BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Kegiatan magang dilaksanakan di PT Dover Chemical Indonesia selama satu semester dengan kedudukan sebagai *Mobile App Developer (Full Stack)* pada divisi Pengembangan Sistem.

Selama program magang ini, bimbingan diberikan dari Bapak Albert Oktovianus dan Bapak Rafli Kamandita selaku mentor teknis, serta pengawasan langsung dari Ibu Lenny Indrawati selaku supervisor.

Fokus utama kegiatan magang adalah pada pengembangan aplikasi *Customer Relationship Management* (CRM) yang digunakan untuk proses input *Sales Order (SO)* di perusahaan. Pengembangan dilakukan secara end-to-end, mulai dari sisi *backend* hingga *frontend*, dengan tujuan untuk meningkatkan efisiensi dalam pencatatan dan manajemen data SO.

Dalam pelaksanaan kegiatan, koordinasi tim dilakukan melalui platform *WhatsApp* untuk komunikasi harian serta *Seafile* untuk pelacakan progres pekerjaan dan dokumentasi pengembangan sistem.

3.2 Tugas yang Dilakukan

Selama melaksanakan magang di PT Dover Chemical Indonesia, penulis bertanggung jawab dalam pengembangan aplikasi *Customer Relationship Management* (CRM) yang digunakan untuk keperluan input draft *Sales Order (SO)*.

Aplikasi dikembangkan dengan arsitektur *client-server*, yaitu model komunikasi di mana perangkat pengguna (*client*) berinteraksi dengan sistem pusat (*server*) untuk mengakses, mengolah, dan menyimpan data. Bagian *frontend* dibangun menggunakan bahasa pemrograman *Kotlin* untuk platform *Android*, sedangkan bagian *backend* dikembangkan menggunakan PHP dan REST API. Komunikasi antara kedua sisi sistem dilakukan melalui *library Retrofit* yang berfungsi untuk mengelola permintaan dan respons HTTP secara efisien.

Langkah-langkah pelaksanaan tugas dapat dirinci sebagai berikut:

- Membangun sistem *backend* menggunakan PHP dan REST API untuk menangani proses CRUD (*Create, Read, Update, Delete*) pada data Sales Order.

- Melakukan pengujian endpoint API secara langsung melalui browser pada *localhost* dengan parameter *query* yang sesuai.
- Mengintegrasikan *frontend* dengan *backend* menggunakan Retrofit sebagai alat komunikasi API, dengan penerapan *fallback mechanism* yang dapat beralih ke data lokal apabila koneksi internet tidak tersedia.
- Mengembangkan tampilan dan fitur interaktif pada sisi *frontend* menggunakan Kotlin dengan implementasi *offline-first architecture*.
- Membangun sistem *local database* menggunakan SQLite dan Room untuk menyimpan data produk, user, dan draft sales order secara *offline*.
- Mengimplementasikan mekanisme sinkronisasi otomatis yang dapat mendeteksi koneksi internet dan melakukan *background sync* data lokal ke server.
- Melaksanakan pengujian menyeluruh terhadap aplikasi di akhir tahap pengembangan untuk memastikan fungsionalitas *offline* dan *online* berjalan sesuai dengan kebutuhan bisnis.

3.3 Uraian Pelaksanaan Magang

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Uraian Pekerjaan Setiap Minggu Selama Pelaksanaan Kerja Magang

Minggu Ke-	Uraian Pelaksanaan Kerja Magang
1	Memahami sistem kerja dan teknologi yang digunakan serta memulai pengembangan frontend aplikasi menggunakan Kotlin.
2	Menyelesaikan pengembangan front-end dan memulai pengembangkan backend (autentikasi, reset kata sandi, unggah dokumen) dan menghubungkannya ke MySQL menggunakan Retrofit dan REST API.
3	Meningkatkan keamanan dengan enkripsi MD5, serta membuat fitur edit profil, ubah kata sandi, filter SO, dan pagination.
	Lanjut pada halaman berikutnya

Tabel 3.1: Uraian Pekerjaan Setiap Minggu Selama Pelaksanaan Kerja Magang (lanjutan)

Minggu Ke-	Uraian Pelaksanaan Kerja Magang
4	Mengoptimalkan pengambilan data dan filtering secara real-time serta mengintegrasikan input draft SO (alamat, item, dropdown, mata uang).
5	Membuat REST API untuk item berdasarkan kategori/unit, mengoptimalkan UI fragment, dan menambahkan fitur hapus draft serta caching.
6	Memperkuat koneksi frontend-backend, memvalidasi hak akses, menambahkan log aktivitas, dan menampilkan unit penjualan default.
7	Membuat fitur pajak dan spinner jual/pinjam, memperbaiki bug API customer item, serta mengembangkan input harga dan kuantitas.
8	Membuat perhitungan pajak dan total transaksi dinamis per item serta mengoptimalkan fitur hapus dan binding RecyclerView.
9	Menyelesaikan form submission dan mengirim data ke tabel `t_draft_so` dan `t_detail_draftso`, serta melakukan pengujian API.
10	Membuat fitur upload lampiran ke database, mengembangkan fitur delete dan edit draft SO sesuai mode tambah/edit.
11	Menyempurnakan fitur edit dengan menampilkan data sales, customer, class, dan attachment, serta memperbaiki validasi input.
12	Mengembangkan fitur multi-file attachment (download, hapus, buka), serta melakukan validasi dan update database.
13	Mengimplementasikan mode offline menggunakan Room dan SQLite untuk login, histori transaksi, dan form.
14	Membuat fitur pengiriman data offline, mengatur koneksi dengan NetworkManager, dan menyimpan form sebagai draft.
15	Menambahkan fitur simpan draft dengan Room, perbaikan bug class name, mode edit offline/online, serta caching lampiran dan upload offline.

Lanjut pada halaman berikutnya

Tabel 3.1: Uraian Pekerjaan Setiap Minggu Selama Pelaksanaan Kerja Magang (lanjutan)

Minggu Ke-	Uraian Pelaksanaan Kerja Magang
16	Perbaikan UI dan bug fetch offline, implementasi MVVM login dan settings, serta koneksi submit form offline dan preload data global.

3.3.1 Software dan Hardware yang digunakan

A. Software

Software yang digunakan selama pengembangan aplikasi CRM untuk input draft Sales Order (SO) di PT Dover Chemical meliputi:

1. Android Studio, merupakan Integrated Development Environment (IDE) resmi yang berasal dari Google untuk membangun aplikasi Android. Perangkat lunak ini menyediakan fitur seperti Android Emulator, Layout Editor, serta dukungan bahasa Kotlin dan Java yang mempermudah proses pengembangan aplikasi mobile [6].

2. Visual Studio Code, adalah platform editor kode yang dikembangkan oleh Microsoft. Aplikasi ini mendukung berbagai bahasa pemrograman serta dilengkapi fitur seperti IntelliSense, debugging, dan integrasi Git. Visual Studio Code juga dapat diperluas menggunakan ekstensi sesuai kebutuhan pengembang [7].

3. XAMPP, merupakan paket perangkat lunak yang berisi Apache, MariaDB/MySQL, PHP, dan Perl, digunakan sebagai server lokal [8].

4. Microsoft SQL Server, merupakan sistem manajemen basis data yang bersifat relasional (RDBMS) yang dikembangkan oleh Microsoft. Perangkat lunak ini menawarkan fitur seperti optimasi query, indexing, serta stored procedure yang mendukung pengelolaan data dalam jumlah besar secara efisien dan aman [9].

5. Seafile, merupakan platform penyimpanan berbasis cloud yang memungkinkan sinkronisasi dan berbagi data secara aman. Seafile mendukung enkripsi dan version control sehingga cocok digunakan untuk kolaborasi dokumen dan menjaga integritas data [10].

B. Hardware

Spesifikasi perangkat keras yang digunakan selama pengembangan aplikasi CRM adalah sebagai berikut:

- Prosesor: Intel Core i5-8265U
- Memori: 16 GB RAM
- Dimensi Layar: 14.1 inci
- Penyimpanan: 512 GB SSD
- Sistem Operasi: Windows 11 Pro

3.4 Perancangan Aplikasi CRM (*Customer Relationship Management*)

3.4.1 Analisis Masalah dan Kebutuhan Pengguna

Dalam pengembangan aplikasi *Input Draft SO* ini, digunakan pendekatan *System Development Life Cycle* (SDLC) model *Rapid Application Development* (RAD), yaitu model pengembangan yang berfokus pada iterasi cepat dan pembuatan prototipe secara berulang. Pendekatan ini memungkinkan pengembang untuk merumuskan kebutuhan sistem baru secara fleksibel dan adaptif terhadap perubahan, sesuai dengan kebutuhan pengguna.

A Proses Bisnis

Proses pengelolaan *sales order* pada PT Dover Chemical Indonesia terdiri dari beberapa tahapan. Pertama, tim penjualan akan mengunjungi pabrik-pabrik klien untuk melakukan presentasi tentang produk yang ingin dibeli serta menyampaikan penawaran yang sesuai dengan kebutuhan klien. Dalam tahapan ini, tim *sales* membutuhkan informasi yang terkait dengan data pelanggan, rincian produk, serta riwayat transaksi sebelumnya.

Tahapan berikutnya dalam pengelolaan pesanan adalah pencatatan pesanan penjualan yang dilakukan oleh tim penjualan setelah mendapatkan konfirmasi dari pihak klien. Proses pencatatan yang dilakukan mencakup pengisian data pelanggan seperti nama pelanggan dan lokasi, detail produk yang dipesan termasuk *item*, kuantitas produk yang dipesan, harga *item*, *unit*, *unit price*, *amount*, serta

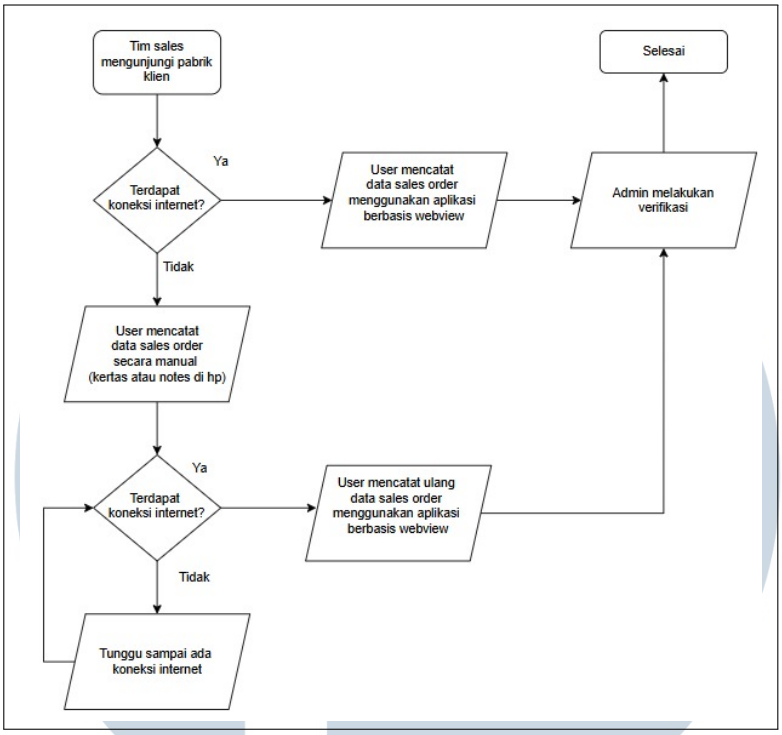
perhitungan total, *subtotal*, dan pajak. Setelah data pesanan dicatat, informasi tersebut kemudian diteruskan kepada tim administratif atau bagian admin yang bertugas untuk melakukan verifikasi. Jika disetujui, maka admin akan melakukan pemrosesan lanjutan yang terdiri dari validasi stok barang serta persiapan dokumen penunjang seperti *invoice* atau surat jalan.

