BAB 3
PELAKSANAAN KERJA MAGANG

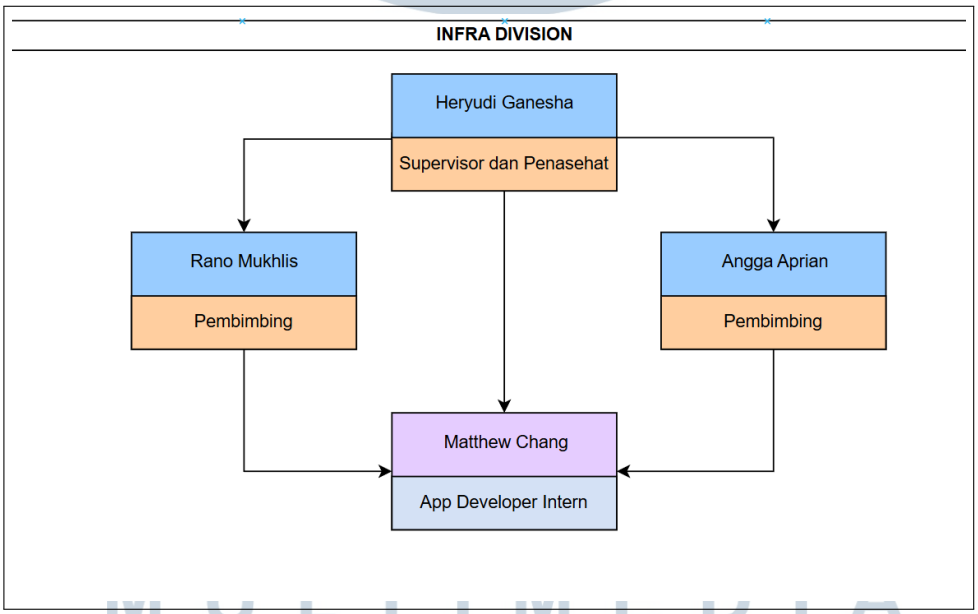
3.1 Kedudukan dan Koordinasi

Dalam kerja magang di PT. Devoteam, memiliki peran dan tanggung jawab dalam koordinasi dan pelaksanaan berbagai tugas.

3.1.1 Kedudukan

Berperan sebagai *App Developer Intern* di divisi INFRA PT. Devoteam, terlibat dalam tim pengembangan aplikasi, termasuk dalam pengembangan aplikasi web serta pengelolaan dan pemeliharaan database, dengan menggunakan teknologi seperti *Greenplum* dan *BigQuery* untuk analisis data.

Bekerja di bawah pengawasan Bapak Heryudi Ganesha sebagai Supervisor dan Penasehat, serta didampingi oleh Bapak Rano Mukhlis dan Bapak Angga Aprian sebagai Pembimbing.



Gambar 3.1. Struktur Supervisi dan Bimbingan Magang

MULTIMEDIA NUSANTARA

3.1.2 Koordinasi

Koordinasi selama magang di PT. Devoteam dilakukan secara fleksibel karena sistem kerja *hybrid*. Saat bekerja di kantor, berkoordinasi langsung dengan tim dan mengikuti pertemuan mingguan untuk membahas progres proyek.

Ketika bekerja *remote*, komunikasi dilakukan melalui pesan instan dan email. Hasil pekerjaan dievaluasi secara rutin oleh pembimbing dan tim, lalu diberikan masukan untuk perbaikan. Juga aktif dalam diskusi teknis agar pengembangan aplikasi tetap sesuai standar perusahaan.

3.2 Tugas yang Dilakukan

Berikut adalah rangkuman tugas-tugas yang telah dilakukan dalam proyek yang diberikan selama menjalani magang:

1. Pembuatan Halaman User Page: Mengembangkan halaman *user page* untuk memungkinkan *User* melakukan submit data dalam aplikasi web HRIS. Halaman ini dirancang untuk mempermudah *User* dalam mengisi dan mengirimkan data yang diperlukan.

2. Pembuatan dan Integrasi Form HRIS: Membuat dan mengintegrasikan *Annual Leave Form* untuk admin HRIS. Halaman ini terhubung langsung dengan backend untuk memastikan data yang dimasukkan oleh admin dapat diproses dan disimpan dengan aman.

3. Pembuatan Fitur Carry Offer dan End Carry Offer: Mengembangkan bagian *carry offer* dan *end carry offer* dalam aplikasi HRIS untuk memudahkan admin dalam mengelola dan memantau penawaran yang sedang berjalan. Endpoint untuk fitur ini juga telah dihubungkan untuk memastikan data diproses dengan benar.

4. Pengembangan Halaman User Master: Menambahkan dan mengembangkan halaman baru untuk *user master*, yang memungkinkan admin HRIS untuk mengelola data *User* dengan efisien, memastikan integrasi yang baik antara frontend dan backend.

5. Validasi Formulir: Mengembangkan validasi agar formulir, seperti

Dental Benefit Form dan *Optical Benefit Form*, hanya dapat disubmit jika semua kolom penting terisi dengan benar, memastikan validitas data yang dikirimkan.

6. Konversi Data SF: Mengkonversi data SF dari *Greenplum* ke *BigQuery* untuk memastikan data dapat dikelola dengan lebih efisien dan lebih cepat diakses. Proses ini melibatkan migrasi data besar yang harus memastikan keakuratan dan kesesuaian format data.

7. Pengujian dan Penyelesaian Konversi Data: Melakukan *test invoke* untuk memastikan bahwa data SF yang telah dikonversi dapat berfungsi dengan baik di sistem *BigQuery*, memastikan integrasi data berjalan lancar.

8. Penyelesaian dan Sinkronisasi Data SF di *BigQuery*: Melakukan penyempurnaan hasil konversi data SF dari *Greenplum* ke *BigQuery* secara berulang selama minggu ke-2 hingga minggu ke-5. Proses ini melibatkan serangkaian *test invoke* untuk memastikan bahwa data yang telah dimigrasikan dapat diakses dan digunakan dengan benar pada sistem HRIS. Selain itu, dilakukan juga pembenahan elemen frontend seperti *breadcrumb* dan halaman *Annual Leave Admin* untuk mendukung integrasi yang lebih baik antar modul.

9. Pengembangan dan Implementasi *Workflow*: Mengembangkan alur kerja (*workflow*) yang terintegrasi untuk mendukung otomatisasi proses pengolahan data SF. Tahapan ini mencakup perancangan, pembuatan, hingga implementasi *workflow* yang diujikan dalam skenario nyata, termasuk untuk proyek Smartfren.

10. *Data Mapping* untuk Integrasi Sistem Baru: Melakukan proses *data mapping* untuk sistem SF dan CIMB, guna menyusun struktur data yang akan digunakan dalam integrasi lintas proyek. Aktivitas ini mencakup identifikasi format, konversi struktur, dan pemetaan antar entitas data untuk memastikan interoperabilitas sistem berjalan optimal.

3.3 Uraian Pelaksanaan Magang

Berikut adalah uraian mengenai pelaksanaan kerja magang yang telah dilakukan. Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

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

Minggu Ke -	Pekerjaan yang dilakukan
1	Onboarding, setup environment, belajar *Google Cloud Skillboost*, membuat fungsi *GET/POST* di *React*, dan implementasi *PWA*.
2	Membuat FAQ dinamis, menghubungkan endpoint, dan mengembangkan form *Dental* serta *Optical Benefit*.
3	Membuat *user page*, fitur *carry offer*, dan konversi data dari *Greenplum* ke *BigQuery*.
4	Melanjutkan konversi data SF ke *BigQuery*.
5	Penyempurnaan hasil konversi data SF dan uji integrasi.
6	Membuat *Annual Leave Form* dan melanjutkan migrasi data SF.
7	Mengembangkan fitur *carry offer*, dan *test invoke* data SF.
8	Validasi dan pengujian hasil konversi data SF.
9	Menyempurnakan data SF dan memperbarui halaman *Annual Leave Admin*.
10	Finalisasi migrasi SF dan pengujian integrasi sistem.
11	Pembaruan tampilan dan sinkronisasi fitur *Leave Form*.
12	Validasi SF dan pengembangan awal *workflow* integrasi.
13	Implementasi *workflow* dan konversi data analitik.
14	*Mapping* awal data SF proyek Smartfren.
15	Lanjutan *mapping* SF dan awal *mapping* CIMB.
16	Penyelesaian *mapping* data CIMB.

3.4 Perancangan Sistem *Leave* pada Aplikasi HRIS

Proyek ini dirancang dengan menggunakan beberapa jenis diagram untuk menggambarkan alur dan fungsionalitas sistem pengajuan *leave* dalam aplikasi HRIS, seperti *Data Flow Diagram* (DFD) dan *Flowchart*. *Data Flow Diagram* (DFD) digunakan untuk menggambarkan aliran data dalam sistem, mulai dari pengajuan *leave* oleh *User* hingga persetujuan atau penolakan oleh *Reporting Manager*. DFD ini mempermudah pemahaman mengenai interaksi antara entitas sistem, seperti *User*, Admin, dan *Reporting Manager*, serta proses pengelolaan data *leave*. *Flowchart* digunakan untuk menggambarkan langkah-langkah rinci dalam

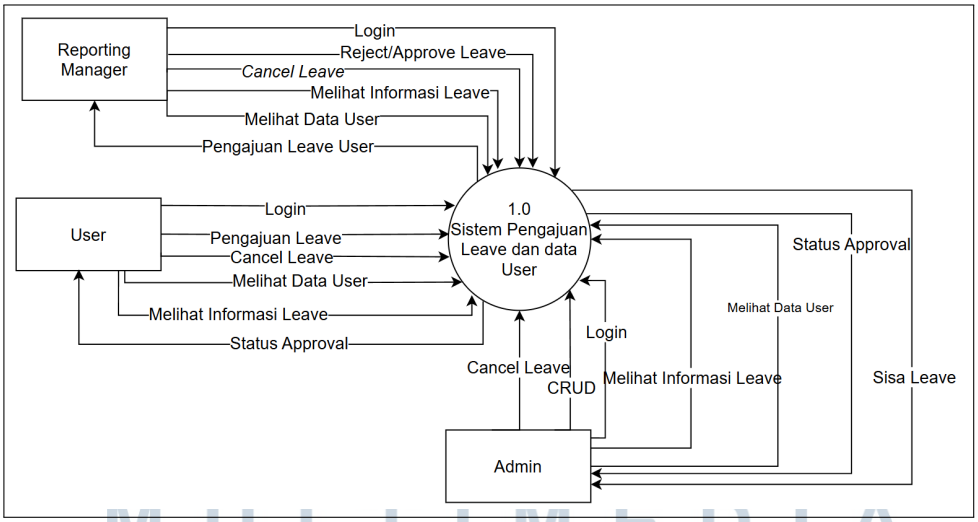
proses pengajuan *leave*, seperti pengecekan *leave balance*, pengisian formulir, dan validasi data. *Flowchart* ini membantu memperjelas alur kerja yang terjadi dalam setiap tahapan pengajuan *leave*.

