.8.

```

1 const handleSubmit = async () => {
2   if (!formData.value.acregister || !formData.value.airlines) {
3     mtowStore.error = "AC Register dan Airlines adalah field wajib
4     !";
5     return;
6   }

```

```

6
7  const response = await mtowStore.createMtow(formData.value);
8  successMessage.value = "Data MTOW berhasil ditambahkan!";
9
10 // Reset form
11 formData.value = {
12   acregister: "",
13   airlines: "",
14   typeaircraft: "",
15   mtow: "",
16   faktorberat: "",
17 };
18 };

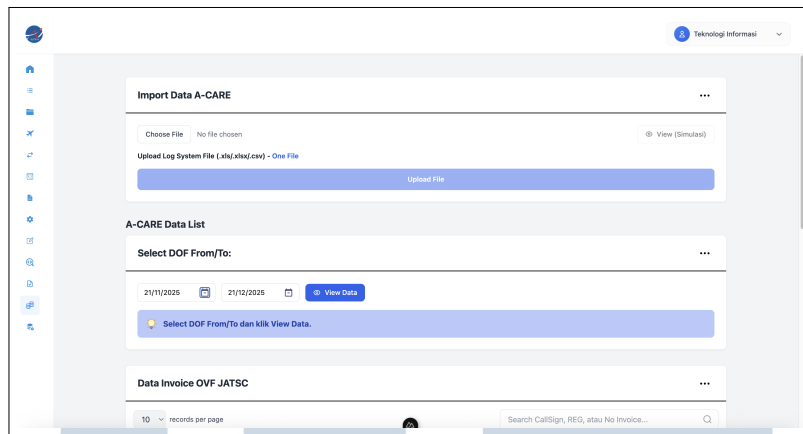
```

Kode 3.8: Fungsi handleSubmit untuk Validasi dan Pengiriman Data MTOW

Berdasarkan Kode 3.8, fungsi handleSubmit melakukan validasi field wajib acregister dan airlines. Jika validasi gagal, pesan error ditampilkan dan proses dihentikan. Jika validasi berhasil, data dikirim ke *store* melalui fungsi createMtow, menampilkan pesan sukses "Data MTOW berhasil ditambahkan!", dan mereset semua field form ke nilai kosong agar siap untuk input data baru. Mekanisme ini memastikan hanya data yang valid dan lengkap yang tersimpan ke database.

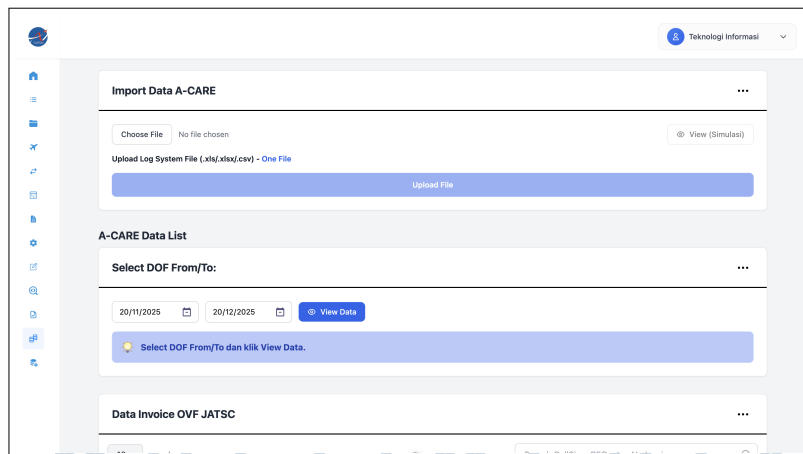
F Halaman Upload A-CARE

Halaman *Upload A-CARE* digunakan untuk mengunggah data invoice ENC dari hasil download aplikasi A-CARE dalam format file Excel (.xls/.xlsx) atau CSV (.csv). Tampilan halaman *Upload A-CARE* dapat dilihat pada Gambar 3.33 dan Gambar 3.34.



Gambar 3.33. Tampilan A-CARE Data List

Berdasarkan Gambar 3.33, bagian "A-CARE Data List" menampilkan tabel data invoice yang telah diunggah sebelumnya. Bagian atas terdapat field *Select DOF From/To* dengan *date picker* untuk memilih rentang tanggal dan tombol "View Data" berwarna biru untuk menampilkan data. Tabel memiliki kolom *ID*, *CALLSIGN*, *ADEP*, *ADES*, *DOF*, *REG*, *AC TYPE*, *AIR CODE*, *POINT IN/OUT*, dan *TIME IN/OUT*. Tabel dilengkapi dengan fitur pencarian untuk memfilter data berdasarkan *CallSign*, *REG*, atau nomor invoice.



Gambar 3.34. Tampilan Import Data A-CARE

Berdasarkan Gambar 3.34, bagian atas halaman terdapat panel "Import Data A-CARE" untuk mengunggah file invoice ENC. Panel ini berisi tombol "Choose File" untuk memilih file dari komputer, informasi format file yang didukung (.xls/.xlsx/.csv), link "One File" sebagai panduan format, dan tombol "Upload File"

berwarna biru untuk mengunggah file ke sistem. Di pojok kanan terdapat tombol "View (Simulasi)" untuk melihat contoh data. File yang diunggah hanya dapat berisi data invoice ENC OVF dari cabang JATSC atau MATSC.