B Analisis Sistem Lama

Sistem lama yang digunakan oleh PT Dover Chemical Indonesia merupakan sistem *Customer Relationship Management* (CRM) berbasis web yang dikembangkan menggunakan *framework* PHP CodeIgniter melalui pendekatan *WebView*. Sistem ini pada awalnya dikembangkan sebagai solusi *Minimum Viable Product* (MVP) untuk mengatasi masalah pencatatan manual yang kurang efisien dalam waktu yang relatif singkat. Meskipun sistem ini berhasil mengatasi kebutuhan yang mendesak dalam pengelolaan pesanan penjualan, terdapat beberapa keterbatasan yang dialami oleh tim penjualan PT Dover Chemical Indonesia.

Keterbatasan pertama dari sistem lama adalah ketidakmampuan untuk beroperasi tanpa koneksi internet. Tim penjualan PT Dover Chemical sering kali beroperasi di berbagai daerah dengan konektivitas internet yang terbatas, khususnya ketika mengunjungi pabrik-pabrik klien di berbagai provinsi yang membatasi penggunaan internet karena alasan keamanan perusahaan. Kondisi ini mengharuskan tim penjualan untuk melakukan pekerjaan ganda, yaitu mencatat data pesanan secara manual terlebih dahulu, kemudian menginput ulang data tersebut ke dalam sistem saat koneksi internet tersedia.

Permasalahan kedua adalah ketidakmampuan sistem *WebView* dalam menangani penyimpanan data secara *offline*. Berbagai informasi yang dibutuhkan oleh tim sales seperti basis data pengguna, unit penjualan produk, hitungan satuan per unit, dan riwayat transaksi tidak dapat diakses tanpa koneksi internet. Hal ini menyebabkan tim *sales* PT Dover Chemical Indonesia mengalami kesulitan saat memberikan presentasi produk atau penawaran harga kepada klien di lokasi dengan keterbatasan koneksi.



Gambar 3.1. Proses Bisnis Sistem Lama

C Analisis Kebutuhan Sistem

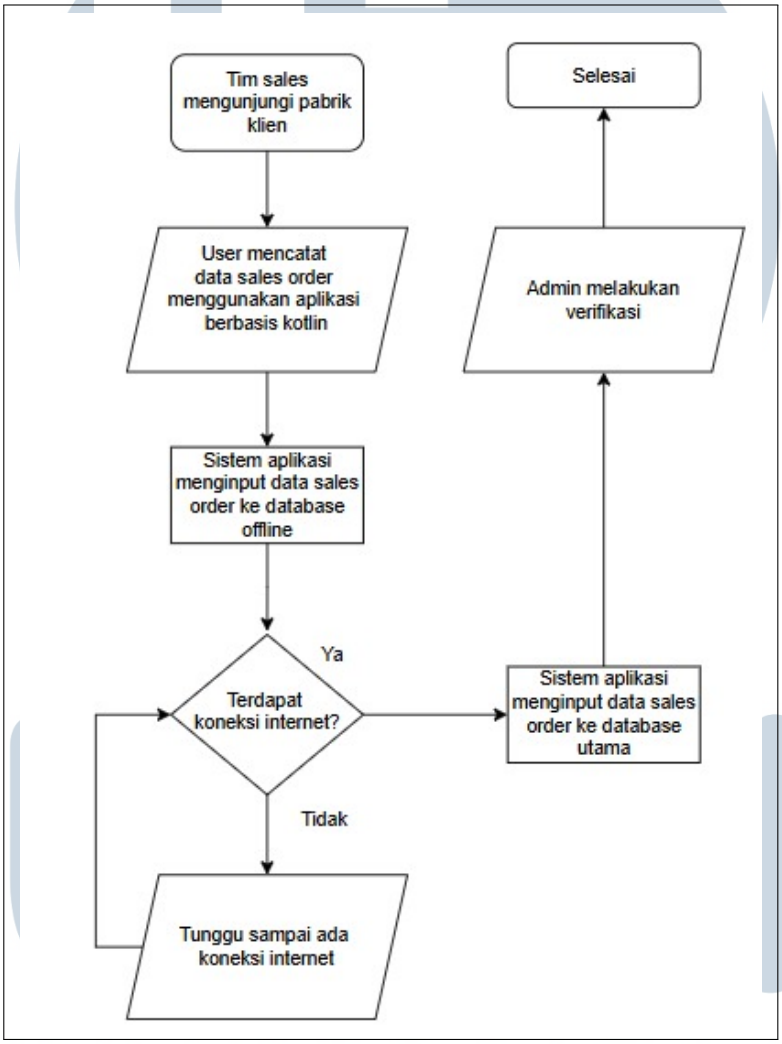
Berdasarkan identifikasi masalah pada sistem lama, analisis kebutuhan sistem untuk pengembangan aplikasi CRM Android dapat dikategorikan menjadi kebutuhan fungsional dan nonfungsional.

Kebutuhan fungsional mencakup kemampuan aplikasi untuk beroperasi secara luring dengan menyimpan data penting seperti basis data pelanggan, informasi produk, dan riwayat transaksi secara lokal menggunakan teknologi penyimpanan yang tersedia pada platform Android.

Sistem baru harus mampu mengelola proses pembuatan *sales order (SO)*, dimulai dari pembuatan draf *SO* baru dengan kolom input seperti nama pelanggan, lokasi, *item*, kuantitas, harga per *item*, *unit*, *unit price*, *amount*, total, *subtotal*, dan pajak. Sistem juga harus menyediakan fitur untuk mengedit draf *SO* yang telah dibuat, melihat riwayat draf, serta menghapus draf yang sudah tidak diperlukan. Selain itu, sistem perlu dilengkapi dengan fitur manajemen pengaturan pengguna untuk mengelola otoritas dan preferensi masing-masing pengguna.

Kebutuhan nonfungsional yang perlu dipenuhi mencakup kemampuan sinkronisasi otomatis yang akan mengirim data ke server pusat ketika koneksi

internet tersedia, sehingga dapat menghilangkan kebutuhan input data berulang dan mengurangi risiko kesalahan manusia. Sistem juga harus memiliki antarmuka pengguna yang mudah dipahami dan responsif dengan menerapkan prinsip desain seperti *Eight Golden Rules* dan psikologi warna untuk memberikan pengalaman pengguna yang lebih baik dibandingkan sistem *WebView* sebelumnya.



Gambar 3.2. Proses Bisnis Sistem Baru

D Analisis Sistem Baru

Sistem baru yang dikembangkan, dibuat menggunakan bahasa pemrograman *Kotlin* dengan implementasi *Room* DAO untuk penyimpanan data lokal, serta integrasi REST API menggunakan *Retrofit* dan backend berbasis PHP.

Keunggulan utama dari sistem baru ini adalah kemampuannya untuk beroperasi secara *offline*. Dengan demikian, tim penjualan tidak lagi bergantung pada koneksi internet saat melakukan proses pencatatan pesanan. Selain itu, sistem ini dilengkapi dengan fitur *sinkronisasi otomatis*, sehingga data yang dicatat saat *offline* akan langsung dikirim ke server ketika koneksi tersedia.

Fitur lain yang diimplementasikan dalam sistem baru adalah *global data fetching* yang dilakukan saat login dan pada halaman beranda. Data yang diperoleh akan disimpan ke database lokal dengan mekanisme *filtering* untuk memastikan hanya data baru yang dimasukkan atau diperbarui. Hal ini bertujuan untuk menjamin ketersediaan informasi terkini seperti data pelanggan, produk, dan riwayat transaksi tanpa memerlukan koneksi internet secara terus-menerus.

Adapun perbandingan permasalahan pada sistem lama dengan solusi pada sistem baru disajikan pada Tabel 3.2 berikut:

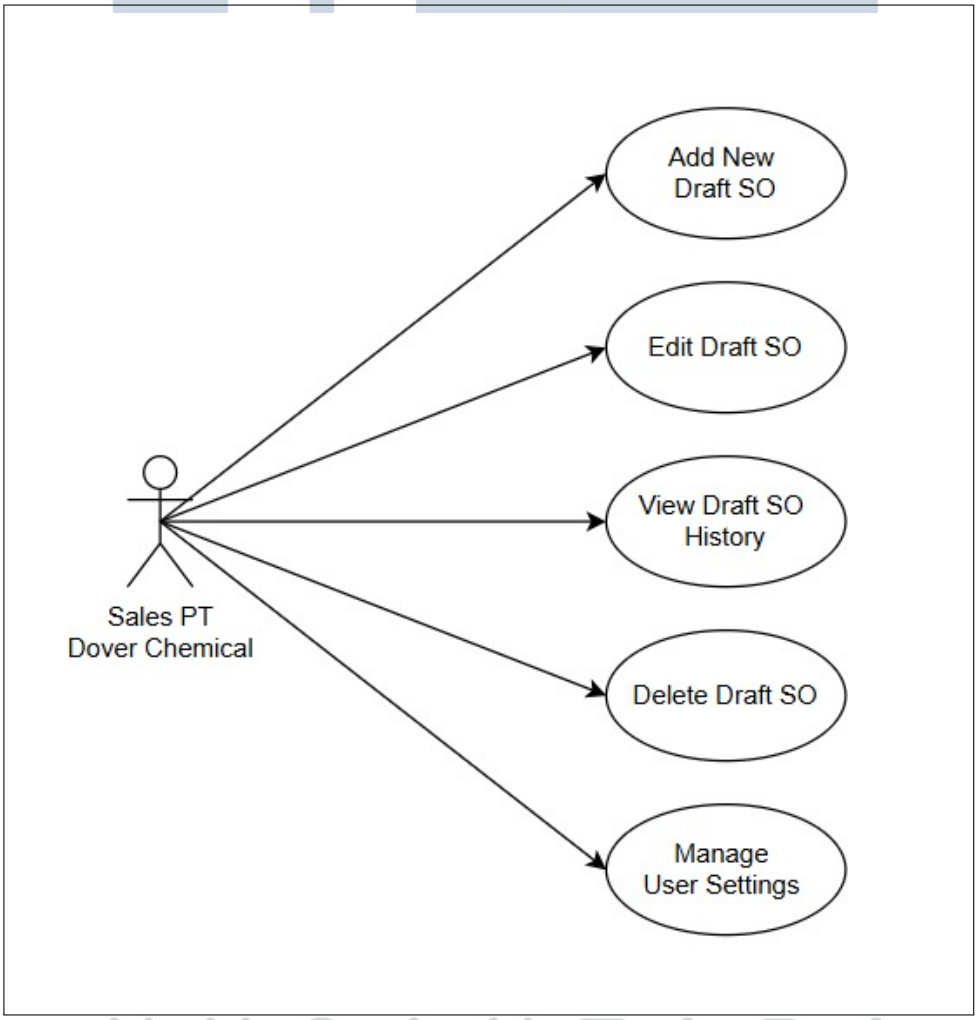
Tabel 3.2. Perbandingan Permasalahan Sistem Lama dan Solusi pada Sistem Baru

No.	**Permasalahan pada Sistem Lama**	**Perbaikan (Solusi) pada Sistem Baru**
1	Ketergantungan terhadap koneksi internet, sehingga aplikasi tidak bisa diakses saat tidak ada jaringan.	Implementasi mode *offline* dengan *Room DAO* untuk penyimpanan data lokal. Aplikasi tetap dapat digunakan tanpa internet.
2	Terjadi pekerjaan ganda: pencatatan manual dan input ulang saat koneksi tersedia.	Fitur *sinkronisasi otomatis* yang mengirim data ke server saat koneksi tersedia, dilengkapi dengan pembaruan status pada tabel transaksi.
3	Tidak dapat mengakses data penting seperti informasi produk, pelanggan, dan riwayat transaksi saat offline, menghambat proses negosiasi.	*Global fetching* pada login dan *HomeFragment*, menyimpan data lokal secara selektif menggunakan mekanisme *filtering*, sehingga dapat memastikan bahwa data pada form dan histori transaksi selalu diperbarui.