3.4.1 *Data Flow Diagram Leave*

Data Flow Diagram (DFD) bertujuan untuk mempermudah pemahaman mengenai alur data yang terjadi dalam sistem. Tahapan ini terdiri dari *Context Diagram*, DFD Level 1, dan DFD Level 2, yang masing-masing menggambarkan tingkatan detail yang berbeda dalam mendeskripsikan aliran data dalam sistem.

A *Context Diagram Leave*

Pada Gambar 3.2 terlihat bahwa sistem pengajuan *leave* dan data *User* berinteraksi dengan tiga entitas eksternal utama, yaitu *User*, Admin, dan *Reporting Manager*. *User* dapat mengajukan *leave*, membatalkan *leave*, serta melihat status persetujuan dan data mereka sendiri. Admin bertugas mengelola data *User* dan *leave*, serta memberikan akses yang diperlukan. *Reporting Manager* memiliki peran dalam memberikan persetujuan atau penolakan atas permohonan *leave User*.



Gambar 3.2. *Context Diagram Leave* dan data *User*

MULTIMEDIA NUSANTARA

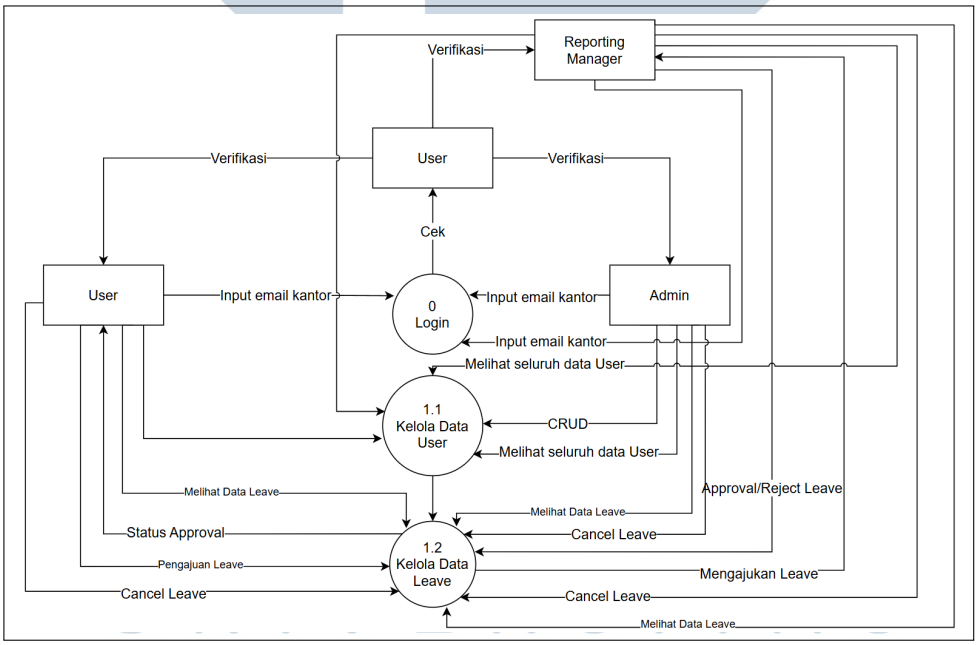
B *Data Flow Diagram* Level 1

Pada Gambar 3.3, *DFD* Level 1 menunjukkan tiga proses utama yang ada dalam sistem pengajuan *leave* dan data *User*. Proses pertama adalah *Login*, yang

melibatkan dua entitas eksternal, yaitu *User* dan Admin, yang memasukkan email kantor mereka untuk mendapatkan akses ke dalam sistem. Proses ini berfungsi untuk memverifikasi keberadaan dan keabsahan email kantor yang digunakan untuk login, serta memastikan hanya *User* yang terdaftar yang dapat mengakses sistem.

Proses kedua adalah Kelola Data *User*, yang berfokus pada pengelolaan informasi *User* oleh Admin. Proses ini memungkinkan Admin untuk melakukan operasi *CRUD* (Create, Read, Update, Delete) pada data *User* yang ada dalam sistem. Admin dapat menambah, mengubah, atau menghapus data *User* sesuai dengan kebutuhan organisasi, serta melihat seluruh data *User* yang tersedia.

Proses ketiga adalah Kelola Data *Leave*, yang memungkinkan *User* untuk mengajukan *leave*, membatalkan *leave*, serta memeriksa status pengajuan mereka. Dalam proses ini, Admin bertugas untuk memproses permohonan *leave* dari *User*, sementara *Reporting Manager* memiliki wewenang untuk menyetujui atau menolak pengajuan yang diajukan. Proses ini juga memungkinkan Admin untuk mengelola status pengajuan *leave*, baik yang diterima maupun yang ditolak.

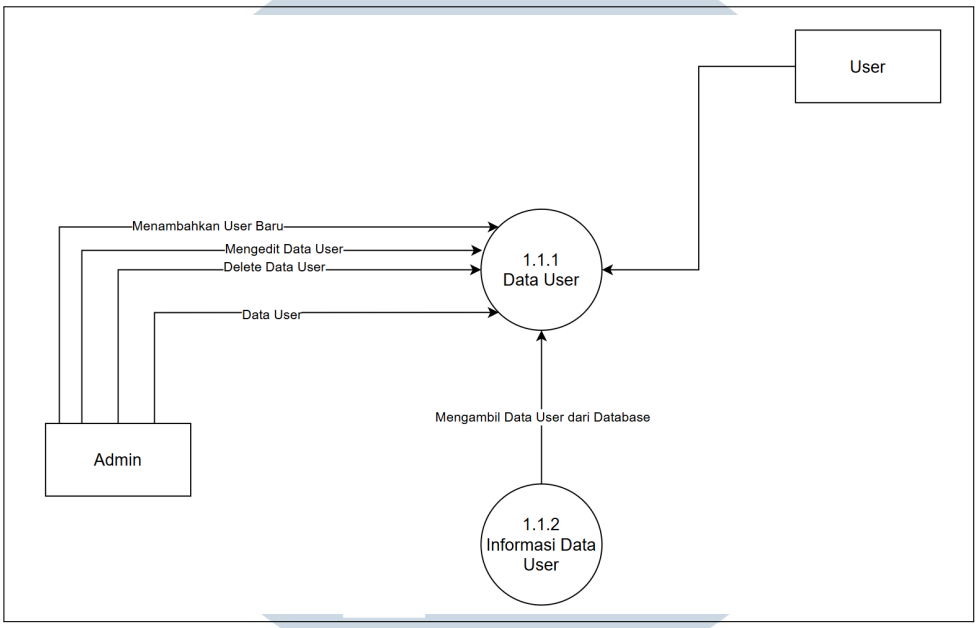


Gambar 3.3. *Data Flow Diagram* Level 1 Kelola Data

C *Data Flow Diagram* level 2

Pada Gambar 3.4, terlihat *Data Flow Diagram* (DFD) Level 2 untuk proses pengelolaan data *User*. Proses ini memungkinkan Admin untuk melakukan operasi seperti menambah, mengedit, dan menghapus data *User* yang ada dalam sistem.

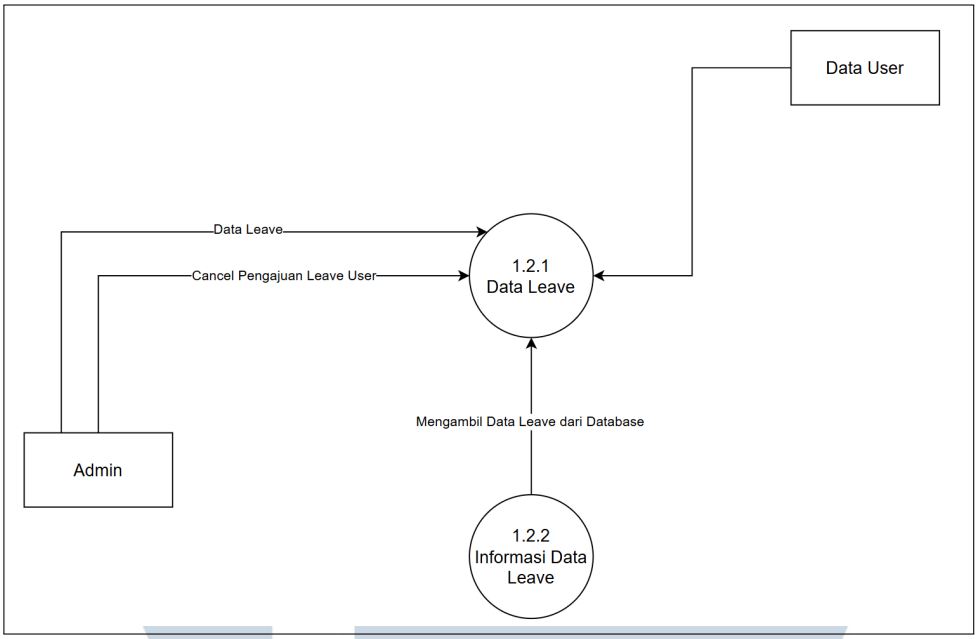
Admin juga dapat memberikan informasi terkait data *User* sesuai dengan kebutuhan organisasi. Proses ini terhubung dengan entitas *User*, yang dapat melihat data mereka melalui sistem.



Gambar 3.4. *Data Flow Diagram* level 2 admin mengelola data *User*

Gambar 3.5 menunjukkan *Data Flow Diagram* (DFD) Level 2 untuk proses pengelolaan data *leave*. Proses ini memungkinkan Admin untuk mengelola data *leave User*. Admin dapat melihat, memperbarui, atau membatalkan pengajuan *leave* yang telah diajukan oleh *User*. Selain itu, *User* juga dapat mengakses informasi mengenai data *leave* mereka, seperti status pengajuan atau *remaining leave balance* yang tersedia. Proses ini mengelola aliran data yang melibatkan Data *Leave* dan Informasi Data *Leave*.