Implementasi proses pemilihan dan upload file menggunakan dua fungsi yaitu `handleFileSelect` untuk validasi file dan `handleUpload` untuk mengunggah file. Implementasi fungsi tersebut dapat dilihat pada Kode 3.9.

```
1 const handleFileSelect = (event: Event) => {
2   const target = event.target as HTMLInputElement;
3   if (target.files && target.files.length > 0) {
4     const file = target.files[0];
5     const allowedExtensions = [".xls", ".xlsx", ".csv"];
6     const fileExtension = file.name.toLowerCase().substring(file.name.lastIndexOf("."));
7
8     if (allowedExtensions.includes(fileExtension)) {
9       selectedFile.value = file;
10      acareStore.clearError();
11    } else {
12      selectedFile.value = null;
13      acareStore.error = "Format file tidak didukung. Gunakan file Excel (.xls/.xlsx) atau CSV (.csv)";
14    }
15  }
16 };
17
18 const handleUpload = async () => {
19   if (!selectedFile.value) {
20     acareStore.error = "Pilih file Excel (.xls/.xlsx) atau CSV terlebih dahulu!";
21     return;
22   }
23
24   await acareStore.uploadFile(selectedFile.value);
25   selectedFile.value = null;
26   if (fileInput.value) fileInput.value.value = "";
27 };
```

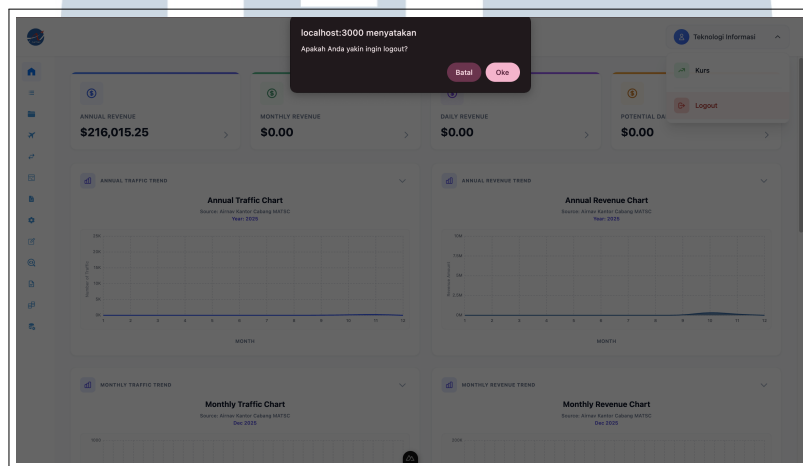
Kode 3.9: Fungsi untuk Validasi dan Upload File A-CARE

Berdasarkan Kode 3.9, fungsi `handleFileSelect` melakukan validasi format file dengan memeriksa ekstensi file apakah termasuk dalam format yang diizinkan (.xls, .xlsx, atau .csv). Jika format valid, file disimpan ke `selectedFile`, jika tidak valid maka pesan error ditampilkan. Fungsi `handleUpload` memeriksa

apakah file sudah dipilih, kemudian mengirim file ke *store* melalui `uploadFile`, dan mereset *file input* setelah proses upload selesai.

G Fitur Logout

Fitur *logout* memungkinkan pengguna untuk keluar dari sistem HAADES dengan aman. Proses *logout* tidak hanya menghapus sesi lokal di aplikasi, tetapi juga melakukan *logout* dari server AirNav Auth untuk memastikan keamanan autentikasi SSO. Tampilan konfirmasi *logout* dapat dilihat pada Gambar 3.35.



Gambar 3.35. Tampilan Konfirmasi Logout

Berdasarkan Gambar 3.35, ketika pengguna mengklik opsi "*Logout*" pada menu *dropdown* profil di pojok kanan atas, sistem menampilkan dialog konfirmasi dengan pesan "Apakah Anda yakin ingin logout?". Dialog dilengkapi dengan dua tombol yaitu "*Batal*" berwarna merah untuk membatalkan aksi dan "*Oke*" berwarna pink untuk melanjutkan proses *logout*. Konfirmasi ini mencegah pengguna keluar dari sistem secara tidak sengaja.

Implementasi proses *logout* menggunakan fungsi `logout` yang menangani penghapusan sesi lokal dan pengalihan ke server SSO. Implementasi fungsi tersebut dapat dilihat pada Kode 3.10.

```
1 const logout = async () => {  
2   // Hapus token dari cookies  
3   accessToken.value = null;  
4   refreshToken.value = null;  
5  
6   // Clear localStorage dan sessionStorage
```

```

7  localStorage.clear();
8  sessionStorage.clear();
9
10 // Buat URL logout SSO
11 const callbackUrl = "http://localhost:3000/login";
12 const redirectUri = btoa(callbackUrl); // Base64 encode
13 const ssoLogoutUrl = 'https://auth.airnavindonesia.co.id/user/
    logout?redirect_uri=${redirectUri}';
14
15 // Redirect ke SSO logout
16 window.location.href = ssoLogoutUrl;
17 };

```

Kode 3.10: Fungsi Logout untuk Menghapus Sesi dan Redirect ke SSO

Berdasarkan Kode 3.10, fungsi `logout` melakukan beberapa tahap yaitu menghapus *access token* dan *refresh token* dari *cookies*, membersihkan semua data di *localStorage* dan *sessionStorage*, membuat URL *logout* SSO dengan parameter `redirect_uri` yang di-*encode* menggunakan Base64, dan mengalihkan pengguna ke server AirNav Auth untuk melakukan *logout* dari sistem SSO. Setelah proses *logout* selesai di server SSO, pengguna akan diarahkan kembali ke halaman *login* HAADES sesuai dengan `callbackUrl` yang telah ditentukan.

3.4 Pengujian Fitur

Pengujian fitur pada sistem HAADES dilakukan menggunakan metode *Black Box Testing*, yaitu metode pengujian perangkat lunak yang difokuskan pada fungsi dari sistem tanpa mengetahui detail implementasi internal seperti kode program. Pendekatan ini bertujuan untuk menguji apakah *output* dari suatu fitur sudah sesuai dengan *input* dan ekspektasi pengguna [8]. Pengujian dilakukan dengan membuat berbagai skenario penggunaan untuk memastikan sistem berfungsi dengan baik dalam kondisi normal maupun kondisi *error*.