3.4.2 Perancangan Sistem

A Use Case Diagram

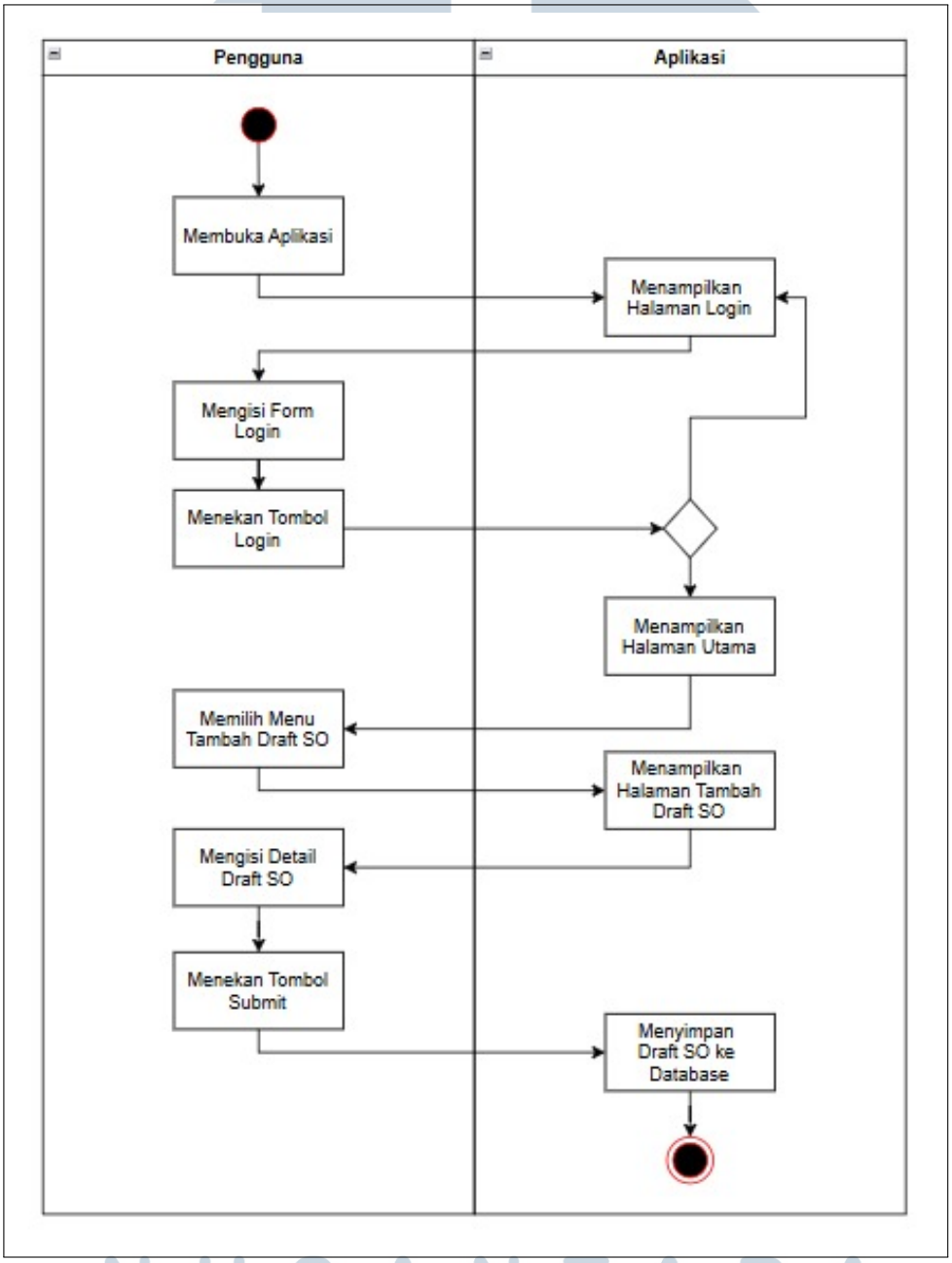
Diagram *use case* menggambarkan sistem berdasarkan perspektif pengguna aplikasi, yang menekankan pada berbagai macam fungsi yang ditawarkan oleh sistem, dan tidak berdasarkan urutan peristiwa. Komponen utama dari diagram *use case* meliputi aktor (tim penjualan PT Dover Chemical), *use case* (fungsi yang dapat digunakan dalam aplikasi), dan relasi yang menghubungkan keduanya. Diagram *use case* untuk aplikasi *input draft sales order* dapat dilihat pada Gambar 3.3.



Gambar 3.3. Diagram *Use Case* Aplikasi *Input Draft Sales Order*

B Activity Diagram

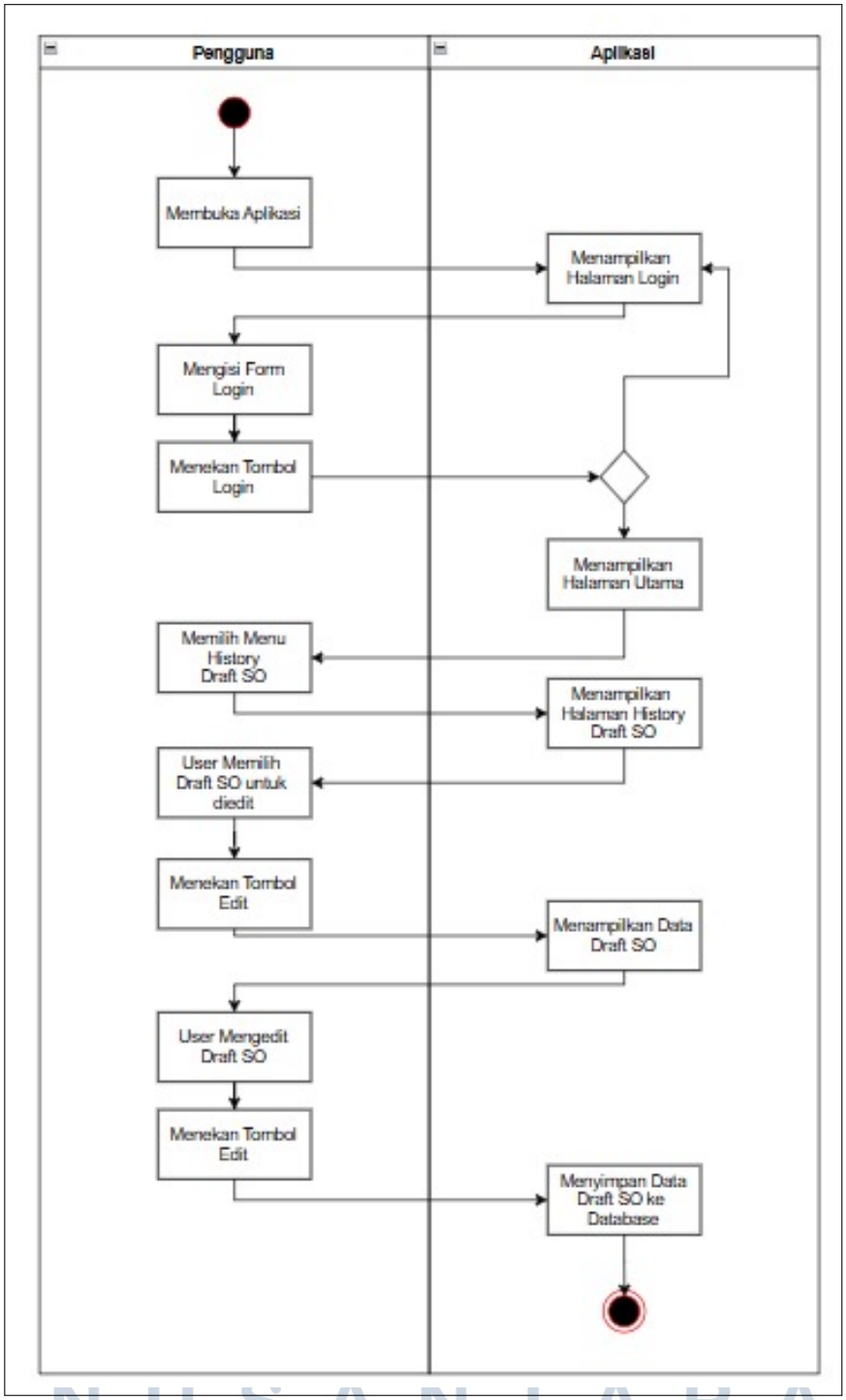
Activity diagram pada Gambar 3.4 menunjukkan alur dari proses yang terjadi ketika pengguna melakukan input *draft sales order* ke dalam aplikasi.



Gambar 3.4. *Activity Diagram Input Draft Sales Order*

Selanjutnya, Gambar 3.5 menampilkan *activity diagram* yang menggambarkan proses ketika pengguna mengedit *draft sales order* yang