UNIVERSITAS MULTIMEDIA NUSANTARA



Gambar 3.5. *Data Flow Diagram* Level 2 admin mengelola data *Leave*

3.4.2 *Flowchart*

A *Flowchart* Admin Mengelola *User*

Gambar 3.6 menggambarkan serangkaian langkah yang terlibat dalam proses pengelolaan data *User* dalam sistem pengajuan *leave* dan data *User*. Proses dimulai dengan penampilan daftar *User* yang tersedia dalam sistem, yang diambil melalui proses *GET User Data*. Setelah data ditampilkan, Admin memiliki opsi untuk melakukan pencarian atau penyortiran berdasarkan kriteria tertentu. Jika Admin memilih untuk mencari *User*, sistem akan melakukan pencarian sesuai input dan menampilkan hasilnya jika ditemukan.

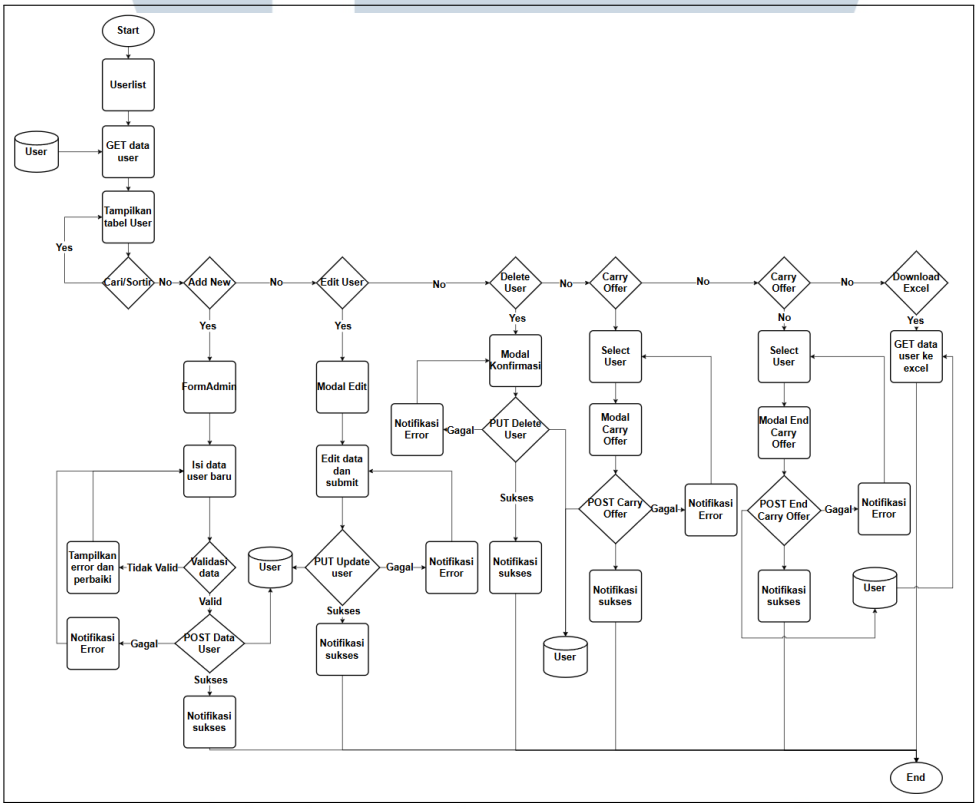
Admin juga dapat menambahkan *User* baru melalui pengisian *FormAdmin*, yang kemudian divalidasi oleh sistem. Jika data yang diisi valid, sistem akan menambahkan *User* ke dalam basis data menggunakan *POST New User*. Namun, jika validasi gagal, sistem akan menampilkan notifikasi kesalahan dan meminta Admin untuk memperbaiki data yang tidak sesuai.

Fitur pengeditan memungkinkan Admin untuk memilih *User* yang ingin diperbarui, membuka *Modal Edit*, dan mengubah data yang diperlukan. Setelah itu, sistem akan melakukan validasi ulang. Jika data valid, sistem akan memperbarui informasi melalui *PUT Update User*. Sebaliknya, jika proses pembaruan gagal, sistem akan memberikan notifikasi kesalahan.

Dalam hal penghapusan data, Admin harus mengonfirmasi terlebih dahulu sebelum sistem mengeksekusi proses penghapusan. Jika proses berjalan dengan baik, sistem akan menampilkan notifikasi sukses. Jika terjadi kegagalan, sistem akan memberikan notifikasi kesalahan.

Admin juga memiliki opsi untuk memberikan penawaran atau tugas tertentu kepada *User* melalui fitur *Carry Offer*. Setelah memilih *User* yang dituju, sistem akan memproses permintaan dan memberikan umpan balik berupa notifikasi keberhasilan atau notifikasi kesalahan tergantung hasilnya.

Selain itu, tersedia pula fitur untuk mengunduh data *User* dalam format Excel. Jika proses pengunduhan berhasil, sistem akan menyediakan file yang dapat diunduh oleh Admin. Jika proses tersebut gagal, sistem akan memberikan notifikasi kesalahan.



Gambar 3.6. *Flowchart* Admin dalam Pengelolaan Data *user*

MULTIMEDIA NUSANTARA

B *Flowchart* Admin Mengelola *Leave*

Pada Gambar 3.7, proses dimulai dengan Admin yang mengambil data riwayat *leave* yang ada dalam sistem melalui proses *GET User Leave Data*. Setelah

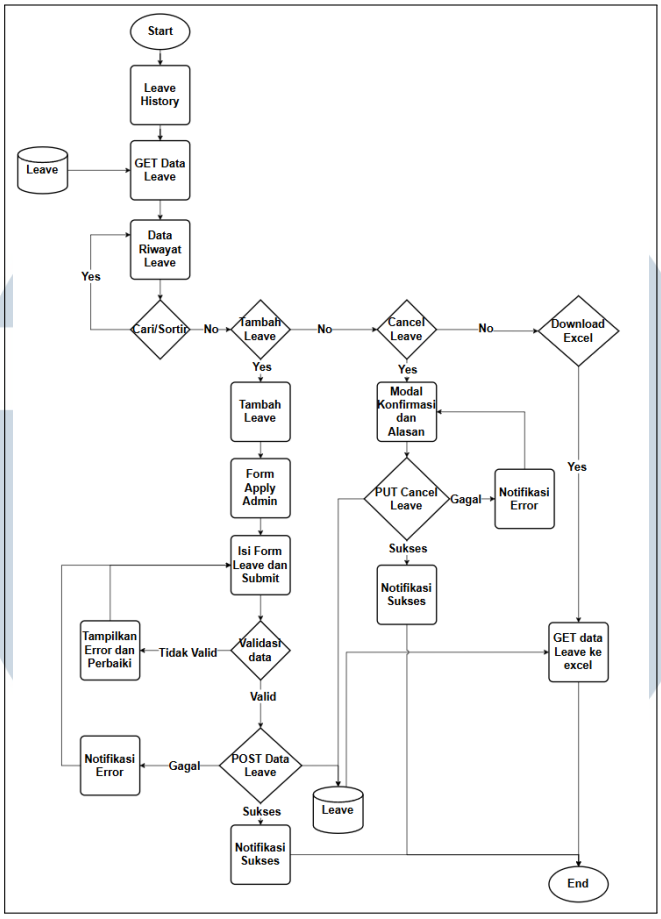
data ditampilkan, Admin memiliki beberapa opsi untuk melanjutkan pengelolaan data tersebut. Admin dapat memilih untuk mencari atau menyortir data *leave* sesuai dengan kriteria yang diinginkan.

Jika Admin memilih untuk menambahkan data *leave*, sistem akan menyediakan *Form Apply Admin* untuk diisi dengan informasi terkait pengajuan *leave*. Setelah form diisi, sistem akan melakukan validasi data untuk memastikan bahwa informasi yang diberikan valid dan sesuai dengan format yang telah ditentukan. Jika data yang dimasukkan valid, sistem akan melakukan *POST New Leave* untuk menyimpan permohonan ke dalam sistem dan menampilkan notifikasi sukses. Jika data tidak valid, sistem akan menampilkan notifikasi kesalahan dan meminta Admin untuk memperbaiki data yang tidak sesuai.

Admin juga memiliki opsi untuk membatalkan *leave* yang telah diajukan oleh *User*. Jika Admin memilih opsi ini, sistem akan menampilkan *Modal Confirmation and Reason*, yang meminta Admin memberikan konfirmasi serta alasan pembatalan. Setelah konfirmasi diberikan, sistem akan menjalankan proses *PUT Cancel Leave* untuk menghapus data yang telah dibatalkan. Jika pembatalan berhasil, sistem akan menampilkan notifikasi keberhasilan. Jika terjadi kesalahan, sistem akan menampilkan notifikasi kesalahan.

Selain itu, Admin dapat mengunduh data *leave* dalam format Excel. Jika proses pengunduhan berhasil, sistem akan menyediakan file Excel yang berisi data *leave* untuk diunduh. Jika pengunduhan gagal, sistem akan memberikan notifikasi kesalahan.

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.7. *Flowchart* Admin Mengelola *Leave*

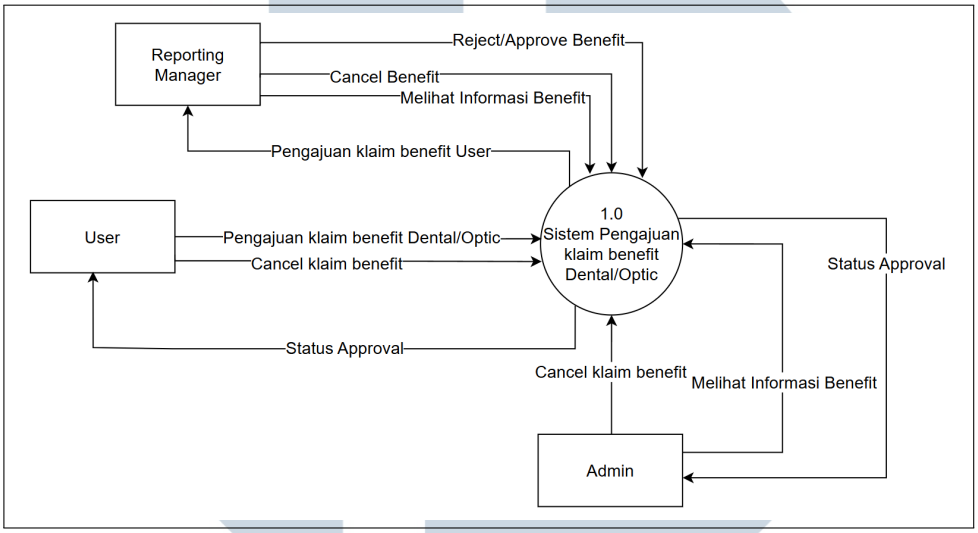
3.4.3 *Data Flow Diagram Benefit*

A *Context Diagram Benefit*

Gambar 3.8 menggambarkan *Context Diagram* untuk pengelolaan klaim *benefit* Dental dan Optical. Diagram ini menunjukkan alur interaksi antara *User*, Admin, *Reporting Manager*, dan Sistem Pengajuan Klaim *Benefit*. Proses dimulai ketika *User* mengajukan klaim dengan memilih jenis klaim yang sesuai, yaitu Dental atau Optical. Setelah klaim diajukan, *User* juga dapat membatalkan klaim yang telah dikirimkan melalui fitur *Cancel Klaim Benefit*, yang kemudian akan diproses oleh sistem.