3.4.1 Fitur Login

Tabel 3.2 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *login* sistem HAADES. Fitur *login* berfungsi sebagai sistem autentikasi menggunakan mekanisme *Single Sign-On* (SSO) melalui AirNav Auth untuk memberikan akses kepada pengguna yang telah terdaftar di sistem AirNav

Indonesia. Fitur ini menggunakan integrasi dengan server AirNav Auth yang menangani proses autentikasi dan validasi kredensial pengguna.

Sistem dilengkapi dengan alur autentikasi yang terdiri dari halaman *welcome*, halaman *login via AirNav Auth*, dan pengalihan ke server AirNav Auth untuk input kredensial. Setelah pengguna berhasil melakukan autentikasi di server AirNav Auth, sistem menerima *access token* dan *refresh token* yang divalidasi sebelum memberikan akses ke *dashboard*. Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk pengalihan ke halaman *login*, validasi *token*, dan proses autentikasi berhasil. Fitur ini memberikan keamanan yang memadai dengan memanfaatkan sistem SSO AirNav Indonesia dan validasi *token* yang tepat.

Tabel 3.2. Pengujian *Black Box* Fitur Login.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses halaman <i>login</i>	Klik tombol "Start" pada halaman <i>welcome</i>	Sistem mengarahkan ke halaman "Masuk ke HAADES"	Sistem mengarahkan ke halaman "Masuk ke HAADES"	Berhasil
2	Pengguna memulai proses SSO	Klik tombol "Login via AirNav Auth"	Sistem mengarahkan ke server AirNav Auth	Sistem mengarahkan ke server AirNav Auth	Berhasil
3	Pengguna tidak mengisi <i>username</i> di AirNav Auth	<i>Username</i> kosong, <i>password</i> diisi	Tampil pesan error "Username wajib diisi"	Tampil pesan error "Username wajib diisi"	Berhasil
4	Pengguna tidak mengisi <i>password</i> di AirNav Auth	<i>Username</i> diisi, <i>password</i> kosong	Tampil pesan error "Password wajib diisi"	Tampil pesan error "Password wajib diisi"	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.2 Pengujian *Black Box* Fitur Login (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
5	Pengguna memasukkan kredensial salah	<i>Username</i> atau <i>password</i> salah	Tampil pesan error "Kredensial tidak valid"	Tampil pesan error "Kredensial tidak valid"	Berhasil
6	Pengguna gagal validasi <i>captcha</i>	<i>Captcha</i> tidak valid	Tampil pesan error "Captcha tidak valid"	Tampil pesan error "Captcha tidak valid"	Berhasil
7	<i>Token</i> tidak valid setelah <i>redirect</i>	<i>Token</i> invalid dari SSO	Sistem <i>redirect</i> ke halaman <i>login</i> dengan pesan error	Sistem <i>redirect</i> ke halaman <i>login</i> dengan pesan error	Berhasil
8	Autentikasi berhasil dengan kredensial benar	<i>Username</i> , <i>password</i> , dan <i>captcha</i> benar	Sistem menerima <i>token</i> , validasi berhasil, dan <i>redirect</i> ke <i>dashboard</i>	Sistem menerima <i>token</i> , validasi berhasil, dan <i>redirect</i> ke <i>dashboard</i>	Berhasil

3.4.2 Fitur Dashboard

Tabel 3.3 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *dashboard* sistem HAADES. Fitur *dashboard* berfungsi sebagai halaman utama yang menampilkan informasi *traffic overflying* dan pendapatan secara periodik dalam bentuk kartu informasi dan grafik. Fitur ini terdiri dari empat kartu informasi pendapatan yaitu *Annual Revenue*, *Monthly Revenue*, *Daily Revenue*, dan *Potential Daily*, serta grafik tren untuk periode tahunan, bulanan, dan harian.

Sistem dilengkapi dengan fitur pemilihan rentang tanggal menggunakan *date picker*, tombol "View Data" untuk menampilkan data sesuai periode yang dipilih, tabel *Revenue Information* yang dapat dibuka atau ditutup, dan fitur *download* grafik dalam format PNG atau JPG. Berdasarkan pengujian *black*

box, semua skenario pengujian berhasil dilakukan termasuk pengambilan data *dashboard*, pemilihan periode tanggal, ekspor grafik, dan pengelolaan tabel informasi. Fitur ini memberikan visualisasi data yang informatif dengan grafik interaktif dan kemampuan ekspor untuk keperluan pelaporan.