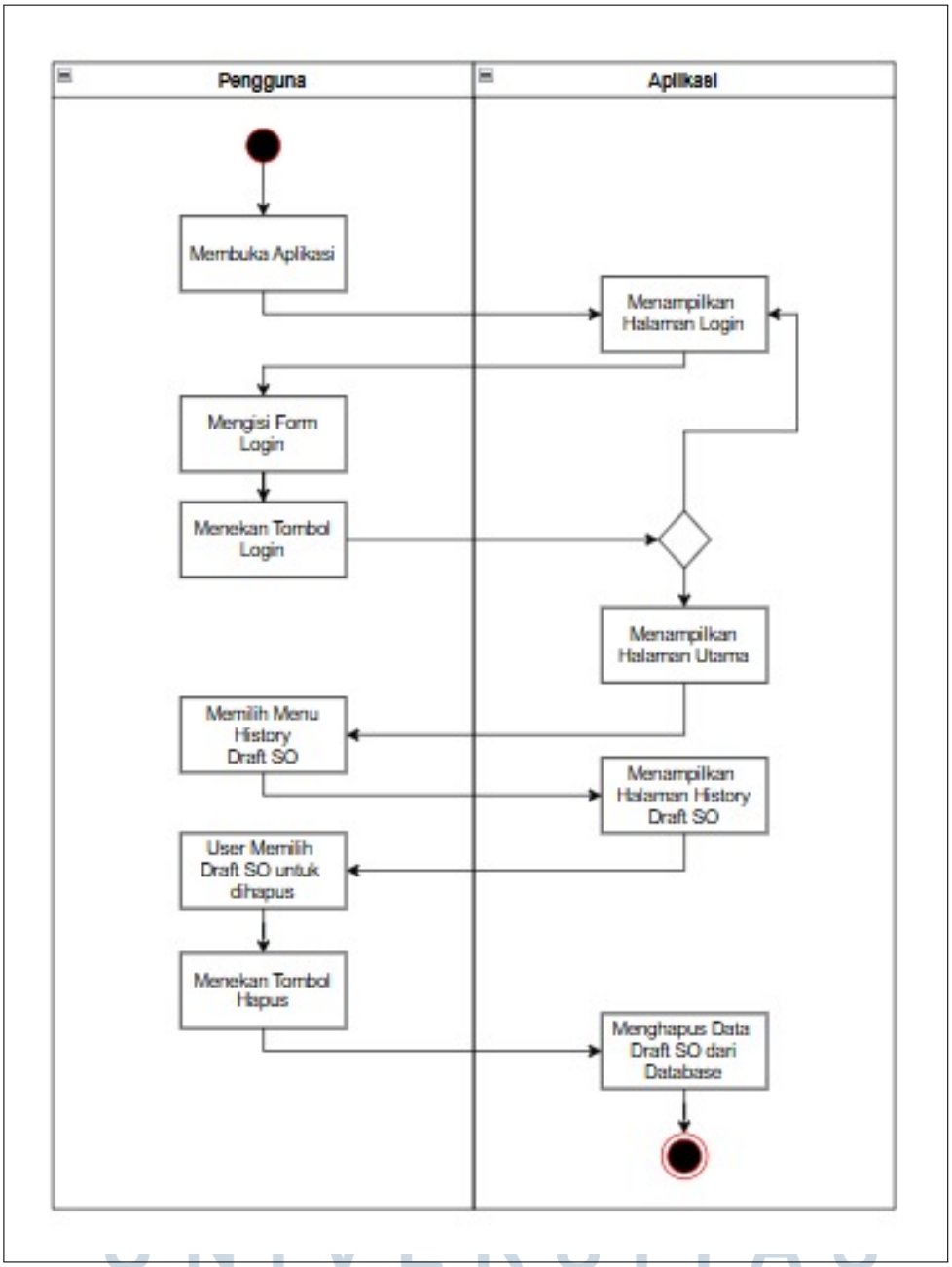
sudah diinput ke dalam aplikasi.



Gambar 3.5. *Activity Diagram Edit Draft Sales Order*

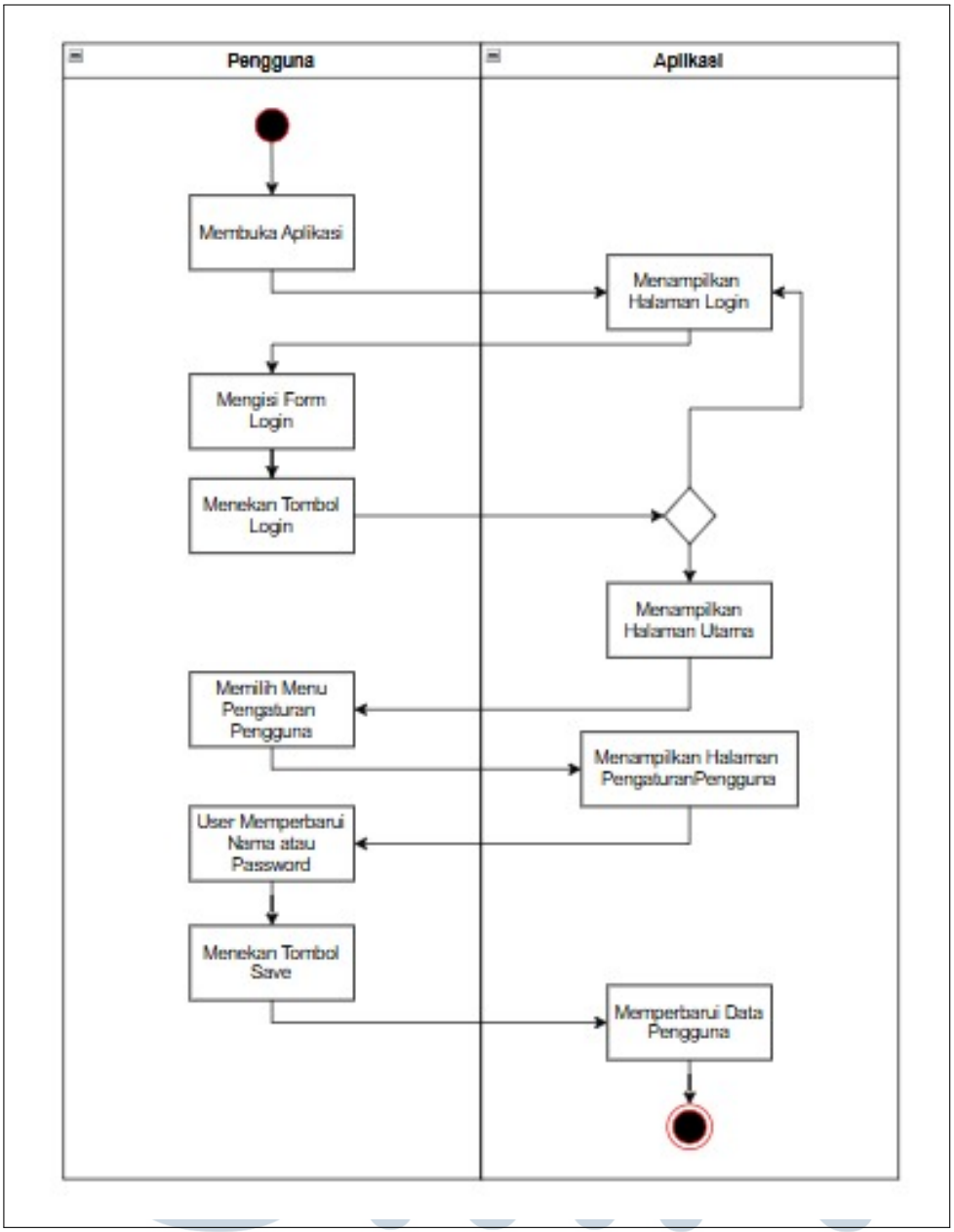
Selanjutnya, Gambar 3.6 menampilkan *activity diagram* yang menggambarkan proses ketika pengguna menghapus data *draft sales order*

dari basis data.



Gambar 3.6. *Activity Diagram Delete Draft Sales Order*

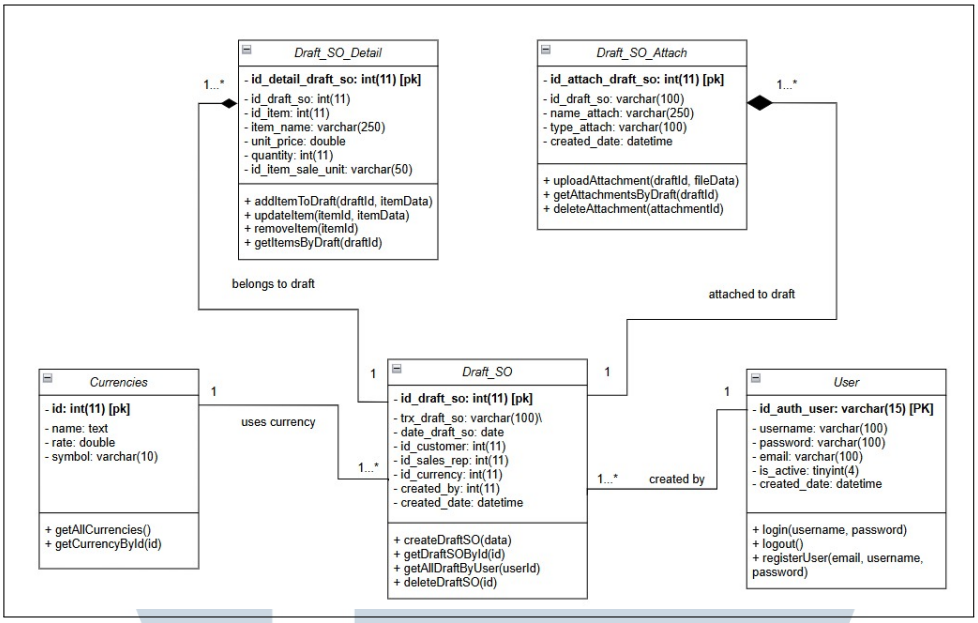
Selanjutnya, Gambar 3.7 memperlihatkan *activity diagram* yang menggambarkan proses pengguna melakukan pembaruan data pribadinya.



Gambar 3.7. *Activity Diagram Edit Informasi Pengguna*

C Class Diagram

Pada Gambar 3.8, terdapat *class diagram* untuk menunjukkan relasi dari setiap *class* yang digunakan sebagai acuan dalam perancangan aplikasi.



Gambar 3.8. *Class Diagram Database History*

Berikut merupakan penjabaran fungsi dari masing-masing tabel pada *class diagram* di atas:

1. User:
   Tabel ini menyimpan informasi data pengguna yang memiliki akses ke dalam sistem. Data yang disimpan meliputi *username*, *email*, *password*, serta status aktif pengguna. Tabel ini digunakan untuk kebutuhan proses autentikasi seperti *login*, *logout*, dan pendaftaran akun (*register*).

2. Currencies:
   Tabel ini menyimpan informasi mengenai mata uang yang digunakan dalam proses transaksi, seperti nama mata uang, nilai tukar, dan simbolnya. Data ini digunakan sebagai referensi ketika pengguna memilih jenis mata uang pada *form Draft Sales Order*.

3. Draft_SO:
   Tabel ini menyimpan data utama dari dokumen *Draft Sales Order* yang dibuat oleh pengguna. Data yang disimpan meliputi informasi pelanggan, perwakilan penjualan (*sales representative*), mata uang yang digunakan, serta informasi waktu pembuatan. Setiap kali pengguna mengisi dan menyimpan *form Draft Sales Order*, satu entri akan tercatat pada tabel ini.

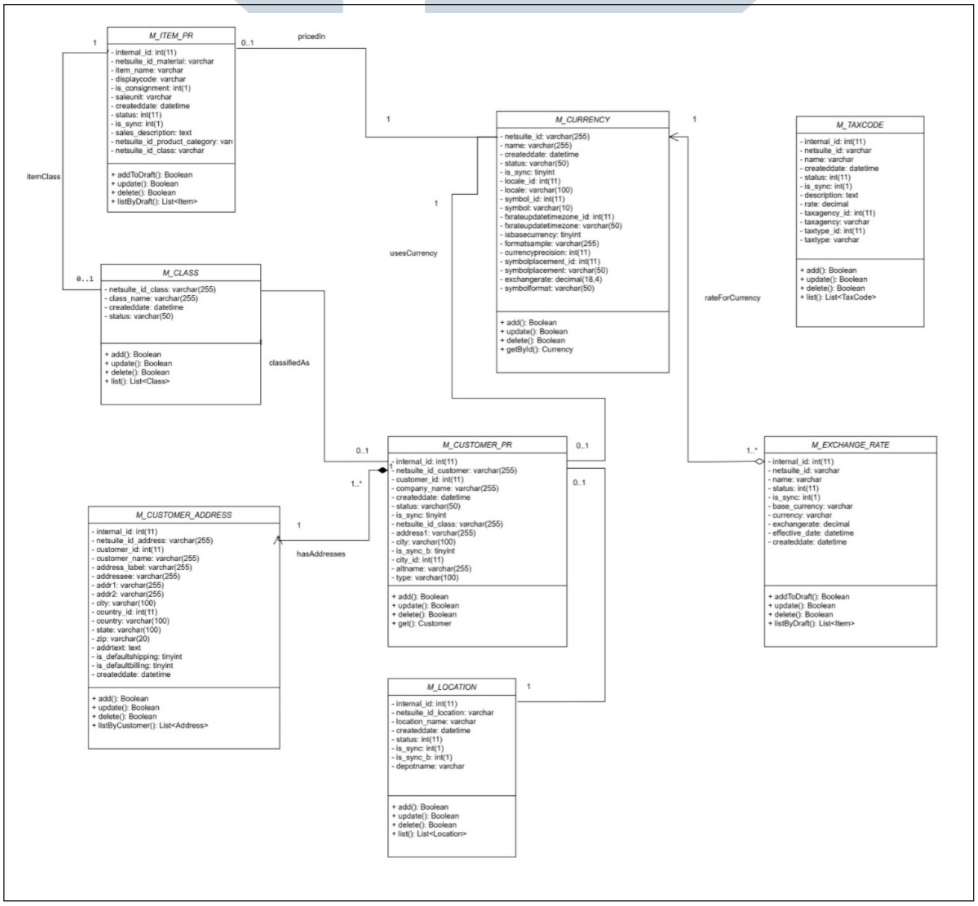
4. Draft_SO_Detail:

   Tabel ini menyimpan detail *item* dari setiap *Draft Sales Order* yang dibuat. Setiap entri pada tabel ini merepresentasikan satu *item* barang dalam satu *Draft Sales Order*, termasuk nama *item*, jumlah (*quantity*), *unit price*, serta satuan penjualan. Satu *Draft Sales Order* dapat memiliki satu atau lebih data *item* yang tercatat pada tabel ini.