Sistem Pengajuan Klaim *Benefit* bertanggung jawab untuk memproses klaim yang diajukan oleh *User* serta mengelola status klaim melalui fitur *Status Approval*. Setelah klaim diterima, *Reporting Manager* akan memeriksa dan memutuskan apakah klaim akan diterima atau ditolak melalui opsi *Reject/Approve Benefit*. Selain

itu, *Reporting Manager* memiliki akses untuk melihat informasi klaim yang masuk melalui fitur *View Benefit Information*. Admin juga terlibat dalam pengelolaan klaim, dengan wewenang untuk melihat informasi klaim serta membatalkan klaim yang diajukan oleh *User* jika dibutuhkan.

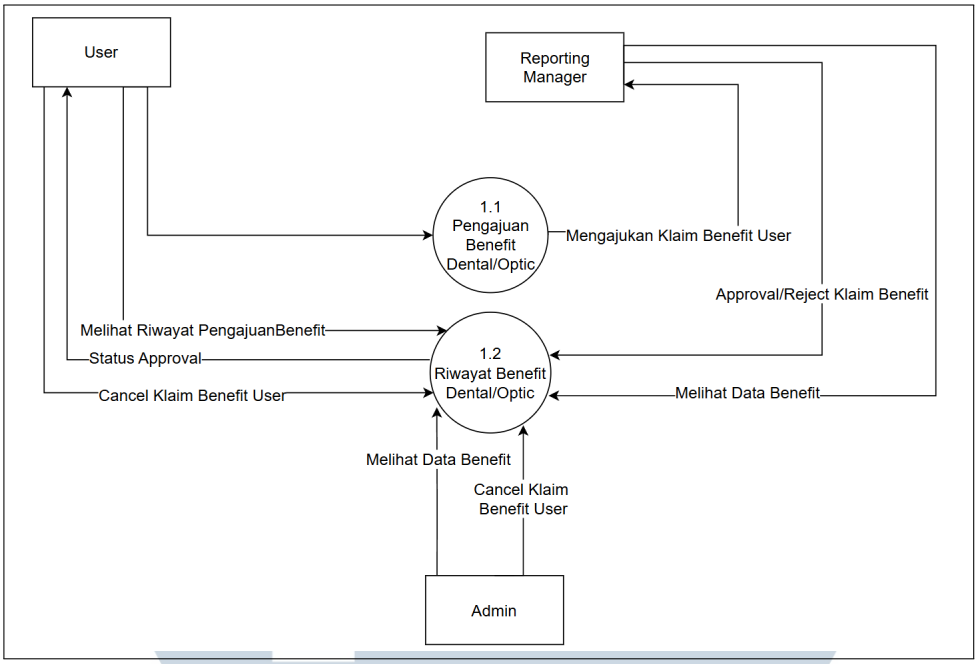


Gambar 3.8. *Context Diagram* untuk Pengelolaan *Benefit* oleh *User*

B *Data Flow Diagram* Level 1

DFD Level 1 menggambarkan alur keseluruhan proses pengajuan klaim *benefit* Dental dan Optical. Diagram ini menunjukkan interaksi antara *User*, Sistem Pengajuan Klaim *Benefit*, dan *Reporting Manager*, serta bagaimana klaim diproses mulai dari pengajuan, pengecekan status, hingga persetujuan atau pembatalan klaim.

Dalam alur yang digambarkan pada Gambar 3.9, *User* dapat mengajukan klaim, melihat status klaim, dan membatalkan klaim yang telah diajukan. Klaim tersebut kemudian diproses oleh sistem dan dikirimkan ke *Reporting Manager* untuk ditinjau dan diputuskan apakah akan disetujui atau ditolak. Admin juga memiliki peran dalam memantau dan membatalkan klaim sesuai kebutuhan operasional.

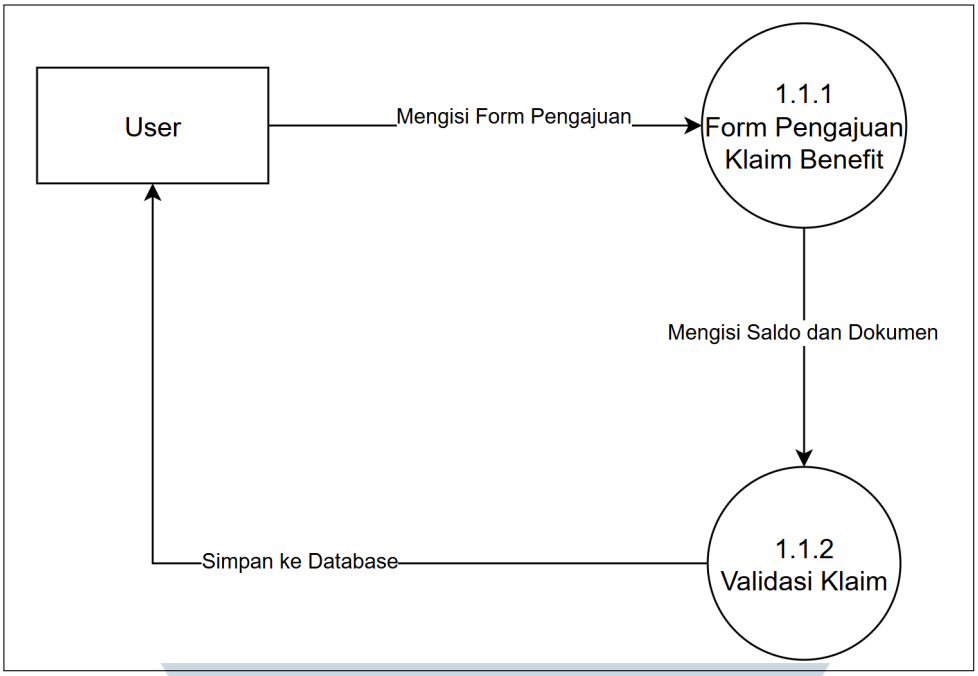


Gambar 3.9. *Data Flow Diagram* Level 1 untuk Pengajuan dan riwayat *Benefit*

C *Data Flow Diagram* Level 2

Gambar 3.10 menggambarkan secara rinci proses pengajuan klaim *benefit* Dental dan Optical yang dilakukan oleh *User* serta bagaimana sistem memproses klaim tersebut. Proses dimulai ketika *User* mengisi *form* pengajuan klaim. Setelah memilih jenis klaim, *User* melengkapi *form* dengan informasi yang diperlukan, seperti sisa saldo dan dokumen pendukung. Data tersebut kemudian disimpan ke dalam *Database*. Setelah itu, sistem memproses data tersebut pada tahap Validasi Klaim, yang berfungsi untuk memeriksa kelengkapan dan keabsahan data yang dikirimkan.

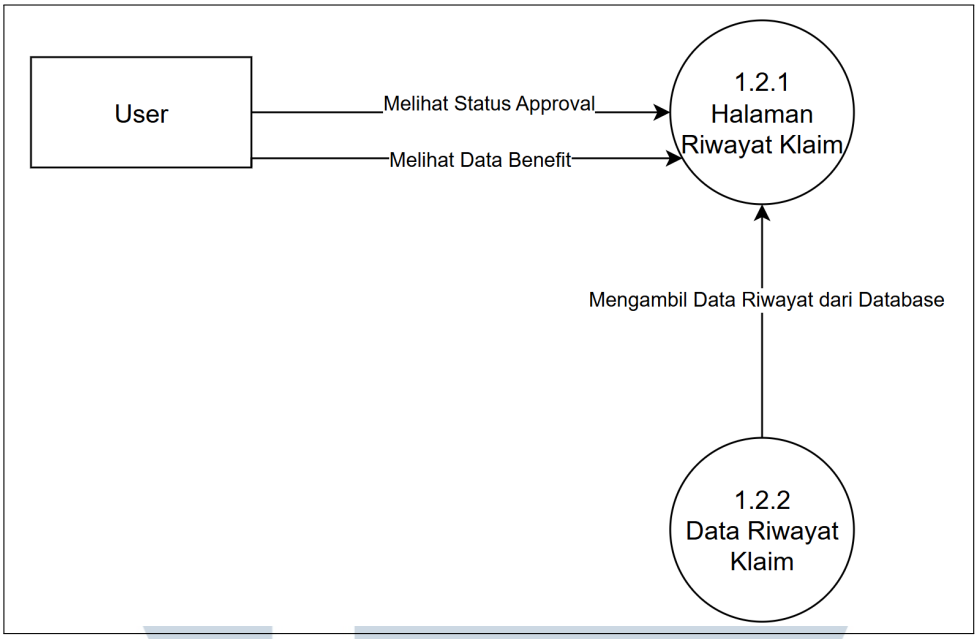
UNIVERSITAS MULTIMEDIA NUSANTARA



Gambar 3.10. *Data Flow Diagram* Level 2 Form Pengajuan *Benefit*

Jika klaim yang diajukan dinyatakan valid, proses berlanjut ke tahap penyimpanan klaim dalam *Database*. Setelah klaim berhasil disimpan, *User* dapat mengakses halaman Riwayat Klaim sebagaimana ditunjukkan pada Gambar 3.11. Pada tahap ini, *User* dapat mengambil data riwayat klaim dari *Database* dan melihat detail serta status persetujuan klaim yang telah diajukan.