Tabel 3.3. Pengujian *Black Box* Fitur Dashboard.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses <i>dashboard</i>	Login berhasil dan masuk ke <i>dashboard</i>	Tampil empat kartu informasi dan grafik <i>default</i>	Tampil empat kartu informasi dan grafik <i>default</i>	Berhasil
2	Pengguna melihat data kartu informasi	Kartu <i>Annual</i> , <i>Monthly</i> , <i>Daily</i> , dan <i>Potential Daily</i>	Tampil nilai pendapatan sesuai periode	Tampil nilai pendapatan sesuai periode	Berhasil
3	Pengguna memilih rentang tanggal	Pilih tanggal <i>From/To</i> menggunakan <i>date picker</i>	Tanggal terpilih dan tersimpan	Tanggal terpilih dan tersimpan	Berhasil
4	Pengguna menampilkan data berdasarkan periode	Klik tombol " <i>View Data</i> " setelah pilih tanggal	Grafik dan tabel menampilkan data sesuai periode	Grafik dan tabel menampilkan data sesuai periode	Berhasil
5	Pengguna membuka tabel <i>Revenue Information</i>	Klik ikon <i>expand</i> pada tabel	Tabel terbuka dan menampilkan detail pendapatan	Tabel terbuka dan menampilkan detail pendapatan	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.3 Pengujian *Black Box* Fitur Dashboard (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
6	Pengguna menutup tabel <i>Revenue Information</i>	Klik ikon <i>collapse</i> pada tabel	Tabel tertutup dan tersembunyi	Tabel tertutup dan tersembunyi	Berhasil
7	Pengguna menggunakan fitur pencarian tabel	Input teks pada <i>search box</i> tabel	Data tabel terfilter sesuai kata kunci	Data tabel terfilter sesuai kata kunci	Berhasil
8	Pengguna membuka <i>dropdown export</i> grafik	Klik ikon <i>dropdown</i> pada grafik	Tampil opsi " <i>Export PNG</i> " dan " <i>Export JPG</i> "	Tampil opsi " <i>Export PNG</i> " dan " <i>Export JPG</i> "	Berhasil
9	Pengguna mengekspor grafik dalam format PNG	Pilih " <i>Export PNG</i> " dari <i>dropdown</i>	Grafik terunduh dalam format PNG	Grafik terunduh dalam format PNG	Berhasil
10	Pengguna mengekspor grafik dalam format JPG	Pilih " <i>Export JPG</i> " dari <i>dropdown</i>	Grafik terunduh dalam format JPG	Grafik terunduh dalam format JPG	Berhasil
11	Pengguna melihat grafik <i>Annual Traffic Chart</i>	Scroll ke bagian grafik tahunan	Tampil grafik <i>traffic</i> per bulan tahun 2025	Tampil grafik <i>traffic</i> per bulan tahun 2025	Berhasil
12	Pengguna melihat grafik <i>Annual Revenue Chart</i>	Scroll ke bagian grafik tahunan	Tampil grafik pendapatan per bulan tahun 2025	Tampil grafik pendapatan per bulan tahun 2025	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.3 Pengujian *Black Box* Fitur Dashboard (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
13	Sistem tidak dapat mengambil data dari API	Koneksi API gagal atau error	Tampil pesan error "Failed to fetch data"	Tampil pesan error "Failed to fetch data"	Berhasil

3.4.3 Fitur Kurs

Tabel 3.4 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *kurs* sistem HAADES. Fitur *kurs* berfungsi untuk melakukan sinkronisasi data penerbangan dengan *update* nilai tukar mata uang sesuai periode waktu tertentu. Fitur ini dapat diakses melalui *dropdown* profil pengguna dengan memilih opsi "Kurs" dan terdiri dari form input untuk menambahkan data kurs baru serta tabel yang menampilkan histori nilai kurs periode sebelumnya.

Sistem dilengkapi dengan form "Input Kurs Validity" yang memiliki field Nilai Rp, Start Date, End Date, dan Tariff. Tabel "All Kurs" menampilkan data dengan kolom NO, NILAI RP, START DATE, END DATE, TARIFF, dan ACTION yang berisi tombol edit dan hapus. Sistem juga dilengkapi validasi untuk memastikan End Date lebih besar dari Start Date dan modal "Edit Kurs" untuk mengubah data yang sudah ada. Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk penambahan data kurs, validasi tanggal, edit data, hapus data, dan fitur pencarian. Fitur ini memberikan pengelolaan data kurs yang efektif dengan validasi yang tepat untuk mencegah kesalahan input.

Tabel 3.4. Pengujian *Black Box* Fitur Kurs.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses halaman kurs	Klik menu "Kurs" dari <i>dropdown</i> profil	Tampil form input dan tabel All Kurs	Tampil form input dan tabel All Kurs	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.4 Pengujian *Black Box* Fitur Kurs (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
2	Pengguna mengisi form kurs lengkap	Isi semua field dan klik "Submit"	Data tersimpan dan tampil di tabel	Data tersimpan dan tampil di tabel	Berhasil
3	Pengguna tidak mengisi Nilai Rp	Field Nilai Rp kosong	Tampil pesan error "Nilai Rp wajib diisi"	Tampil pesan error "Nilai Rp wajib diisi"	Berhasil
4	Pengguna tidak mengisi Start Date	Field Start Date kosong	Tampil pesan error "Start Date wajib diisi"	Tampil pesan error "Start Date wajib diisi"	Berhasil
5	Pengguna tidak mengisi End Date	Field End Date kosong	Tampil pesan error "End Date wajib diisi"	Tampil pesan error "End Date wajib diisi"	Berhasil
6	Pengguna memasukkan End Date lebih kecil dari Start Date	End Date < Start Date	Tampil pesan error "End Date harus lebih besar dari Start Date"	Tampil pesan error "End Date harus lebih besar dari Start Date"	Berhasil
7	Pengguna menggunakan date picker	Klik icon kalender pada field tanggal	Tampil kalender untuk memilih tanggal	Tampil kalender untuk memilih tanggal	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.4 Pengujian *Black Box* Fitur Kurs (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
8	Pengguna mengubah jumlah data per halaman	Pilih opsi 10, 25, atau 50 dari <i>dropdown</i>	Tabel menampilkan jumlah data sesuai pilihan	Tabel menampilkan jumlah data sesuai pilihan	Berhasil
9	Pengguna mencari data kurs	Input kata kunci di <i>search box</i>	Data tabel terfilter sesuai kata kunci	Data tabel terfilter sesuai kata kunci	Berhasil
10	Pengguna mengklik tombol edit	Klik icon edit pada kolom <i>ACTION</i>	Tampil modal " <i>Edit Kurs</i> " dengan data terisi	Tampil modal " <i>Edit Kurs</i> " dengan data terisi	Berhasil
11	Pengguna mengubah data di modal edit	Ubah data dan klik " <i>Update</i> "	Data terupdate di tabel	Data terupdate di tabel	Berhasil
12	Pengguna membatalkan edit	Klik tombol " <i>Cancel</i> " di modal edit	Modal tertutup tanpa perubahan data	Modal tertutup tanpa perubahan data	Berhasil
13	Pengguna menghapus data kurs	Klik icon hapus pada kolom <i>ACTION</i>	Tampil konfirmasi dan data terhapus jika dikonfirmasi	Tampil konfirmasi dan data terhapus jika dikonfirmasi	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.4 Pengujian *Black Box* Fitur Kurs (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
14	Pengguna mengekspor data kurs	Klik tombol <i>download</i> di pojok kanan atas	Data kurs terunduh dalam format file	Data kurs terunduh dalam format file	Berhasil
15	Sistem gagal menyimpan data	Koneksi API gagal saat submit	Tampil pesan error " <i>Failed to save data</i> "	Tampil pesan error " <i>Failed to save data</i> "	Berhasil