5. Draft_SO_Attach:

   Tabel ini menyimpan data berkas (*file*) yang dilampirkan oleh pengguna dalam satu *Draft Sales Order*. Berkas ini dapat berupa gambar, dokumen, atau jenis lainnya yang berkaitan dengan transaksi. Dalam satu *Draft Sales Order*, pengguna dapat melampirkan satu, lebih dari satu, atau tidak sama sekali *file attachment*.

Pada Gambar 3.9, terdapat *class diagram* untuk menunjukkan relasi dari setiap *class* yang digunakan sebagai acuan dalam perancangan aplikasi.



Gambar 3.9. *Class Diagram Master Data*

Berikut merupakan penjabaran fungsi dari masing-masing tabel pada *class diagram* di atas:

1. M_CLASS:
   Tabel ini menyimpan data kategori atau kelas dari transaksi dan entitas dalam sistem ERP. Informasi yang dicatat meliputi *internal ID*, *NetSuite ID*, nama kelas, serta status dan informasi sinkronisasi. Data ini digunakan sebagai acuan pengelompokan transaksi atau entitas pelanggan.

2. M_CURRENCY:
   Tabel ini menyimpan daftar mata uang beserta simbol, kurs, dan preferensi tampilan seperti penempatan simbol dan presisi desimal. Data ini digunakan saat menentukan jenis mata uang untuk transaksi atau laporan keuangan.

3. M_CUSTOMER_PR:
   Tabel ini menyimpan informasi utama pelanggan, termasuk nama perusahaan, ID pelanggan, alamat, perwakilan penjualan, serta preferensi mata uang, pajak, dan status entitas. Tabel ini menjadi pusat referensi data pelanggan dalam berbagai proses bisnis.

4. M_CUSTOMER_ADDRESS:
   Tabel ini menyimpan alamat lengkap pelanggan, baik untuk pengiriman maupun penagihan. Setiap entri mencakup label alamat, kota, negara, dan kode pos, serta status default untuk pengiriman atau penagihan. Relasi tabel ini terhubung ke `M_CUSTOMER_PR` melalui kolom `customer_id`.

5. M_EXCHANGE_RATE:
   Tabel ini mencatat kurs tukar antar mata uang beserta tanggal efektivitas dan nilai tukar jual/beli. Tabel ini digunakan untuk konversi nilai transaksi lintas mata uang secara akurat sesuai periode waktu tertentu.

6. M_ITEM_PR:
   Tabel ini menyimpan daftar barang atau item, termasuk nama item, satuan, jenis, deskripsi penjualan, dan mata uang yang digunakan. Juga mencakup informasi pengelompokan seperti kategori produk dan departemen. Item digunakan dalam transaksi seperti penjualan atau pengadaan.

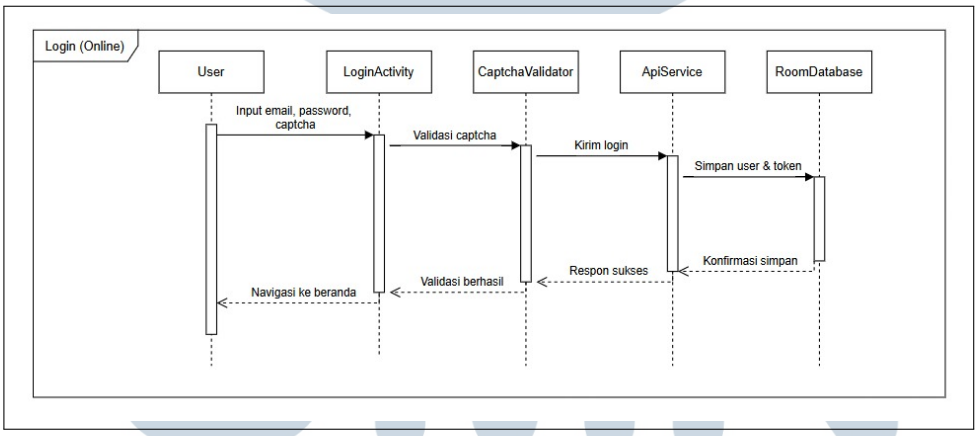
7. M_LOCATION:
   Tabel ini mencatat data lokasi gudang atau cabang, seperti nama lokasi, nama

depot, dan status sinkronisasi. Lokasi digunakan untuk menentukan sumber atau tujuan pengiriman dalam transaksi barang.

8. M_TAXCODE:
   Tabel ini menyimpan kode pajak beserta tarif, deskripsi, dan relasi ke entitas pajak seperti agensi dan jenis pajak. Data ini digunakan saat menghitung nilai pajak dalam transaksi penjualan atau pembelian.

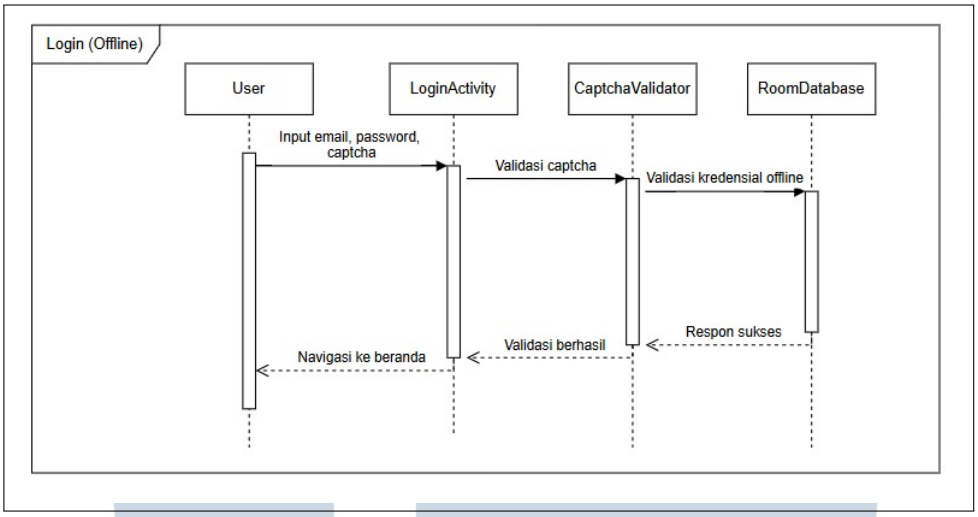
D Sequence Diagram

Gambar 3.10 menunjukkan urutan interaksi antar objek pada proses *login* saat aplikasi berada dalam kondisi *online*. Dalam proses *login* secara *online*, pengguna akan mengisi *e-mail* perusahaan dan kata sandi, melakukan autentikasi *captcha*, kemudian sistem akan memavalidasi data yang dimasukkan pengguna ke database yang ada. Setelah sistem mendapatkan informasi apakah pengguna sudah terdaftar di *database* atau belum, maka sistem akan memberikan *feedback* kepada pengguna.



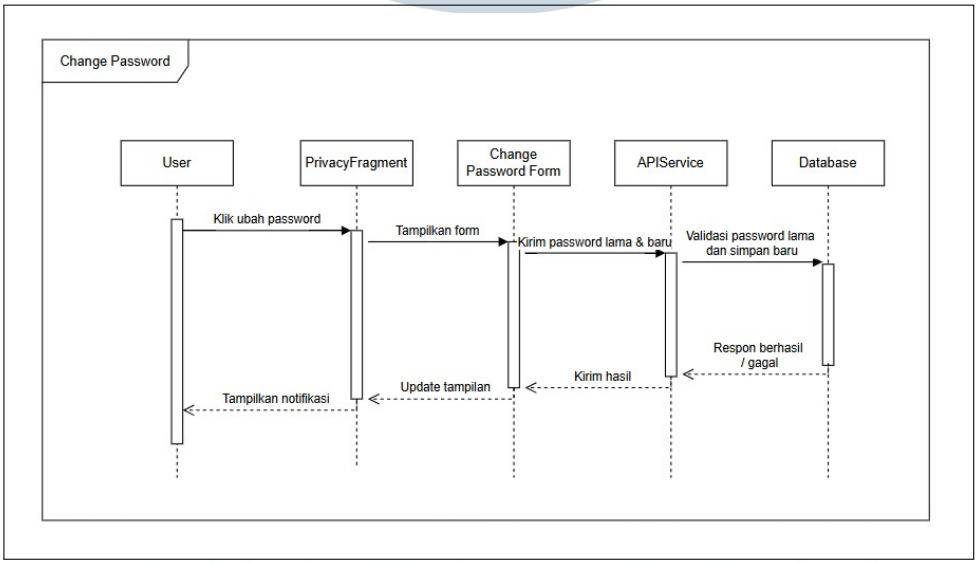
Gambar 3.10. *Sequence Diagram Login (Online)*

Gambar 3.11 menunjukkan urutan interaksi antar objek pada proses *login* ketika aplikasi digunakan dalam kondisi *offline*. Diagram ini menggambarkan mekanisme autentikasi yang dilakukan secara lokal melalui penyimpanan data pada *database* internal perangkat.