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.11. *Data Flow Diagram* Level 2: Riwayat Pengajuan *Benefit*

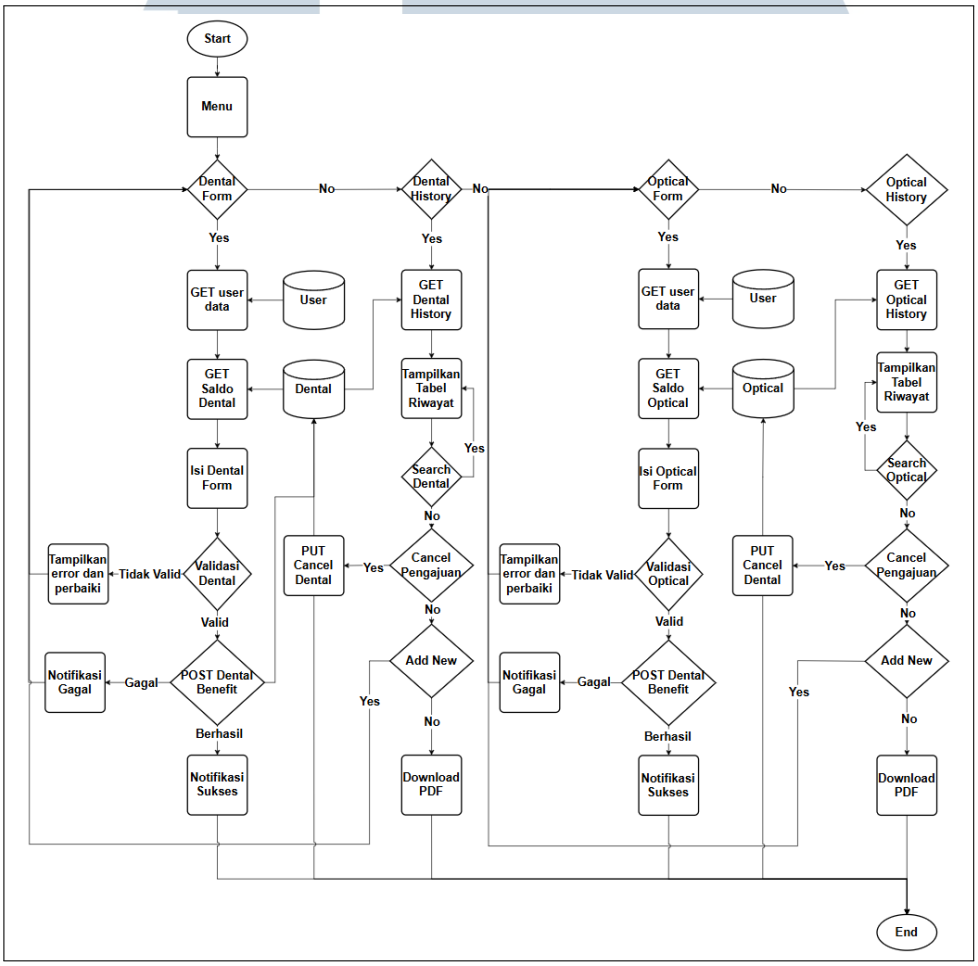
3.4.4 *Flowchart Benefit*

Gambar 3.12 menggambarkan alur proses pengajuan *benefit* untuk layanan Dental dan Optical. Proses ini melibatkan beberapa tahapan penting seperti pengecekan saldo, pengisian *form*, validasi data, serta pengajuan *benefit* ke sistem. Alur dimulai dengan *User* memilih menu utama antara *Dental Form* atau *Optical Form*.

Jika *User* memilih *Dental Form*, langkah pertama adalah mengambil data *User* melalui *GET User Data* untuk memverifikasi status. Selanjutnya, sistem melakukan pengecekan saldo menggunakan *GET Saldo Dental*. Jika saldo mencukupi, *User* akan diarahkan untuk mengisi *form Dental*. Setelah form diisi, sistem memvalidasi data yang telah dimasukkan. Jika data tidak valid, sistem menampilkan notifikasi kesalahan dan meminta *User* memperbaiki informasi tersebut. Jika valid, sistem melanjutkan proses pengajuan menggunakan *POST Dental Benefit*. Jika berhasil, akan muncul notifikasi sukses, dan jika gagal, akan ditampilkan notifikasi gagal. *User* juga dapat membatalkan pengajuan melalui *PUT Cancel Dental*, yang akan menghentikan seluruh proses. Selain itu, *User* dapat melihat *Tabel Riwayat* pengajuan sebelumnya dan mengunduh laporan dalam bentuk PDF.

Untuk *Optical Form*, alur proses serupa dengan *Dental Form*. *User* memulai

dengan mengecek saldo melalui *GET Saldo Optical*, lalu mengisi *form Optical*. Setelah pengisian, sistem memvalidasi data. Jika terdapat kesalahan, sistem akan menampilkan pesan error. Jika valid, pengajuan diproses melalui *POST Optical Benefit*. Jika berhasil, sistem memberikan notifikasi sukses. Jika gagal, ditampilkan notifikasi kegagalan. *User* juga dapat membatalkan pengajuan melalui *PUT Cancel Optical*, serta melihat riwayat dan mengunduh laporan dalam format PDF.



Gambar 3.12. *Flowchart user* dalam Pengajuan *Benefit*

3.5 Pengembangan

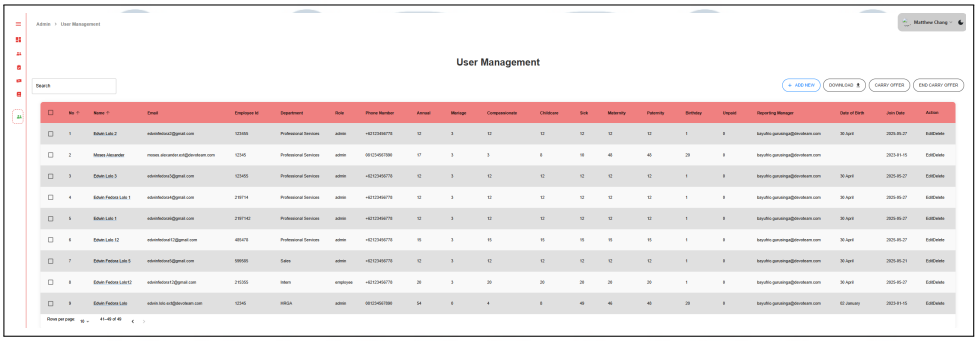
Pada implementasi *frontend* sistem HRIS ini, digunakan *React.js* sebagai *framework* utama dalam pengembangan tampilan *UI*. *React.js* dipilih karena kemampuannya dalam membangun aplikasi web yang cepat, responsif, dan efisien, dengan memanfaatkan konsep *virtual DOM*. Konsep ini memungkinkan pembaruan tampilan aplikasi tanpa perlu memuat ulang seluruh halaman, sehingga

meningkatkan performa dan memberikan pengalaman *User* yang lebih baik, terutama pada aplikasi yang membutuhkan pembaruan data secara *real-time*.

Sistem ini mengadopsi pendekatan *Single Page Application* (*SPA*), di mana bagian *frontend* dan *backend* dipisahkan ke dalam dua *codebase* yang terpisah dan saling berkomunikasi melalui *API*. Pendekatan *SPA* ini memungkinkan aplikasi memberikan pengalaman *User* yang lebih interaktif dan responsif karena hanya bagian tertentu dari halaman yang diperbarui. Selain itu, metode ini juga mempermudah proses pengelolaan aplikasi dan meminimalkan waktu tunggu saat *User* berpindah antar halaman.

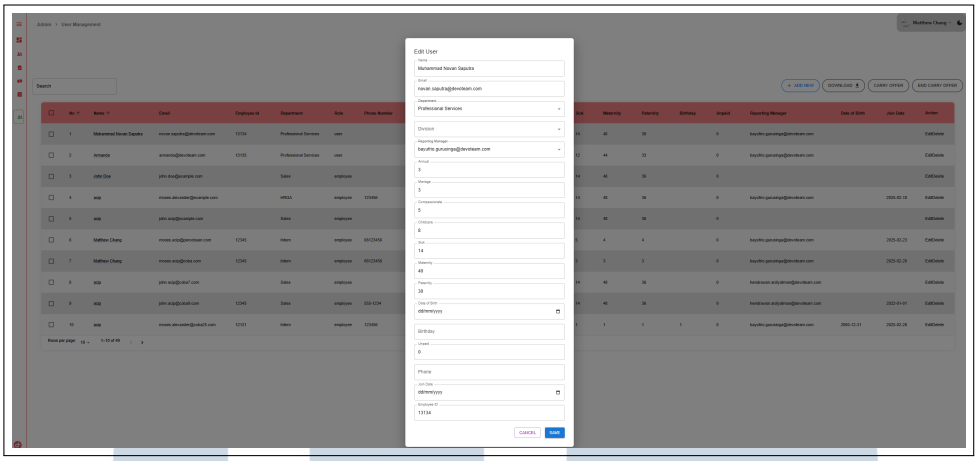
3.5.1 Halaman *UserList*

Halaman *UserList* merupakan pusat manajemen data *User* dalam aplikasi HRIS. Pada halaman ini, Admin dapat melihat seluruh daftar *User* beserta informasi detail seperti nama, email, ID *User*, departemen, peran, nomor telepon, serta hak *leave* yang dimiliki (seperti *annual*, *marriage*, *sick*, dan lain-lain). Halaman ini juga dilengkapi dengan berbagai fitur yang memudahkan Admin dalam mengelola data *User* secara efisien dan terintegrasi dengan *backend*. Jika tombol *Add New* ditekan, Admin akan diarahkan ke halaman *Form Admin* seperti yang ditampilkan pada Gambar 3.18.



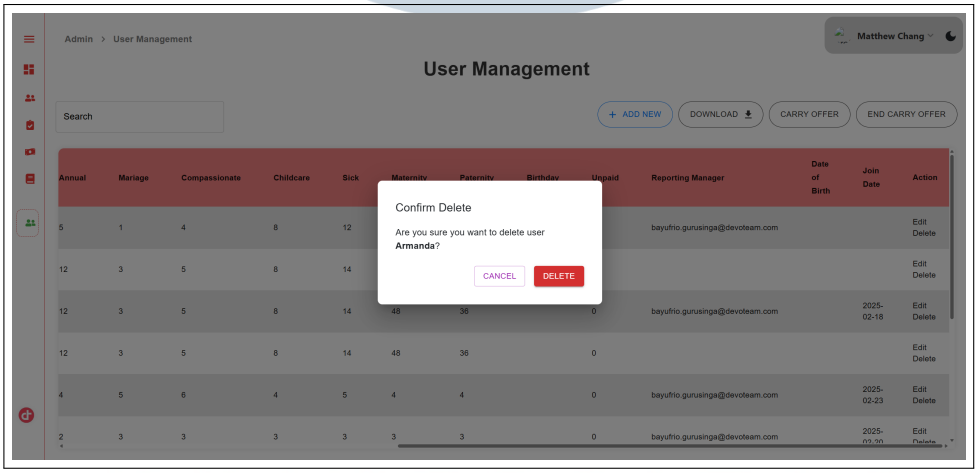
Gambar 3.13. Halaman *UserList*

Fitur *Edit* digunakan untuk memperbarui data *User* yang sudah ada. Admin dapat mengklik tombol *Edit* pada baris *User* tertentu, lalu *modal edit* akan muncul. Admin dapat mengubah informasi dalam modal tersebut dan menekan tombol *Save*. Data yang telah diperbarui akan dikirim ke *backend* menggunakan metode *PUT*, dan tampilan *UserList* akan diperbarui dengan informasi terbaru.