3.4.4 Fitur FPS (Flight Progress Strip)

Tabel 3.5 menyajikan hasil pengujian *black box* yang dilakukan pada fitur FPS (*Flight Progress Strip*) sistem HAADES. Fitur FPS berfungsi untuk melakukan validasi dan penginputan data penerbangan lintas (*overflying*) berdasarkan data dukung FPS dan hanya dapat diakses oleh akun cabang JATSC. Menu ini berfungsi sebagai pendukung validasi data penerbangan selain dari sumber AFTN dan AIDC. Fitur ini terdiri dari bagian "*Query Data*" untuk memilih tanggal penerbangan, tabel "*Data Invoice OVF JATSC*" untuk menampilkan data, dan tiga tombol aksi yaitu "*Create FPS*", "*Create FPS 2*", dan "*Single FPS*" untuk input data manual.

Sistem dilengkapi dengan modal "*Create FPS*" yang memiliki field *Flight Number*, *ADEP*, *ADES*, *DOF*, *rute*, serta waktu masuk dan keluar wilayah. Modal "*Create FPS 2*" memiliki struktur yang sama dengan tambahan field untuk *Aircraft Registration*, *Aircraft Type*, dan *Route*. Modal "*Single FPS*" digunakan untuk melakukan *query* dan menampilkan data penerbangan berdasarkan tanggal dan *callsign* dengan tabel *List Data* yang menampilkan detail penerbangan. Modal "*Validasi FPS*" menampilkan detail data penerbangan untuk dilakukan verifikasi sebelum disimpan ke basis data. Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk pemilihan tanggal, penambahan data FPS melalui berbagai modal, validasi data, dan pencarian data. Fitur ini memberikan fleksibilitas input data penerbangan secara manual dengan berbagai opsi sesuai kebutuhan validasi.

Tabel 3.5. Pengujian *Black Box* Fitur FPS.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses halaman FPS	Login sebagai akun JATSC dan akses menu FPS	Tampil bagian <i>Query Data</i> dan tabel data	Tampil bagian <i>Query Data</i> dan tabel data	Berhasil
2	Pengguna memilih tanggal penerbangan	Pilih tanggal dari <i>date picker</i>	Tanggal terpilih pada field <i>Select Date</i>	Tanggal terpilih pada field <i>Select Date</i>	Berhasil
3	Pengguna menampilkan data penerbangan	Klik tombol "View" setelah pilih tanggal	Tabel menampilkan data penerbangan sesuai tanggal	Tabel menampilkan data penerbangan sesuai tanggal	Berhasil
4	Pengguna membuka modal <i>Create FPS</i>	Klik tombol "Create FPS"	Tampil modal form <i>Create FPS</i>	Tampil modal form <i>Create FPS</i>	Berhasil
5	Pengguna mengisi form <i>Create FPS</i>	Isi semua field dan klik "Create"	Data tersimpan dan tampil di tabel	Data tersimpan dan tampil di tabel	Berhasil
6	Pengguna tidak mengisi field wajib di <i>Create FPS</i>	Field wajib kosong	Tampil pesan error field wajib diisi	Tampil pesan error field wajib diisi	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.5 Pengujian *Black Box* Fitur FPS (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
7	Pengguna membatalkan <i>Create FPS</i>	Klik tombol "Close" di modal	Modal tertutup tanpa menyimpan data	Modal tertutup tanpa menyimpan data	Berhasil
8	Pengguna membuka modal <i>Create FPS 2</i>	Klik tombol "Create FPS 2"	Tampil modal form <i>Create FPS 2</i>	Tampil modal form <i>Create FPS 2</i>	Berhasil
9	Pengguna mengisi form <i>Create FPS 2</i>	Isi semua field termasuk <i>Aircraft Registration</i> dan klik "Create"	Data tersimpan dan tampil di tabel	Data tersimpan dan tampil di tabel	Berhasil
10	Pengguna membuka modal <i>Single FPS</i>	Klik tombol "Single FPS"	Tampil modal <i>Query Data</i> dan <i>List Data</i>	Tampil modal <i>Query Data</i> dan <i>List Data</i>	Berhasil
11	Pengguna melakukan <i>query</i> di <i>Single FPS</i>	Isi tanggal dan <i>callsign</i> , klik "View"	Tabel <i>List Data</i> menampilkan hasil <i>query</i>	Tabel <i>List Data</i> menampilkan hasil <i>query</i>	Berhasil
12	Pengguna membuat FPS dari <i>Single FPS</i>	Klik tombol "Create FPS" di modal <i>Single FPS</i>	Tampil form untuk membuat FPS baru	Tampil form untuk membuat FPS baru	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.5 Pengujian *Black Box* Fitur FPS (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
13	Pengguna menutup modal <i>Single FPS</i>	Klik tombol "Close"	Modal <i>Single FPS</i> tertutup	Modal <i>Single FPS</i> tertutup	Berhasil
14	Pengguna membuka detail data untuk validasi	Klik tombol <i>action</i> pada tabel	Tampil modal <i>Validasi FPS</i> dengan detail data	Tampil modal <i>Validasi FPS</i> dengan detail data	Berhasil
15	Pengguna memvalidasi data FPS	Klik tombol "Validate" di modal validasi	Status FPS berubah menjadi valid dan tersimpan	Status FPS berubah menjadi valid dan tersimpan	Berhasil
16	Pengguna menggunakan fitur pencarian	Input kata kunci di <i>search box</i> tabel	Data tabel terfilter sesuai kata kunci	Data tabel terfilter sesuai kata kunci	Berhasil
17	Pengguna mengubah jumlah data per halaman	Pilih opsi jumlah data dari <i>dropdown</i>	Tabel menampilkan jumlah data sesuai pilihan	Tabel menampilkan jumlah data sesuai pilihan	Berhasil
18	Sistem gagal menyimpan data FPS	Koneksi API gagal saat submit	Tampil pesan error "Failed to save data"	Tampil pesan error "Failed to save data"	Berhasil

3.4.5 Fitur Entry MTOW

Tabel 3.6 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *Entry MTOW* sistem HAADES. Fitur *Entry MTOW* berfungsi untuk menambahkan data registrasi pesawat yang belum ada di database lokal HAADES. Fitur ini

terdiri dari form *"Form Input MTOW"* untuk input manual dengan field *AC REG*, *AIRLINES*, *TYPE AIRCRAFT*, *MTOW*, dan *FAKTOR BERAT*, serta bagian *"MTOW File Upload"* untuk mengunggah data MTOW secara massal melalui file.

Sistem dilengkapi dengan validasi field wajib pada *AC REG* dan *AIRLINES* yang ditandai dengan asterisk (*). Bagian *file upload* memiliki area *file selector* dengan tombol *"Browse"* untuk memilih file, tombol *"EMPTY"* berwarna merah untuk membatalkan pilihan file, dan tombol *"UPLOAD"* berwarna hijau untuk mengunggah file ke sistem. Sistem hanya menerima file dalam format Excel (.xls/.xlsx) atau CSV (.csv). Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk input data manual, validasi field wajib, upload file MTOW, dan validasi format file. Fitur ini memberikan fleksibilitas dalam menambahkan data MTOW baik secara manual maupun massal melalui file upload.

Tabel 3.6. Pengujian *Black Box* Fitur Entry MTOW.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses halaman <i>Entry MTOW</i>	Klik menu <i>Entry MTOW</i>	Tampil form input dan bagian <i>file upload</i>	Tampil form input dan bagian <i>file upload</i>	Berhasil
2	Pengguna mengisi form lengkap	Isi semua field dan klik <i>"Submit"</i>	Data tersimpan ke database	Data tersimpan ke database	Berhasil
3	Pengguna tidak mengisi <i>AC REG</i>	Field <i>AC REG</i> kosong	Tampil pesan error <i>"AC Register adalah field wajib"</i>	Tampil pesan error <i>"AC Register adalah field wajib"</i>	Berhasil
4	Pengguna tidak mengisi <i>AIRLINES</i>	Field <i>AIRLINES</i> kosong	Tampil pesan error <i>"Airlines adalah field wajib"</i>	Tampil pesan error <i>"Airlines adalah field wajib"</i>	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.6 Pengujian *Black Box* Fitur Entry MTOW (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
5	Pengguna mengisi <i>MTOW</i> dengan format salah	Input karakter non-angka di field <i>MTOW</i>	Tampil pesan error format tidak valid	Tampil pesan error format tidak valid	Berhasil
6	Pengguna mengisi <i>FAKTOR BERAT</i> dengan format salah	Input karakter non-angka di field <i>FAKTOR BERAT</i>	Tampil pesan error format tidak valid	Tampil pesan error format tidak valid	Berhasil
7	Data berhasil disimpan	Submit form dengan data valid	Form direset dan tampil pesan sukses	Form direset dan tampil pesan sukses	Berhasil
8	Pengguna membuka <i>file selector</i>	Klik tombol " <i>Browse</i> "	Tampil dialog pemilihan file	Tampil dialog pemilihan file	Berhasil
9	Pengguna memilih file Excel	Pilih file .xls atau .xlsx	File terpilih dan nama file tampil	File terpilih dan nama file tampil	Berhasil
10	Pengguna memilih file CSV	Pilih file .csv	File terpilih dan nama file tampil	File terpilih dan nama file tampil	Berhasil
11	Pengguna memilih file dengan format tidak didukung	Pilih file .pdf atau format lain	Tampil pesan error " <i>Format file tidak didukung</i> "	Tampil pesan error " <i>Format file tidak didukung</i> "	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.6 Pengujian *Black Box* Fitur Entry MTOW (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
12	Pengguna membatalkan pilihan file	Klik tombol "EMPTY"	Pilihan file dibatalkan dan area kosong kembali	Pilihan file dibatalkan dan area kosong kembali	Berhasil
13	Pengguna mengunggah file tanpa memilih file	Klik "UPLOAD" tanpa pilih file	Tampil pesan error "Pilih file terlebih dahulu"	Tampil pesan error "Pilih file terlebih dahulu"	Berhasil
14	Pengguna mengunggah file yang valid	Pilih file Excel/CSV dan klik "UPLOAD"	File terupload dan data MTOW tersimpan	File terupload dan data MTOW tersimpan	Berhasil
15	File upload dengan data duplikat	Upload file berisi AC REG yang sudah ada	Sistem menampilkan peringatan data duplikat	Sistem menampilkan peringatan data duplikat	Berhasil
16	Sistem gagal menyimpan data	Koneksi API gagal	Tampil pesan error "Failed to save data"	Tampil pesan error "Failed to save data"	Berhasil

3.4.6 Fitur Upload A-CARE

Tabel 3.7 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *Upload A-CARE* sistem HAADES. Fitur *Upload A-CARE* berfungsi untuk mengunggah data invoice ENC dari hasil download aplikasi A-CARE dalam format file Excel (.xls/.xlsx) atau CSV (.csv). Fitur ini terdiri dari panel "Import Data A-CARE" untuk upload file, bagian "A-CARE Data List" dengan field *Select DOF From/To* untuk memilih rentang tanggal dan tombol "View Data" untuk menampilkan data, serta tabel "Data Invoice OVF JATSC" yang menampilkan data

invoice yang telah diunggah.

Sistem dilengkapi dengan tombol "*Choose File*" untuk memilih file dari komputer, informasi format file yang didukung (.xls/.xlsx/.csv), link "*One File*" sebagai panduan format, tombol "*Upload File*" berwarna biru untuk mengunggah file ke sistem, dan tombol "*View (Simulasi)*" untuk melihat contoh data. Tabel menampilkan kolom *ID*, *CALLSIGN*, *ADEP*, *ADES*, *DOF*, *REG*, *AC TYPE*, *AIR CODE*, *POINT IN/OUT*, dan *TIME IN/OUT*, dilengkapi dengan fitur pencarian untuk memfilter data berdasarkan *CallSign*, *REG*, atau nomor invoice. File yang diunggah hanya dapat berisi data invoice ENC OVF dari cabang JATSC atau MATSC. Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk pemilihan file, validasi format file, upload file, pemilihan rentang tanggal, dan pencarian data. Fitur ini memberikan kemudahan dalam mengunggah dan mengelola data invoice A-CARE secara massal.

Tabel 3.7. Pengujian *Black Box* Fitur Upload A-CARE.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna mengakses halaman <i>Upload A-CARE</i>	Klik menu <i>Upload A-CARE</i>	Tampil panel <i>Import</i> , bagian <i>Data List</i> , dan tabel	Tampil panel <i>Import</i> , bagian <i>Data List</i> , dan tabel	Berhasil
2	Pengguna membuka <i>file selector</i>	Klik tombol " <i>Choose File</i> "	Tampil dialog pemilihan file	Tampil dialog pemilihan file	Berhasil
3	Pengguna memilih file Excel	Pilih file .xls atau .xlsx	File terpilih dan nama file tampil	File terpilih dan nama file tampil	Berhasil
4	Pengguna memilih file CSV	Pilih file .csv	File terpilih dan nama file tampil	File terpilih dan nama file tampil	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.7 Pengujian *Black Box* Fitur Upload A-CARE (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
5	Pengguna memilih file dengan format tidak didukung	Pilih file selain .xls/.xlsx/.csv	Tampil pesan error "Format file tidak didukung"	Tampil pesan error "Format file tidak didukung"	Berhasil
6	Pengguna mengklik link "One File"	Klik link "One File"	Tampil atau download panduan format file	Tampil atau download panduan format file	Berhasil
7	Pengguna mengklik tombol "View (Simulasi)"	Klik tombol "View (Simulasi)"	Tampil contoh data A-CARE	Tampil contoh data A-CARE	Berhasil
8	Pengguna mengunggah file tanpa memilih file	Klik "Upload File" tanpa pilih file	Tampil pesan error "Pilih file terlebih dahulu"	Tampil pesan error "Pilih file terlebih dahulu"	Berhasil
9	Pengguna mengunggah file yang valid	Pilih file Excel/CSV dan klik "Upload File"	File terupload dan data tampil di tabel	File terupload dan data tampil di tabel	Berhasil
10	File upload dengan format header salah	Upload file dengan header tidak sesuai	Tampil pesan error "Format header tidak sesuai"	Tampil pesan error "Format header tidak sesuai"	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.7 Pengujian *Black Box* Fitur Upload A-CARE (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
11	File upload dengan data duplikat	Upload file berisi data yang sudah ada	Sistem menampilkan peringatan data duplikat	Sistem menampilkan peringatan data duplikat	Berhasil
12	Pengguna memilih rentang tanggal	Pilih <i>DOF From/To</i> menggunakan <i>date picker</i>	Tanggal terpilih pada field	Tanggal terpilih pada field	Berhasil
13	Pengguna menampilkan data berdasarkan tanggal	Pilih tanggal dan klik "View Data"	Tabel menampilkan data sesuai rentang tanggal	Tabel menampilkan data sesuai rentang tanggal	Berhasil
14	Pengguna menggunakan fitur pencarian	Input kata kunci di <i>search box</i>	Data tabel terfilter sesuai <i>CallSign</i> , <i>REG</i> , atau <i>invoice</i>	Data tabel terfilter sesuai <i>CallSign</i> , <i>REG</i> , atau <i>invoice</i>	Berhasil
15	Pengguna mengubah jumlah data per halaman	Pilih opsi jumlah data dari <i>dropdown</i>	Tabel menampilkan jumlah data sesuai pilihan	Tabel menampilkan jumlah data sesuai pilihan	Berhasil
16	Sistem gagal mengunggah file	Koneksi API gagal saat upload	Tampil pesan error "Failed to upload file"	Tampil pesan error "Failed to upload file"	Berhasil

3.4.7 Fitur Logout

Tabel 3.8 menyajikan hasil pengujian *black box* yang dilakukan pada fitur *logout* sistem HAADES. Fitur *logout* memungkinkan pengguna untuk keluar

dari sistem HAADES dengan aman. Proses *logout* tidak hanya menghapus sesi lokal di aplikasi, tetapi juga melakukan *logout* dari server AirNav Auth untuk memastikan keamanan autentikasi SSO. Fitur ini dapat diakses melalui *dropdown* profil pengguna di pojok kanan atas dengan memilih opsi "*Logout*".

Sistem dilengkapi dengan dialog konfirmasi yang menampilkan pesan "Apakah Anda yakin ingin logout?" ketika pengguna mengklik opsi *logout*. Dialog ini memiliki dua tombol yaitu "*Batal*" berwarna merah untuk membatalkan aksi dan "*Oke*" berwarna pink untuk melanjutkan proses *logout*. Setelah konfirmasi, sistem akan menghapus *access token* dan *refresh token* dari *cookies*, membersihkan semua data di *localStorage* dan *sessionStorage*, kemudian mengalihkan pengguna ke server AirNav Auth untuk melakukan *logout* dari sistem SSO. Setelah proses *logout* selesai di server SSO, pengguna akan diarahkan kembali ke halaman *login* HAADES. Berdasarkan pengujian *black box*, semua skenario pengujian berhasil dilakukan termasuk konfirmasi *logout*, pembatalan *logout*, dan proses *logout* lengkap. Fitur ini memberikan keamanan yang memadai dengan konfirmasi sebelum keluar dan penghapusan sesi yang menyeluruh baik di aplikasi maupun di server SSO.

Tabel 3.8. Pengujian *Black Box* Fitur Logout.

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
1	Pengguna membuka <i>dropdown</i> profil	Klik icon profil di pojok kanan atas	Tampil menu <i>dropdown</i> dengan opsi " <i>Logout</i> "	Tampil menu <i>dropdown</i> dengan opsi " <i>Logout</i> "	Berhasil
2	Pengguna mengklik opsi " <i>Logout</i> "	Klik " <i>Logout</i> " dari <i>dropdown</i>	Tampil dialog konfirmasi "Apakah Anda yakin ingin logout?"	Tampil dialog konfirmasi "Apakah Anda yakin ingin logout?"	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.8 Pengujian *Black Box* Fitur Logout (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
3	Dialog konfirmasi menampilkan dua tombol	Dialog konfirmasi terbuka	Tampil tombol "Batal" dan "Oke"	Tampil tombol "Batal" dan "Oke"	Berhasil
4	Pengguna membatalkan <i>logout</i>	Klik tombol "Batal" pada dialog	Dialog tertutup dan pengguna tetap di halaman	Dialog tertutup dan pengguna tetap di halaman	Berhasil
5	Pengguna mengkonfirmasi <i>logout</i>	Klik tombol "Oke" pada dialog	Sistem memproses <i>logout</i>	Sistem memproses <i>logout</i>	Berhasil
6	Sistem menghapus <i>token</i> lokal	Proses <i>logout</i> dimulai	<i>Access token</i> dan <i>refresh token</i> dihapus dari <i>cookies</i>	<i>Access token</i> dan <i>refresh token</i> dihapus dari <i>cookies</i>	Berhasil
7	Sistem membersihkan <i>storage</i>	Proses <i>logout</i> berlanjut	<i>localStorage</i> dan <i>sessionStorage</i> dibersihkan	<i>localStorage</i> dan <i>sessionStorage</i> dibersihkan	Berhasil
8	Sistem mengalihkan ke server SSO	Setelah hapus sesi lokal	Pengguna dialihkan ke URL <i>logout</i> AirNav Auth	Pengguna dialihkan ke URL <i>logout</i> AirNav Auth	Berhasil
9	Server SSO melakukan <i>logout</i>	Di server AirNav Auth	Sesi SSO dihapus di server	Sesi SSO dihapus di server	Berhasil
Lanjut pada halaman berikutnya					

Tabel 3.8 Pengujian *Black Box* Fitur Logout (lanjutan)

No.	Skenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Hasil
10	Pengguna diarahkan kembali ke halaman <i>login</i>	Setelah <i>logout</i> SSO selesai	Pengguna berada di halaman <i>login</i> HAADES	Pengguna berada di halaman <i>login</i> HAADES	Berhasil
11	Pengguna mencoba akses halaman dengan <i>token</i> lama	Gunakan <i>token</i> yang sudah di- <i>logout</i>	Sistem menolak akses dan <i>redirect</i> ke <i>login</i>	Sistem menolak akses dan <i>redirect</i> ke <i>login</i>	Berhasil
12	Pengguna menekan tombol <i>back</i> di browser	Setelah <i>logout</i> selesai	Sistem tetap di halaman <i>login</i> atau <i>redirect</i> ke <i>login</i>	Sistem tetap di halaman <i>login</i> atau <i>redirect</i> ke <i>login</i>	Berhasil

3.5 Kendala

Selama menjalani kerja magang, terdapat beberapa kendala yang muncul dalam proses perancangan dan pengembangan website HAADES, antara lain:

1. Belum memahami framework Nuxt.js yang digunakan sebagai teknologi utama dalam pengembangan website, sehingga mengalami kesulitan dalam memahami struktur project, routing system, dan konsep seperti server-side rendering (SSR).
2. Belum memahami sistem kerja AirNav dan industri penerbangan secara menyeluruh, terutama terkait istilah-istilah teknis seperti MTOW (Maximum Take-Off Weight), overflying, dan point system yang merupakan komponen penting dalam sistem HAADES.

3.6 Solusi

Untuk mengatasi kendala-kendala tersebut, solusi yang diterapkan selama pelaksanaan proyek magang antara lain:

1. Mempelajari dokumentasi resmi Nuxt.js secara mendalam dan mengikuti berbagai tutorial online untuk memahami konsep-konsep dasar hingga implementasi praktis dari framework Nuxt.js.
2. Melakukan diskusi rutin dengan pembimbing lapangan untuk memahami domain bisnis AirNav Indonesia, mempelajari sistem HAADES versi sebelumnya, serta mencari informasi tambahan tentang industri penerbangan melalui berbagai sumber untuk memperluas wawasan terkait sistem AirNav.