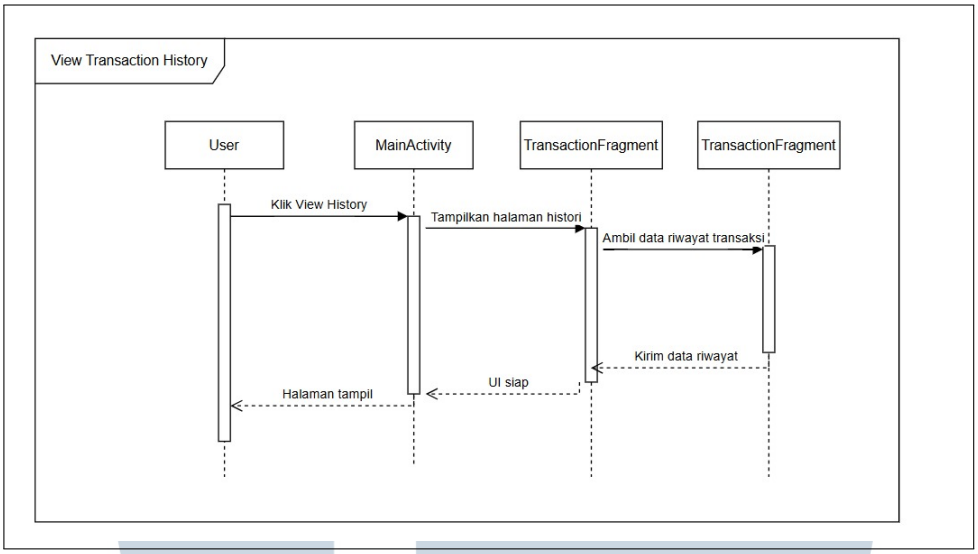
Gambar 3.11. *Sequence Diagram Login (Offline)*

Gambar 3.12 menampilkan alur interaksi antar objek saat pengguna melakukan proses *change password*. Proses ini melibatkan validasi kata sandi lama, input kata sandi baru, dan pembaruan data pengguna ke dalam *database* melalui sistem yang tersedia.



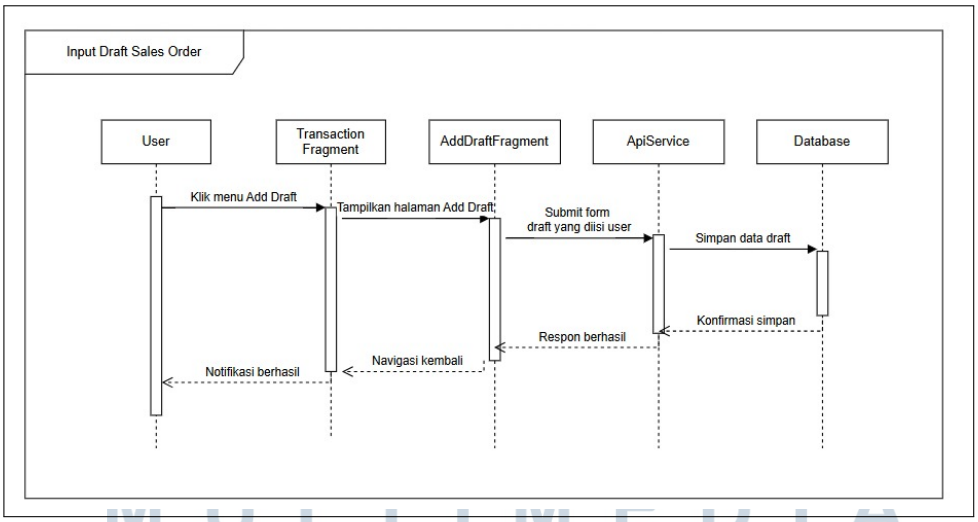
Gambar 3.12. *Sequence Diagram Change Password*

Gambar 3.13 memperlihatkan proses interaksi antar objek yang terjadi saat pengguna mengakses fitur *view transaction history*. Diagram ini menyajikan alur komunikasi antara antarmuka pengguna dan sistem dalam mengambil serta menampilkan data histori transaksi dari sumber penyimpanan yang tersedia.



Gambar 3.13. *Sequence Diagram View Transaction History*

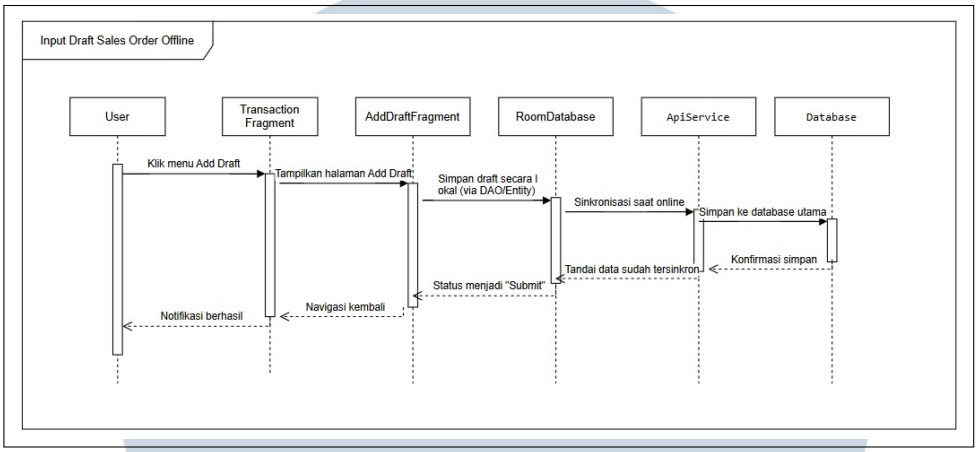
Gambar 3.14 mengilustrasikan rangkaian interaksi antar objek ketika pengguna menginput *draft sales order* dalam kondisi *online*. Pada proses ini, sistem terlebih dahulu mengambil data referensi dari *API*, kemudian pengguna mengisi formulir, dan data yang telah dikirim akan disimpan ke dalam *database* melalui layanan jaringan.



Gambar 3.14. *Sequence Diagram Input Draft Sales Order (Online)*

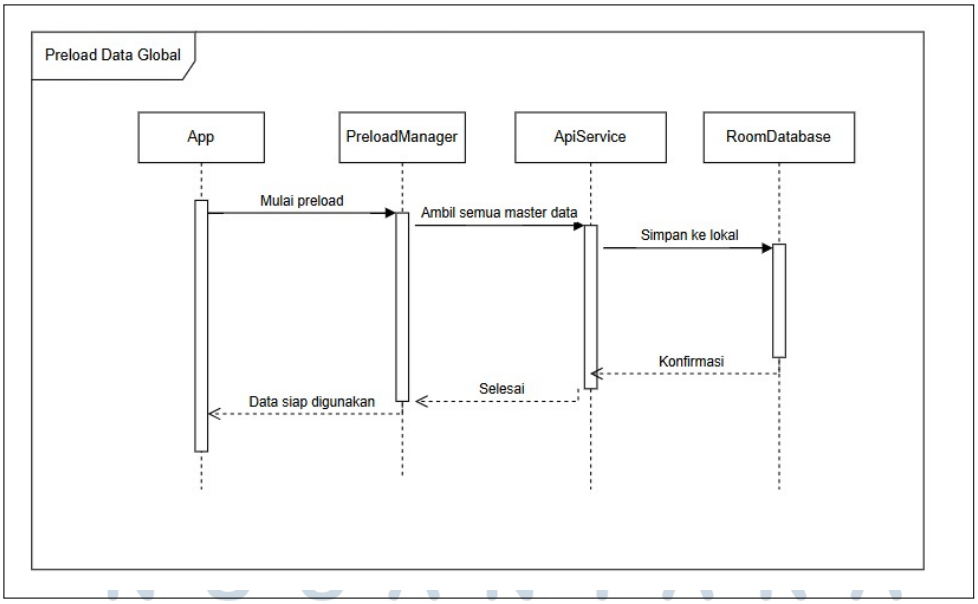
Gambar 3.15 menggambarkan proses pengisian *draft sales order* saat aplikasi berada dalam kondisi *offline*. Dalam keadaan ini, data transaksi yang diinput oleh pengguna akan disimpan secara lokal ke dalam *database* dengan status

pending. Ketika koneksi internet tersedia kembali, sistem akan melakukan proses sinkronisasi otomatis untuk mengirim data ke server.



Gambar 3.15. *Sequence Diagram Input Draft Sales Order (Offline)*

Gambar 3.16 menunjukkan urutan interaksi antar objek dalam proses *preload data global*. Aktivitas ini dijalankan saat aplikasi dimulai untuk mengambil berbagai data referensi dari *API* seperti pelanggan, item, lokasi, dan mata uang. Seluruh data yang berhasil dimuat kemudian disimpan secara lokal ke dalam *database* agar dapat diakses saat *offline*.



Gambar 3.16. *Sequence Diagram Preload Data Global*

3.4.3 Tampilan Aplikasi

Aplikasi mobile ini dikembangkan menggunakan bahasa pemrograman Kotlin untuk bagian *front-end*, dengan dukungan *back-end* berbasis REST API yang dibangun menggunakan bahasa pemrograman PHP dan terhubung ke sistem basis data MySQL. Seluruh proses pengelolaan data, seperti autentikasi pengguna, pengiriman data transaksi, hingga pengambilan data referensi dilakukan melalui jalur *API* yang aman dan terstruktur.

Desain antarmuka aplikasi ini juga memperhatikan prinsip-prinsip dalam psikologi warna, di mana warna-warna seperti biru digunakan untuk memberikan kesan profesional dan dapat dipercaya, sementara warna biru muda digunakan untuk memberikan rasa aman, misalnya ketika user sedang mengisi *form draft SO*. Pemilihan warna yang tepat ini dimaksudkan untuk membentuk persepsi pengguna yang positif terhadap aplikasi.

Lebih lanjut, perancangan UI aplikasi ini mengacu pada *Eight Golden Rules of Interface Design* yang dikemukakan oleh Ben Shneiderman, yaitu:

1. *Strive for consistency*: Setiap elemen tampilan memiliki gaya dan interaksi yang konsisten di seluruh halaman aplikasi.

2. *Enable frequent users to use shortcuts*: Navigasi berupa *header* dan *side-bar* disediakan melalui ikon-ikon dan tombol utama pada halaman beranda.

3. *Offer informative feedback*: Sistem memberikan umpan balik langsung melalui *snackbar* atau dialog, seperti saat autentikasi gagal atau transaksi berhasil.

4. *Design dialogs to yield closure*: Setiap aksi penting seperti login atau pembuatan draft akan diikuti dengan notifikasi atau transisi yang menandakan proses telah selesai.

5. *Offer simple error handling*: Pesan kesalahan ditampilkan secara spesifik dan informatif untuk memudahkan perbaikan oleh pengguna.

6. *Permit easy reversal of actions*: Aplikasi menyediakan tombol kembali dan konfirmasi sebelum aksi kritis dilakukan.

7. *Support internal locus of control*: Pengguna memiliki kendali penuh terhadap navigasi dan aksi dalam aplikasi.

8. *Reduce short-term memory load*: Informasi penting seperti nama pengguna dan data perusahaan ditampilkan secara langsung di layar utama, sehingga pengguna tidak perlu mencarinya di menu lain atau mengingat secara manual. Selain itu, dalam proses pengisian *form Draft SO*, data seperti nama *sales*, lokasi *default*, dan nilai *sale unit* telah terset secara otomatis berdasarkan preferensi atau data terakhir yang digunakan, sehingga pengguna tidak perlu mengingat atau mengisi ulang data yang bersifat berulang.

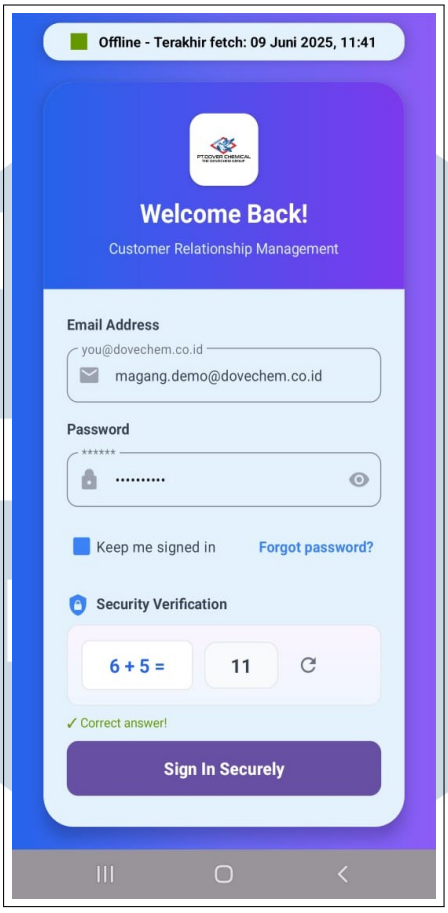
Berikut ini merupakan penjelasan dari masing-masing tampilan (halaman) yang terdapat dalam aplikasi yang telah dirancang.

A Halaman Login

Gambar 3.17 menampilkan tampilan halaman *login* yang digunakan oleh pengguna untuk masuk ke dalam sistem. Hanya pengguna yang telah terdaftar di dalam basis data internal yang dapat melakukan proses autentikasi ini. Sebelum menekan tombol *login*, pengguna diwajibkan untuk mengisi *captcha* sebagai langkah validasi tambahan guna menghindari *bot* atau aktivitas mencurigakan lainnya.

Jika terjadi kesalahan saat menginputkan alamat *email* atau *password*, maka akan muncul notifikasi berupa *snackbar* di bagian bawah layar yang memberikan pesan kesalahan yang spesifik. Misalnya, jika *email* tidak terdaftar, maka sistem akan menampilkan peringatan bahwa *email* tidak dikenali. Sebaliknya, jika *email* sudah benar namun *password* salah, maka akan ditampilkan pesan bahwa *password* tidak sesuai.

Selain itu, sistem ini juga dilengkapi dengan mekanisme *global fetching* yang berfungsi untuk mengambil dan menyimpan data-data penting dari *server*, seperti data yang dibutuhkan oleh *form Add Draft*, *History Transaction*, dan data milik pengguna. Pendekatan ini dirancang untuk menjamin ketersediaan data secara *offline* pada saat dibutuhkan oleh aplikasi.



Gambar 3.17. *Halaman Login*

B Halaman Beranda

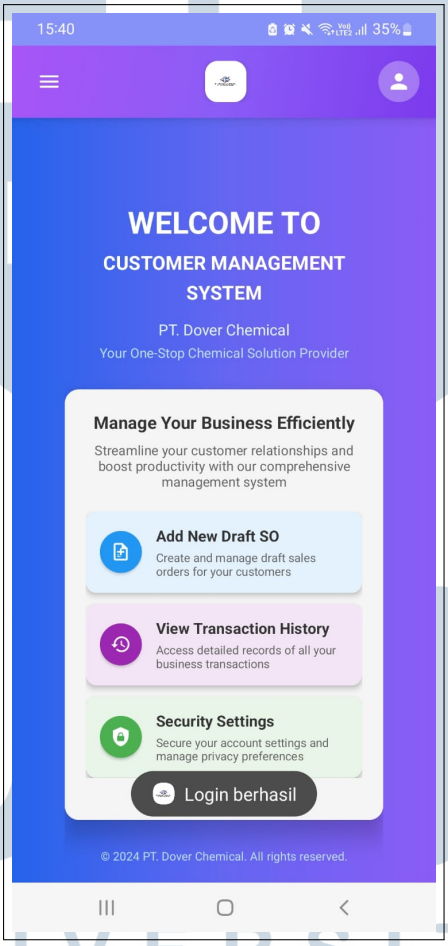
Gambar 3.18 menunjukkan tampilan halaman *Home* yang muncul setelah pengguna berhasil masuk ke dalam sistem. Pada halaman ini, pengguna disambut dengan pesan *"Welcome to Customer Management System"* beserta nama perusahaan, sebagai bentuk personalisasi yang meningkatkan pengalaman pengguna.

Tampilan utama berisi tiga *card* interaktif yang masing-masing mewakili fitur penting dalam aplikasi. Ketiga *card* ini dirancang agar mudah dikenali dan dapat ditekan untuk langsung mengakses fungsinya, sehingga mempermudah navigasi bagi pengguna.

- Add New Draft SO: Mengarahkan pengguna untuk membuat dan mengelola *draft sales order* baru.

- View Transaction History: Menyediakan akses ke riwayat transaksi pengguna yang tersimpan di dalam sistem.
- Security Settings: Memberikan opsi bagi pengguna untuk mengatur preferensi keamanan dan privasi akun mereka.

Desain ini dibuat dengan mempertimbangkan kemudahan akses dan efisiensi, sehingga pengguna dapat langsung memilih fitur yang dibutuhkan tanpa harus menavigasi ke banyak halaman.



Gambar 3.18. *Halaman Beranda*

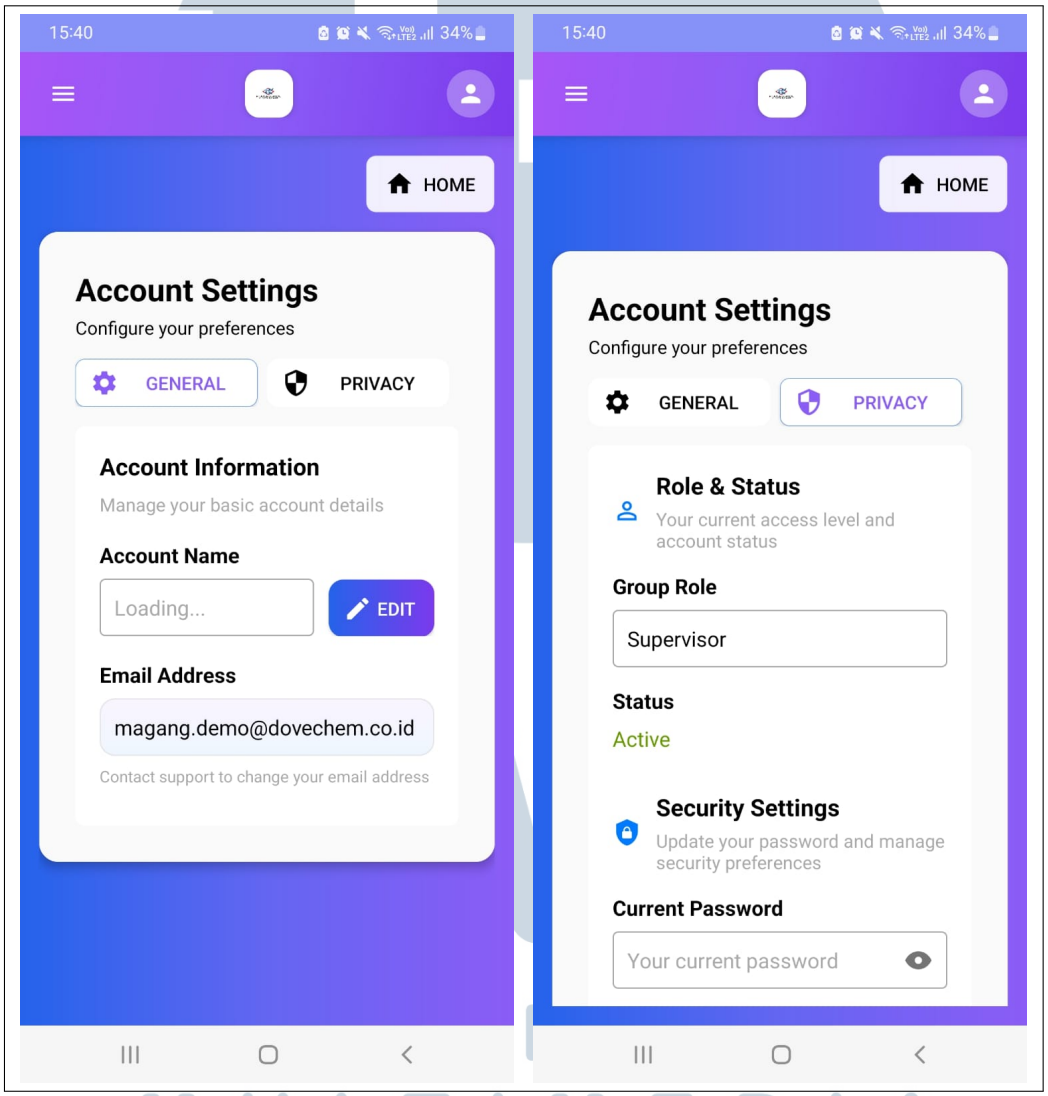
C Halaman Pengaturan

Gambar 3.19 memperlihatkan tampilan halaman *account settings* yang terdiri atas dua bagian utama, yaitu tab *general* dan *privacy*.

Pada bagian *general*, pengguna dapat melihat peran serta status akun saat ini, dan melakukan pengaturan keamanan seperti mengubah kata sandi. Sementara

itu, tab *privacy* menampilkan informasi akun dasar seperti nama akun dan alamat surel yang digunakan dalam sistem.

Kedua tampilan ini disatukan dalam satu antarmuka agar pengguna dapat dengan mudah menavigasi dan mengelola preferensi akun mereka tanpa perlu berpindah halaman. Desain ini mendukung kenyamanan serta efisiensi dalam pengelolaan akun secara terpusat.



Gambar 3.19. *Tampilan Halaman Pengaturan Akun (General dan Privacy)*

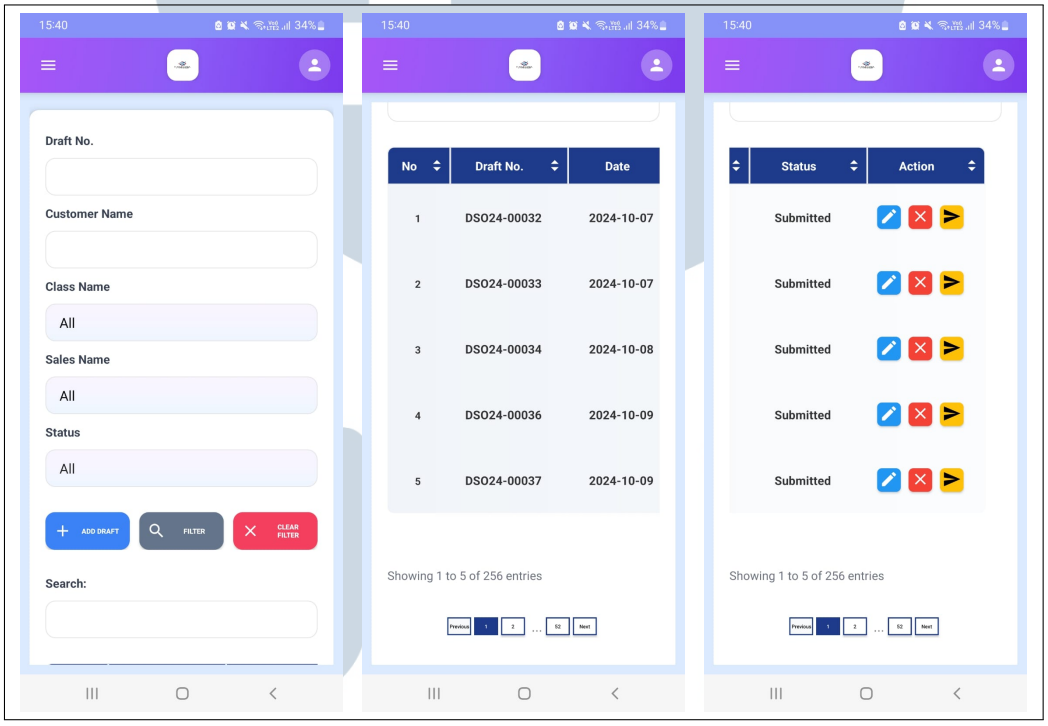
D Halaman Histori Transaksi

Gambar 3.20 menampilkan halaman *histori transaksi* yang memungkinkan pengguna untuk menelusuri riwayat transaksi sebelumnya melalui beragam filter.

Filter ini mempermudah pencarian data berdasarkan kriteria tertentu seperti tanggal, jenis transaksi, atau status pengajuan.

Pada setiap entri transaksi, ditampilkan status saat ini, apakah *pending* atau *submitted*. Status *pending* menunjukkan bahwa data belum tersimpan ke basis data, sementara status *submitted* berarti data telah tersimpan dan terverifikasi dalam sistem.

Selain melihat detail transaksi, pengguna juga dapat melakukan aksi lanjutan, yaitu mengedit data baik yang masih berstatus *pending* maupun yang sudah *submitted*, serta menghapus entri transaksi bila diperlukan. Penyatuan berbagai fitur ini dalam satu halaman mendukung fleksibilitas pengguna dalam mengelola dan meninjau data transaksi secara efisien.



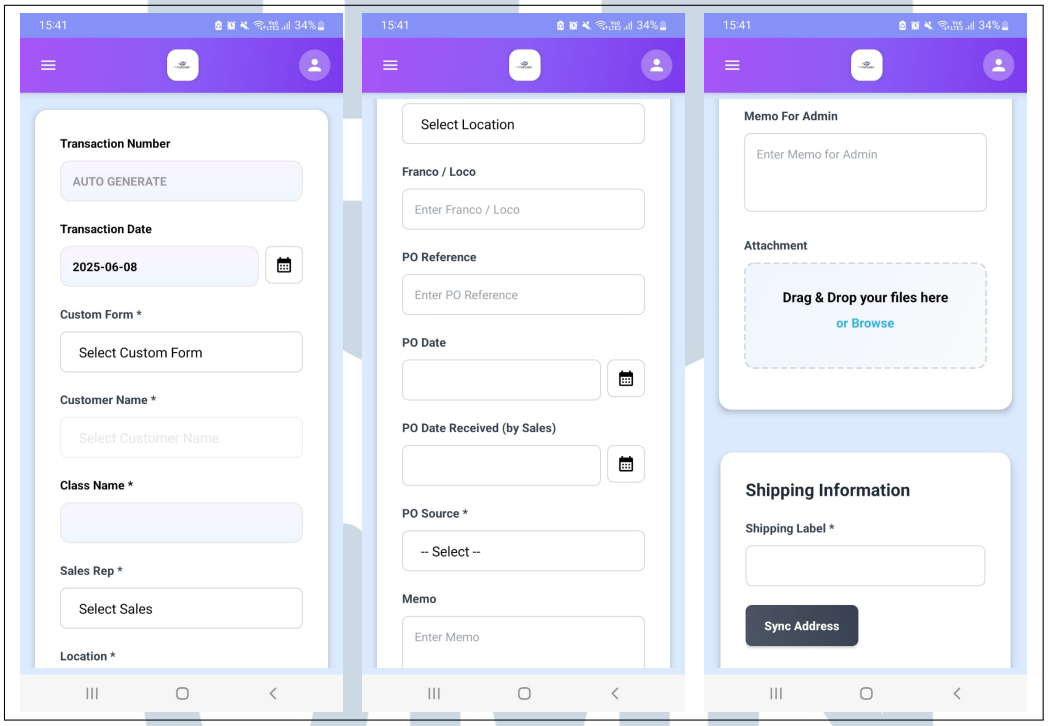
Gambar 3.20. *Tampilan halaman histori transaksi dengan fitur filter, status, dan aksi lanjutan*

E Halaman Input Draft Sales Order

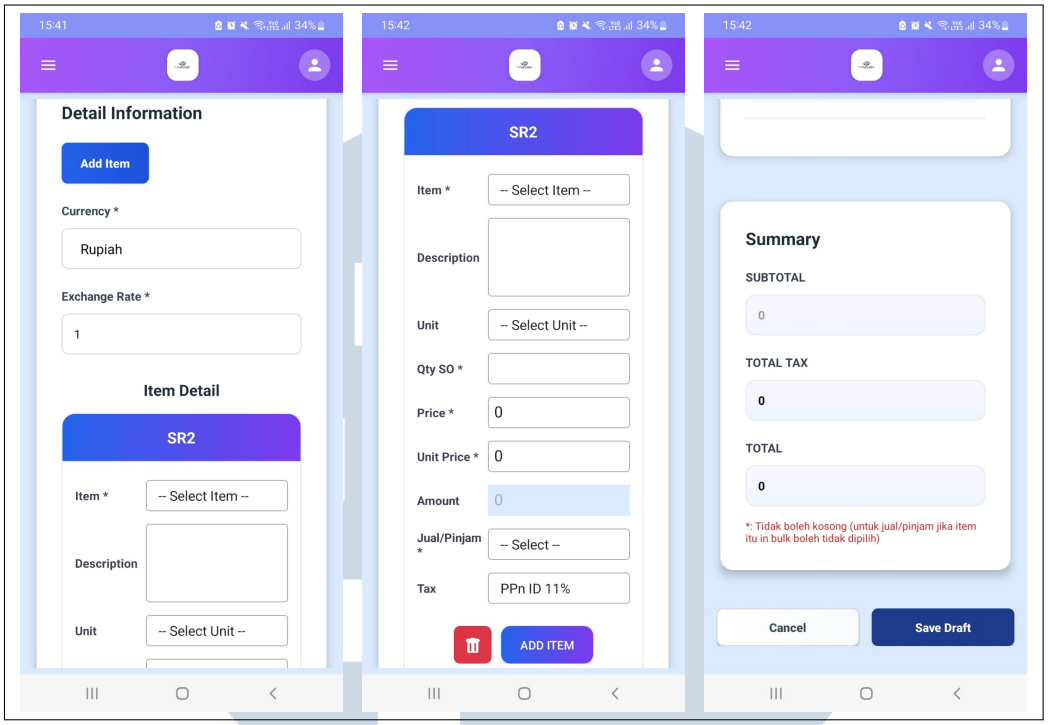
Halaman *Input Draft Sales Order* dirancang untuk mendukung proses pencatatan pemesanan dari pelanggan secara efisien dan fleksibel. Pada bagian awal, pengguna dapat memilih pelanggan dari daftar yang tersedia menggunakan fitur pencarian, kemudian mengisi informasi utama seperti tanggal pesanan, metode

pembayaran, dan nomor referensi. Setelah informasi dasar diisi, pengguna dapat menambahkan rincian barang dengan menekan tombol *Add Item*, yang akan memunculkan formulir untuk entri data item. Formulir ini mencakup input nama produk, kuantitas, satuan, harga satuan, serta catatan tambahan jika diperlukan.

Setiap perubahan pada kuantitas atau harga satuan akan langsung memperbarui nilai *subtotal* secara otomatis melalui mekanisme *text watcher*, yang secara real-time juga menghitung *total tax* dan *grand total*. Di bagian bawah halaman, terdapat dua tombol aksi, yaitu *Cancel* untuk keluar tanpa menyimpan, dan *Save Draft* untuk menyimpan data sementara ke dalam sistem.



Gambar 3.21. *Tampilan awal halaman input draft, pemilihan customer, dan pengisian informasi sales order*



Gambar 3.22. *Tampilan penambahan item, perhitungan subtotal, dan aksi penyimpanan draft*

3.4.4 Pengujian Sistem

Pengujian *black box* dilakukan pada tahap pemeliharaan sistem untuk memastikan bahwa aplikasi tetap berjalan sesuai dengan kebutuhan dan spesifikasi setelah dilakukan perubahan. Metode ini berfokus pada pengujian data masukan dan keluaran tanpa melihat langsung ke kode sumber, sehingga dapat menilai fungsionalitas aplikasi dari sudut pandang pengguna akhir.

Tabel 3.3. Black Box Testing Sistem Aplikasi Customer Management

No	Fitur	Kasus	Output	Hasil Uji	Kesimpulan
1	Login	Email dan password valid	Pengguna berhasil login ke sistem	Sesuai harapan	Lulus
2	Login	Email tidak terdaftar	Muncul notifikasi email tidak dikenali	Sesuai harapan	Lulus

Lanjut pada halaman berikutnya

Tabel 3.3: Black Box Testing Sistem Aplikasi Customer Management (lanjutan)

No	Fitur	Kasus	Output	Hasil Uji	Kesimpulan
3	Login	Password salah	Muncul pesan error login	Sesuai harapan	Lulus
4	Login	Captcha kosong	Validasi gagal ditampilkan	Sesuai harapan	Lulus
5	Beranda	Klik Add Draft	Beralih ke form input draft	Sesuai harapan	Lulus
6	Beranda	Klik View History	Halaman histori tampil	Sesuai harapan	Lulus
7	Beranda	Klik Pengaturan	Halaman akun terbuka	Sesuai harapan	Lulus
8	Akun	Ubah password	Password berhasil diubah	Sesuai harapan	Lulus
9	Akun	Edit nama/email	Data akun tersimpan	Sesuai harapan	Lulus
10	Riwayat	Filter transaksi	Data sesuai filter tampil	Sesuai harapan	Lulus
11	Riwayat	Edit transaksi	Form edit tampil otomatis	Sesuai harapan	Lulus
12	Riwayat	Hapus transaksi	Data dihapus dan notifikasi	Sesuai harapan	Lulus
13	Draft	Tambah item	Item tampil dan subtotal otomatis	Sesuai harapan	Lulus
14	Draft	Ubah qty/harga	Total berubah otomatis	Sesuai harapan	Lulus
15	Draft	Simpan draft	Draft tersimpan ke sistem	Sesuai harapan	Lulus
16	Fitur *Offline*	Login offline	Pengguna dapat login tanpa koneksi	Sesuai harapan	Lulus
17	Fitur *Offline*	Ambil data offline	Data global tersedia saat offline	Sesuai harapan	Lulus

Lanjut pada halaman berikutnya

Tabel 3.3: Black Box Testing Sistem Aplikasi Customer Management (lanjutan)

No	Fitur	Kasus	Output	Hasil Uji	Kesimpulan
18	Fitur *Offline*	Isi form offline	Form dapat diisi tanpa kendala	Sesuai harapan	Lulus
19	Fitur *Offline*	Hitung subtotal, pajak, total	Perhitungan berjalan akurat offline	Sesuai harapan	Lulus

3.5 Kendala dan Solusi yang Ditemukan

A Kendala

Kendala yang ditemui selama proses pengujian sistem adalah sebagai berikut.

1. Beberapa fitur tidak berjalan sempurna pada tahap awal karena adanya ketidaksesuaian antara rancangan awal dan implementasi aktual.
2. Terjadi kendala saat pengujian notifikasi file unduhan, di mana file tidak langsung muncul di folder *Downloads* atau notifikasi hilang terlalu cepat.

B Solusi

Solusi yang diterapkan untuk mengatasi kendala tersebut antara lain sebagai berikut.

1. Melakukan *debugging* terhadap fitur yang bermasalah dan melakukan sinkronisasi ulang antara implementasi dan desain sistem awal.
2. Mengecek ulang konfigurasi direktori penyimpanan file dan menambahkan *permission* serta pengaturan notifikasi agar proses unduhan dapat berjalan sesuai harapan.