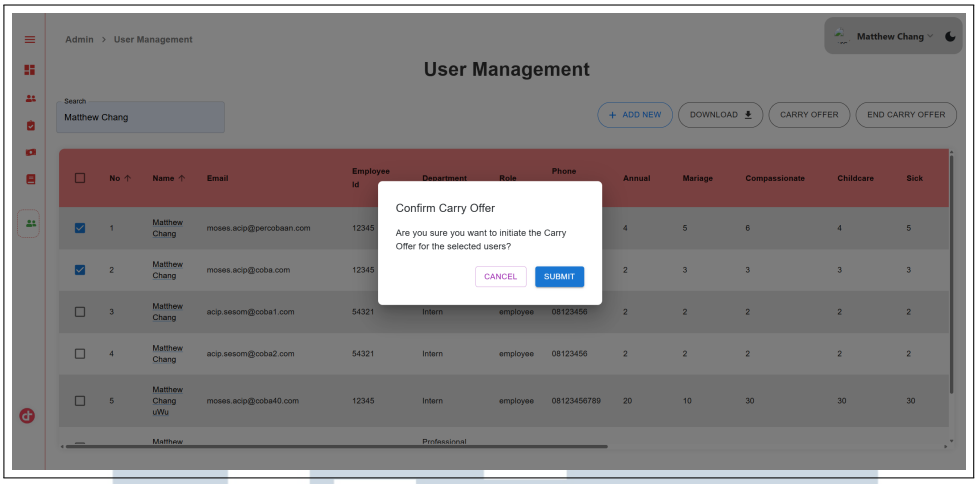
Gambar 3.14. *Modal Edit* pada Halaman *UserList*

Fitur *Delete* berfungsi untuk menghapus *User* dari sistem. Admin dapat mengklik tombol *Delete* pada baris *User* yang diinginkan. Setelah itu, sistem akan menampilkan *modal konfirmasi*. Jika Admin mengonfirmasi penghapusan, permintaan akan dikirim ke *backend* menggunakan metode *PUT* ke endpoint *delete*, dan *User* akan dihapus dari tabel *UserList*.



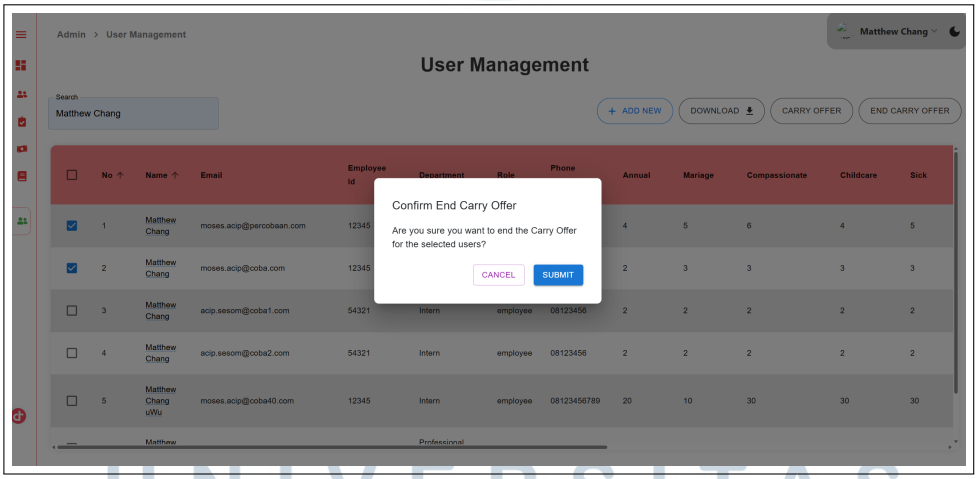
Gambar 3.15. *Modal Konfirmasi Penghapusan User*

Fitur *Carry Offer* memungkinkan Admin untuk memindahkan sisa *leave* tahun sebelumnya ke tahun berjalan bagi *User* yang dipilih. Admin dapat memilih satu atau beberapa *User* melalui *checkbox*, kemudian menekan tombol *Carry Offer*. Setelah muncul konfirmasi dalam *modal*, permintaan akan dikirim ke *backend* menggunakan metode *POST*, dan data *leave User* akan diperbarui sesuai hasil permintaan tersebut.



Gambar 3.16. *Modal Carry Offer* pada Halaman *UserList*

Fitur *End Carry Offer* digunakan untuk mengakhiri periode *carry offer leave* bagi *User* yang dipilih. Setelah memilih *User*, Admin dapat menekan tombol *End Carry Offer*. Sistem kemudian akan menampilkan *modal konfirmasi*. Jika dikonfirmasi, permintaan akan dikirim ke *backend* menggunakan metode *POST*, dan proses *carry offer* akan dihentikan untuk *User* tersebut.

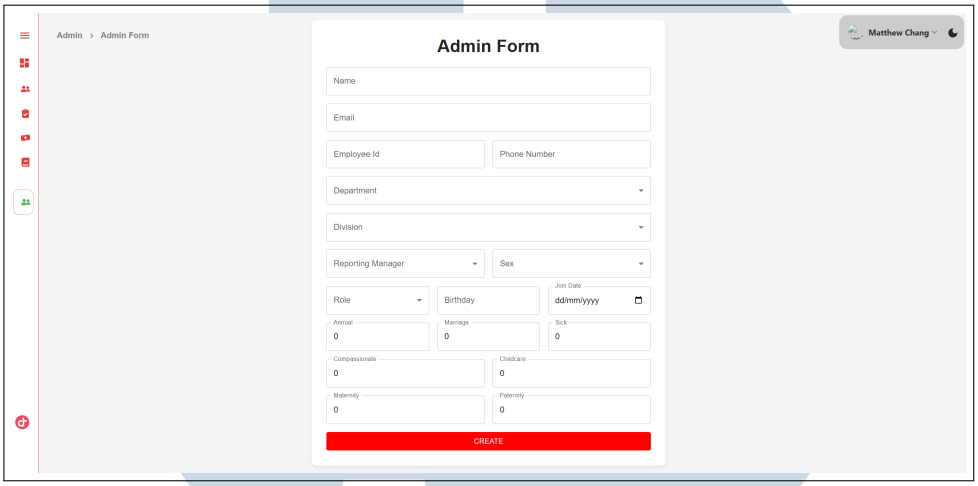


Gambar 3.17. *Modal End Carry Offer* pada Halaman *UserList*

3.5.2 Halaman *Form Admin*

Admin dapat menambahkan data *User* baru ke dalam sistem HRIS melalui halaman *Form Admin* seperti yang ditunjukkan pada Gambar 3.18. *Form* ini mencakup input untuk nama, email, ID *User*, nomor telepon, departemen, divisi, *reporting manager*, jenis kelamin, peran, tanggal lahir, tanggal masuk, serta kuota

leave untuk berbagai jenis cuti. Setelah *form* diisi dan divalidasi, data akan dikirim ke *backend* melalui *API* dan disimpan ke dalam *database*. Jika proses berhasil, sistem akan menampilkan notifikasi sukses dan *form* akan direset secara otomatis. Jika terjadi kesalahan, sistem akan menampilkan notifikasi kesalahan.

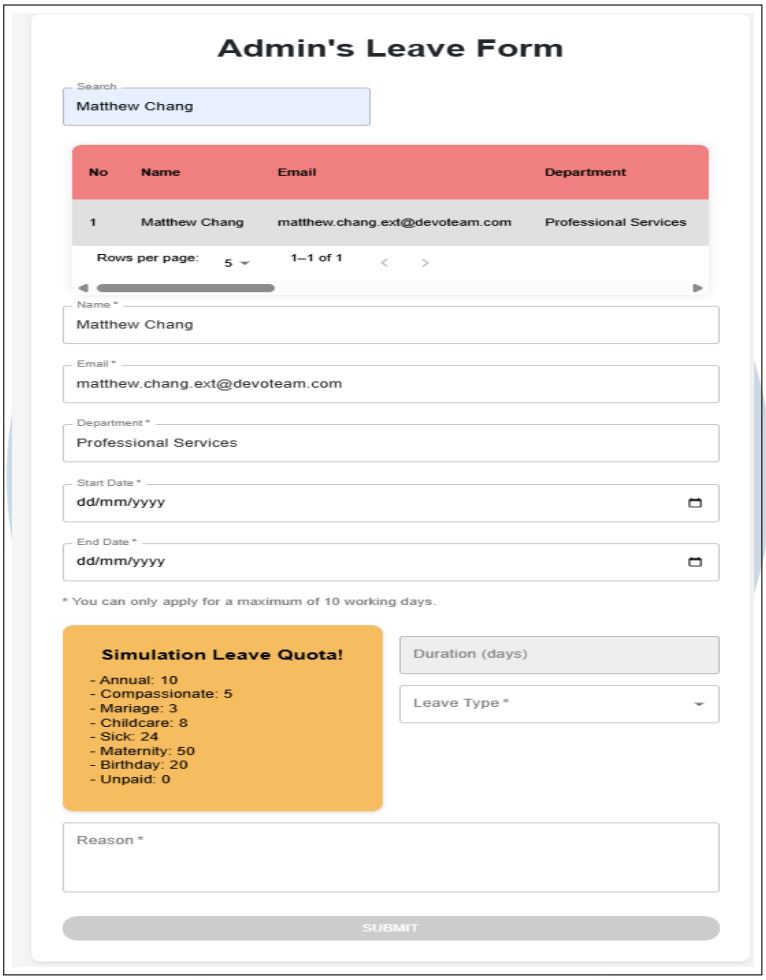


Gambar 3.18. *Form Admin* untuk Penambahan Data *User*

3.5.3 *Form Annual Leave* oleh Admin

Halaman *Form Annual Leave Admin* berfungsi untuk membantu *User* yang lupa mengajukan *leave*. Dalam kondisi seperti ini, Admin memiliki peran untuk mengajukan *leave* atas nama *User*. Karena sistem tidak mengizinkan pengajuan *leave* untuk tanggal yang sudah lewat, Admin diberikan akses untuk mengajukan baik pada tanggal yang telah lewat maupun yang akan datang. Ilustrasi halaman ini dapat dilihat pada Gambar 3.19.

UNIVERSITAS MULTIMEDIA NUSANTARA

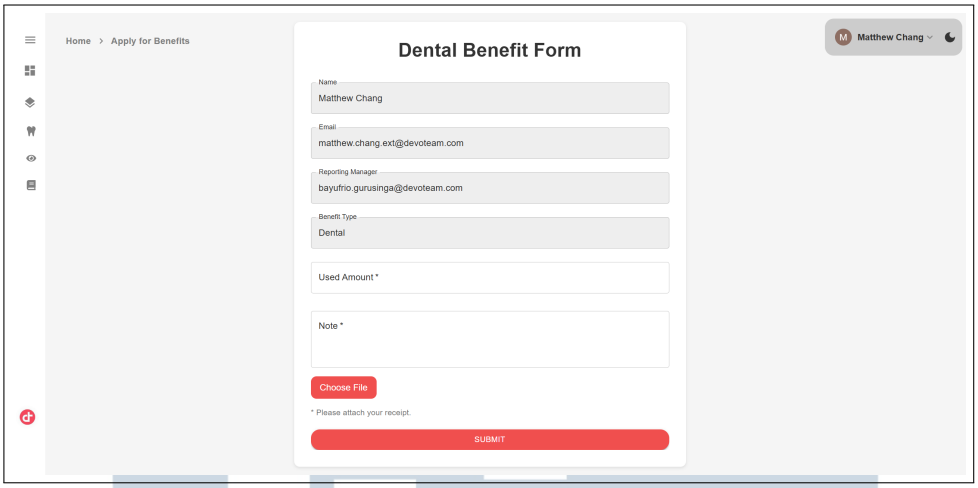


Gambar 3.19. *Form Annual Leave* oleh Admin

3.5.4 Halaman *Dental Benefit*

Halaman *DentalForm* digunakan oleh *User* atau Admin untuk mengajukan klaim *benefit* jenis dental. Pada halaman ini, *User* dapat mengisi data pengajuan, mengunggah bukti klaim berupa struk atau kwitansi, serta mengirim permohonan klaim ke sistem.

UNIVERSITAS MULTIMEDIA NUSANTARA

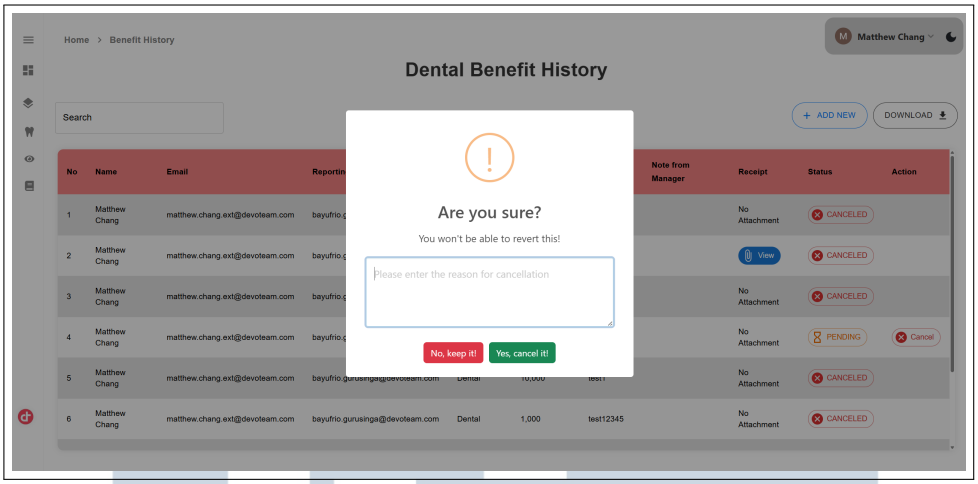


Gambar 3.20. Halaman *DentalForm* untuk Pengajuan *Benefit*

Halaman *DentalHistory* digunakan untuk melihat riwayat pengajuan klaim *dental benefit* oleh *User*. Di halaman ini, *User* dapat memantau status pengajuan, mengunduh riwayat klaim, dan membatalkan klaim apabila diperlukan.

Gambar 3.21. Halaman *DentalHistory* untuk Riwayat Klaim Dental

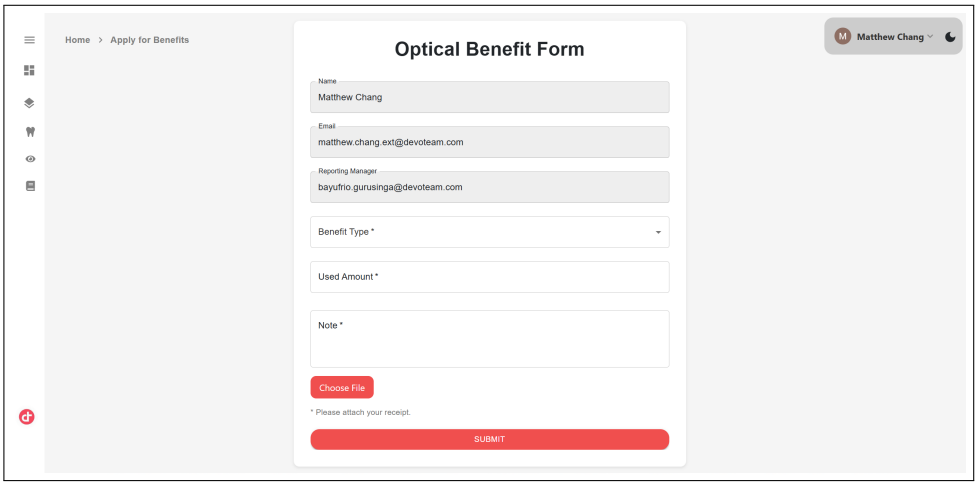
Jika status pengajuan masih dalam kondisi *Pending* atau *Approved*, *User* memiliki opsi untuk membatalkan klaim dengan mengklik tombol *Cancel*. Setelah itu, sistem akan meminta alasan pembatalan, kemudian mengirim permintaan ke *backend* menggunakan metode *PUT API*. Status pengajuan akan diperbarui menjadi *Cancelled*.



Gambar 3.22. *Modal Cancel* Pengajuan Dental oleh *User*

3.5.5 Halaman *Optical Benefit*

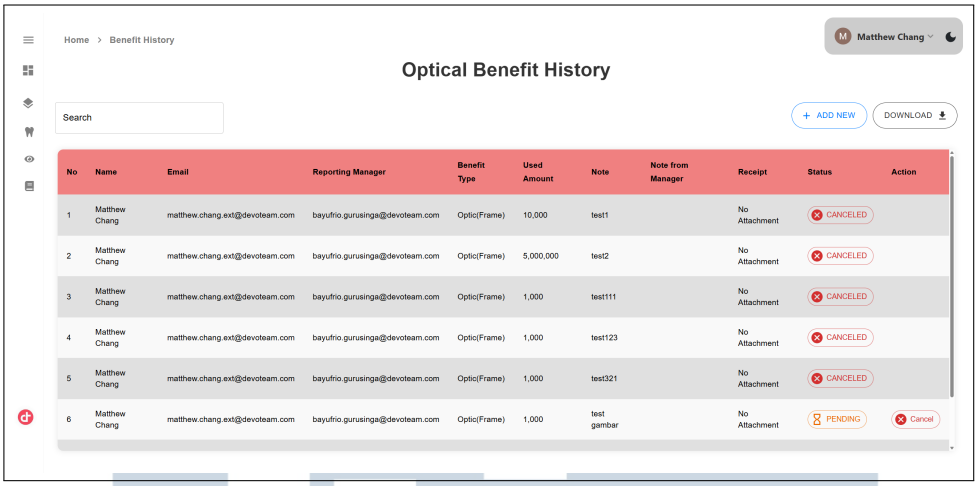
Halaman *OpticalForm* digunakan oleh *User* atau Admin untuk mengajukan klaim *benefit* jenis optical, baik untuk *frame* maupun *lense*. Pada halaman ini, *User* dapat mengisi data pengajuan, memilih tipe *benefit*, mengunggah bukti pembayaran, serta mengirimkan permohonan klaim ke sistem.



Gambar 3.23. Halaman *OpticalForm* untuk Pengajuan *Benefit*

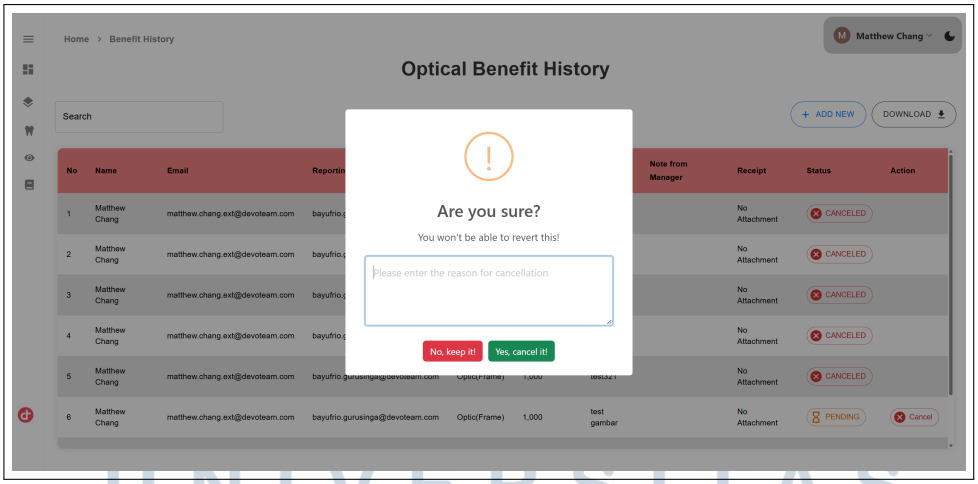
Halaman *OpticalHistory* digunakan untuk menampilkan riwayat pengajuan klaim *optical benefit*. Di halaman ini, *User* dapat memantau status pengajuan, mengunduh riwayat dalam format tertentu, dan membatalkan klaim apabila masih memungkinkan.

MULTIMEDIA NUSANTARA



Gambar 3.24. Halaman *OpticalHistory* untuk Riwayat Klaim Optical

Sama seperti pada klaim dental, jika status klaim masih berada pada tahap *Pending* atau *Approved*, *User* dapat melakukan pembatalan dengan menekan tombol *Cancel*. Sistem akan meminta alasan pembatalan, kemudian mengirimkan permintaan ke *backend* melalui metode *PUT API*. Setelah itu, status klaim akan diperbarui menjadi *Cancelled*.



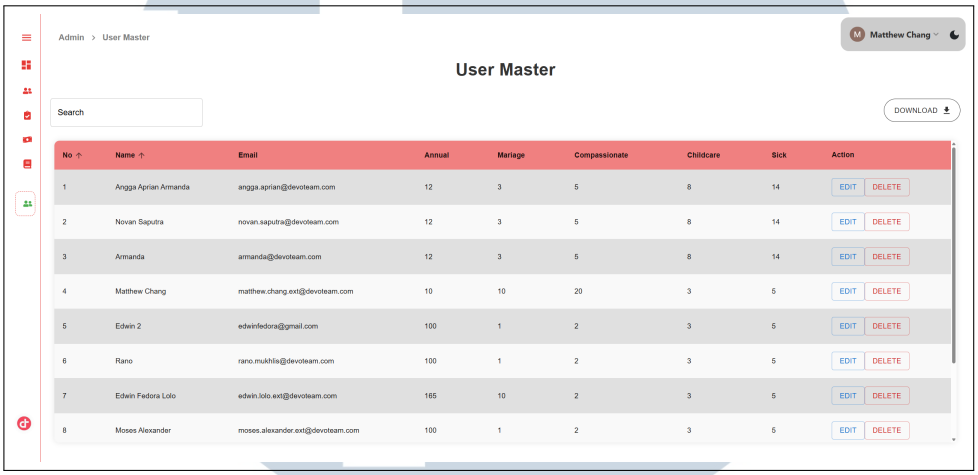
Gambar 3.25. *Modal Cancel* Pengajuan Optical oleh *User*

3.5.6 Fitur lain yang dibuat

A Halaman *UserMaster*

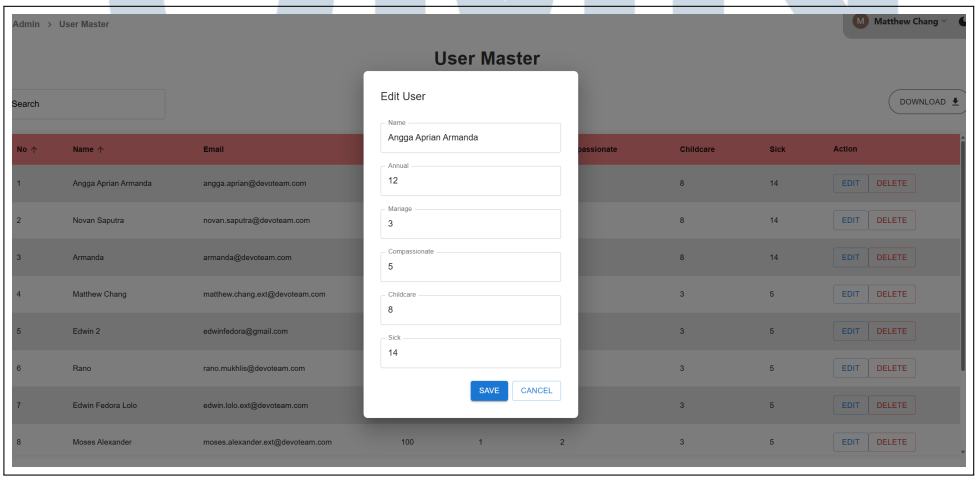
Halaman *UserMaster* berfungsi sebagai pusat pengelolaan data induk *User*, khususnya untuk mengatur dan memantau hak *leave User* secara detail. Di

halaman ini, Admin dapat melihat daftar *User* beserta rincian hak *leave* mereka seperti *annual*, *marriage*, *compassionate*, *childcare*, dan *sick*. Fitur pencarian dan pengurutan disediakan untuk memudahkan navigasi data. Selain itu, Admin dapat melakukan aksi edit dan hapus data *User*. Halaman ini juga menyediakan fitur untuk mengunduh data *User* dalam format Excel guna keperluan pelaporan atau pencadangan data.



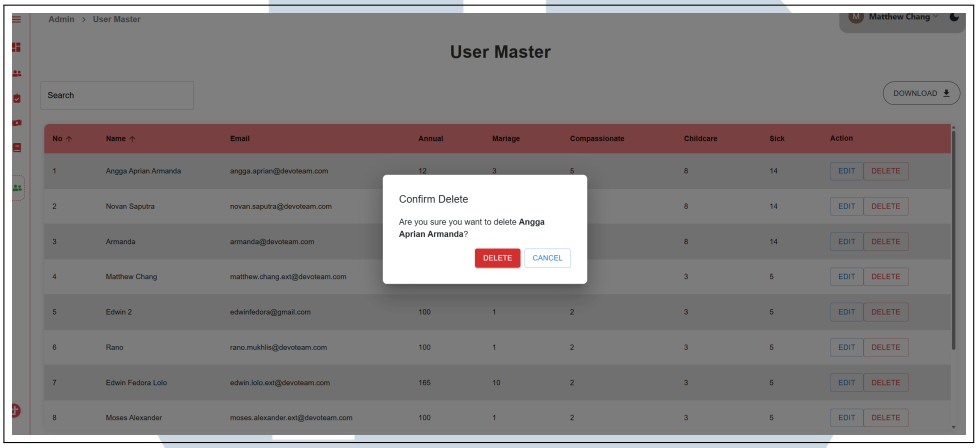
Gambar 3.26. Halaman *UserMaster* untuk Monitoring Hak *Leave*

Admin dapat mengedit hak *leave User* dengan menekan tombol *Edit* pada baris yang diinginkan. Setelah itu, *modal edit* akan muncul dan Admin dapat memperbarui nilai hak *leave* (seperti *annual*, *marriage*, dan lainnya), lalu menekan tombol *Save*. Data yang diperbarui akan dikirim ke *backend* dan langsung diterapkan dalam sistem.



Gambar 3.27. *Modal Edit* pada Halaman *UserMaster*

Fitur *Delete* digunakan untuk menghapus *User* dari data induk. Admin dapat mengklik tombol *Delete* pada baris *User* yang ingin dihapus. Setelah muncul *modal konfirmasi*, Admin dapat mengonfirmasi penghapusan. Jika disetujui, data *User* akan dihapus melalui proses yang dikirim ke *backend*.



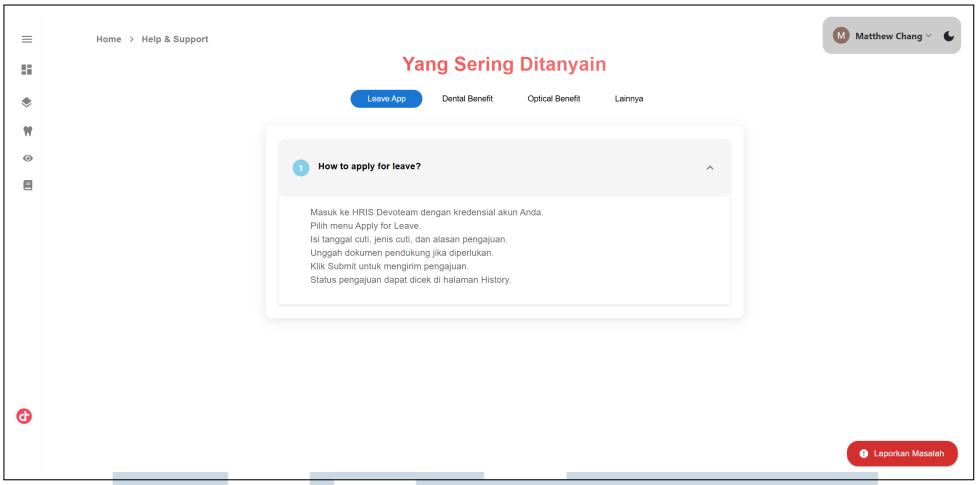
Gambar 3.28. *Modal Konfirmasi Hapus* pada Halaman *UserMaster*

B Halaman *Help and Support*

Halaman *Help and Support* berfungsi sebagai pusat bantuan bagi *User* atau Admin yang membutuhkan informasi atau ingin melaporkan masalah terkait aplikasi HRIS. Pada halaman ini, *User* dapat melihat daftar pertanyaan yang sering diajukan (*FAQ*) yang dikelompokkan berdasarkan kategori seperti *Leave App*, *Dental Benefit*, *Optical Benefit*, dan kategori lainnya.

Data *FAQ* diambil secara langsung dari *backend* menggunakan *endpoint* `GET /faq`. Dengan demikian, setiap kali halaman dibuka, aplikasi akan mengirimkan permintaan *GET* ke *backend* untuk memperoleh daftar *FAQ* terbaru yang tersimpan dalam sistem.

UNIVERSITAS MULTIMEDIA NUSANTARA



Gambar 3.29. Halaman *Help and Support* pada Aplikasi

3.6 Kendala dan Solusi yang Ditemukan

3.6.1 Kendala

Proses pengembangan aplikasi HRIS selama kegiatan magang dihadapkan pada beberapa tantangan, baik yang bersifat teknis maupun non-teknis. Berikut adalah kendala-kendala utama yang ditemui:

1. Terlibat dalam beberapa proyek secara bersamaan menjadi tantangan dalam hal manajemen waktu dan penentuan prioritas untuk memastikan kontribusi yang optimal pada setiap tugas, termasuk pengembangan aplikasi HRIS.
2. Tantangan dalam memastikan sinkronisasi data yang akurat antara antarmuka *frontend* dan sistem *backend*.

3.6.2 Solusi

Untuk mengatasi berbagai kendala yang muncul selama proses magang, dilakukan beberapa langkah penyelesaian sebagai berikut:

1. Manajemen waktu yang lebih efektif diterapkan dengan menetapkan skala prioritas terhadap tugas-tugas yang harus diselesaikan. Selain itu, dilakukan koordinasi yang lebih intensif dengan atasan dan tim proyek guna memastikan bahwa alokasi waktu dapat difokuskan pada pengembangan aplikasi HRIS sebagai tugas utama.

2. Permasalahan pada *backend* segera dianalisis dan diselesaikan. Hasil analisis serta solusi yang diterapkan kemudian dilaporkan kepada senior developer yang bertanggung jawab. Laporan ini mencakup penjelasan perubahan serta identifikasi akar masalah agar dapat ditindaklanjuti lebih lanjut secara struktural.

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA