BAB IV
HASIL DAN ANALISIS PENELITIAN

4.1 Menetapkan Tujuan *Prototype*

Pengembangan aplikasi dimulai dengan menentukan kebutuhan (*Establish Prototype Objectives*) dasar pada *dashboard* penjualan. Peneliti dan pengguna bersama-sama mendefinisikan format *dashboard* penjualan. Tujuan tersebut harus spesifik dan terukur agar memudahkan peneliti dalam melakukan pengembangan.

Dashboard penjualan tersebut akan digunakan untuk memantau data penjualan harian dan bulanan untuk mencapai target yang sudah ditetapkan. Pengguna bisa menganalisis tren transaksi penjualan pelanggan dan performa produk pada setiap periode. Hasil analisis tersebut bisa memberikan gambaran kepada pengguna untuk lebih cepat dan tepat dalam membuat keputusan bisnis berdasarkan data yang sudah teruji validitasnya.

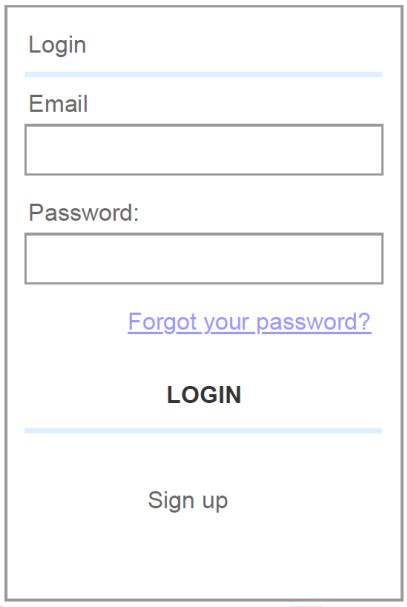
4.2 Menetapkan Fungsionalitas *Prototype*

Berdasarkan tujuan *prototype* yang sudah ditentukan, peneliti menetapkan fitur dan fungsionalitas (*Define Prototype Functionality*) dari *dashboard* yang akan dikembangkan. Peneliti membuat daftar fungsionalitas *dashboard* tersebut, yaitu :

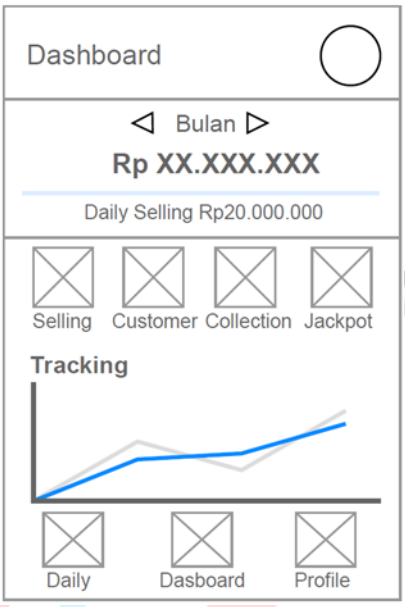
1. Menampilkan ringkasan data penjualan yang *update* setiap hari dan membandingkan antara penjualan dengan target.
2. Menampilkan visualisasi data dalam bentuk grafik untuk menggambarkan tren penjualan.
3. Menampilkan data pesanan (*pending order*) yang belum terkirim sampai periode tertentu. Pengguna bisa menghitung estimasi penjualan (*selling*) yang akan dicapai dari pesanan tersebut ditambah dengan pesanan yang sudah terkirim.

4. Menampilakan daftar pengiriman atau surat jalan pada setiap hari untuk memudahkan pengguna monitoring pesanan.

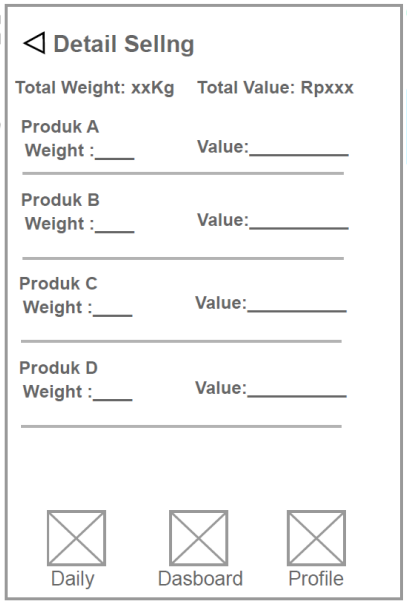
4.2.1 *Prototype* Awal



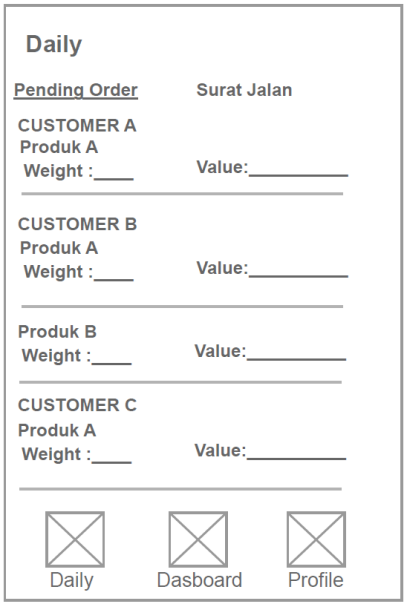
Gambar 4.1 Mockup Login



Gambar 4.2 Mockup Dashboard



Gambar 4.3 Mockup Daily Selling



Gambar 4.4 Mockup Pending Order & Surat Jalan

4.2.2 Perancangan Sistem

Setelah *prototype* awal disepakati antara peneliti dengan pengguna, peneliti mulai mengembangkan *dashboard* penjualan, mulai dari perancangan sistem.

Desain diagram sistem menggunakan metodologi OOAD, menggunakan teknik UML untuk mengembangkan *use case* diagram, *activity diagram, sequence diagram*, *class* diagram, dan spesifikasi *use case* yang mendeskripsikannya secara lengkap

4.2.3 *Use Case* Diagram

Diagram yang digunakan dalam UML untuk menggambarkan hubungan antara pengguna atau aktor dengan sistem disebut *use case* diagram. Skema ini membantu dalam representasi fungsi yang ditawarkan oleh sistem, serta cara di mana pengguna atau pemangku kepentingan terlibat dengan fungsi-fungsi ini dalam mencapai tujuan.

Komponen utama dalam membuat use case diagram meliputi:

a. Aktor (*Actor*)

   Aktor bisa berupa individu, aplikasi atau sistem lain, atau perangkat keras yang berinteraksi dengan sistem.

b. *Use Case*

   Deskripsi fungsi atau layanan yang disediakan oleh sistem untuk memenuhi kebutuhan aktor. *Use case* biasanya menggambarkan satu skenario utama dari interaksi.

c. Sistem (*System Boundary*)

   Sistem *boundary* adalah batasan yang memisahkan sistem yang sedang dianalisis dari lingkungannya. Semua *use case* berada di dalam sistem *boundary*, sedangkan aktor berada di luar.

d. Hubungan (*Relationships*)

Hubungan (*relationships*) adalah garis penghubung antara aktor dengan satu atau beberapa *use case*, serta penghubung antara *use case* itu sendiri. Hubungan ini memberikan informasi tentang interaksi dan ketergantungan antar elemen dalam diagram. Hubungan dalam suatu sistem dapat bermanifestasi dalam berbagai bentuk seperti inklusi (*include*), ekstensi (*extend*), atau asosiasi (*association*).

a. Inklusi (*Include*)

Menunjukkan satu *use case* selalu menggunakan use case lain yang menjadi bagian dari perilakunya.

b. Ekstensi (*Extend*)

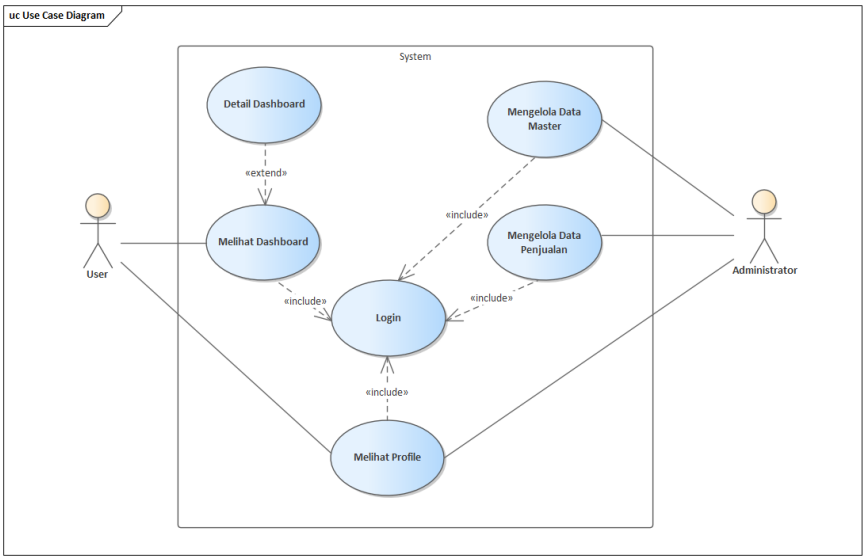
Menunjukkan sebuah *use case* bisa memperluas perilaku *use case* lain di bawah kondisi tertentu.

c. Asosiasi (*Association*)

Menghubungkan aktor dengan *use case* dan menggambarkan ada interaksi diantara keduanya.

Use Case diagram sangat membantu dalam memahami kebutuhan fungsional sistem dari perspektif pengguna. Menyediakan cara yang mudah dimengerti untuk berkomunikasi antara tim pengembang dan *stakeholder*. Membantu dalam mendokumentasikan persyaratan dan skenario penggunaan sistem. Menjadi dasar bagi pengembangan lebih lanjut, termasuk perancangan rinci, pengujian, dan dokumentasi.

Use case diagram merupakan alat efektif dalam memodelkan interaksi antara aktor dengan sistem, memberikan gambaran jelas tentang bagaimana sistem seharusnya bekerja dalam berbagai skenario penggunaan.



Gambar 4.5 Use Case Diagram

Gambar 4.5 menggambarkan aktivitas aktor pada *use case* yang harus tersedia pada sistem, yaitu :

a. Melihat *Dashboard*

Tampilan beranda menampilkan informasi pencapaian penjualan, jumlah *customer* yang sudah melakukan transaksi pembelian, dan pencapaian penagihan piutang.

b. Melihat *Profile*

Halaman profil menampilkan informasi pengguna, yaitu nama, email, atasan, cabang, dan regional.

c. Mengelola Data Penjualan

Adminitrator menggunakan halaman ini untuk melakukan *update* data penjualan yang berhubungan dengan halaman *dashboard*.

d. Mengelola Data Master

Halaman ini digunakan oleh *Administrator* untuk maintain *Branch, Salesman*, dan *User*. *Hanya User aktif yang* punya otorisasi untuk melihat data *dashboard* penjualan.

4.2.4 Spesifikasi *Use Case*

Dalam proses analisis dan desain sistem, spesifikasi *use case* digunakan untuk memastikan bahwa semua kebutuhan pengguna dan sistem terpenuhi. Dokumen ini memberikan deskripsi lengkap tentang *use case* dan mendefinisikan bagaimana sistem harus berperilaku dalam berbagai situasi penggunaan.

Spesifikasi *Use Case* membantu dalam memastikan bahwa semua kebutuhan dan skenario penggunaan dipahami dan didokumentasikan dengan jelas. Spesifikasi tersebut menjadi fasilitas yang efektif untuk komunikasi antar pihak pengembang, pengguna, dan pemangku kepentingan serta menjadi panduan untuk pengembangan, pengujian, dan dokumentasi sistem lebih lanjut.

Tabel 4.1 *Spesifikasi Use Case Login*

Use Case Name	Login	
Actor	User	
Description	User harus login ke dalam aplikasi saat melihat dashboard	
Trigger	Klik button Login	
Pre-Condition	User tidak bisa mengakses untuk melihat dashboard	
Post-Condition	User mempunyai akses ke dalam aplikasi sesuai dengan otorisasi.	
Normal Course	**User**	**Sistem**
	1. Membuka aplikasi	

		2. Menampilkan halam login
	3. Memasukan username dan password	
		4. Memverifikasi username, password, dan mengirimkan kode verifikasi
	5. Menginput kode verifikasi	
		6. Memverifikasi login
		7. Menampilkan halaman utama dashboard
Alternative flows	1. Aplikasi menampikan pesan login tidak berhasil masuk ke dalam aplikasi **2.** User menghubungi menghubungi Administrator	

Tabel 4.1 diatas menjelaskan *use case* untuk *login* pada saat membuka aplikasi. Kode verifikasi dikirimkan melalui akun email yang digunakan pada saat login.

Tabel 4.2 *Spesifikasi Use Case Melihat Dashboard*

Use Case Name	Melihat Dashboard	
Actor	User	
Description	Use Case menjelaskan alur saat melihat dashboard	
Trigger	Login berhasil diverifikasi	
Pre-Condition	Semua User sudah berhasil mengakses ke dalam aplikasi sesuai dengan rolenya	
Post-Condition	Data pencapaian yang tampil pada dashboard dapat dijadikan acuan dalam melakukan push penjualan	
Normal Course	**User**	**Sistem**
	1. User sudah berhasil login ke dalam aplikasi	

		2. Menampilkan dashboard
	3. Klik button Selling	
		4. Menampilkan daftar penjualan per grup produk
	5. Memilih button Customer	
		6. Menampilkan daftar penjualan per customer
	7. Memilih button Collection	
		8. Menampilkan daftar pembayaran per customer
Alternative flows	-	

Tabel 4.2 menjelaskan setelah berhasil *login* ke aplikasi maka tampil *dashboard*. Halaman *dashboard* menampilkan *reward*, *selling*, *customer*, *collection*, dan *jackpot*.

Tabel 4.3 *Spesifikasi Use Case Melihat Profil*

Use Case Name	Melihat Profil	
Actor	User	
Description	Use Case menjelaskan alur dari melihat halaman profile	
Trigger	Login berhasil diverifikasi	
Pre-Condition	All User sudah berhasil mengakses ke dalam aplikasi sesuai dengan rolenya	
Post-Condition	Menampilkan halaman profile user	
Normal Course	**User**	**Sistem**
	1. Melakukan login dengan	

	menggunakan kode verifikasi 2. Memilih button profile	
		3. Menampilkan data profile user
Alternative flows	-	

Tabel 4.3 menjelaskan *use case* setelah *User* berhasil *login* ke dalam aplikasi dan memilih tombol *profile*. Halaman *profile* menampilkan nama, email, atasan, cabang, dan regional.

Tabel 4.4 *Spesifikasi Use Case Mengelola Data Penjualan*

Use Case Name	Mengelola Data Penjualan	
Actor	Administrator	
Description	Use Case menjelaskan proses maintain master data penjualan	
Trigger	Login berhasil diverifikasi	
Pre-Condition	Administrator sudah berhasil mengakses ke dalam aplikasi	
Post-Condition	Menampilkan halaman admin	
Normal Course	**Administrator**	**Sistem**
	1. Melakukan login dengan menggunakan kode verifikasi 2. Memilih button update data	
		3. Menampilkan update data ke dalam database 4. Menampilkan status update data
Alternative flows	1. Aplikasi menampilkan pesan jika proses update semua data gagal. 2. Administrator melakukan update satu per satu	

Tabel 4.4 menjelaskan *use case* setelah *administrator* berhasil *login* ke dalam aplikasi dan memilih tombol *update* data. *Update* data *selling* dan *collection* bisa dilakukan dalam satu kali proses atau masing-masing data.

Tabel 4.5 *Spesifikasi Use Case Mengelola Data Master*

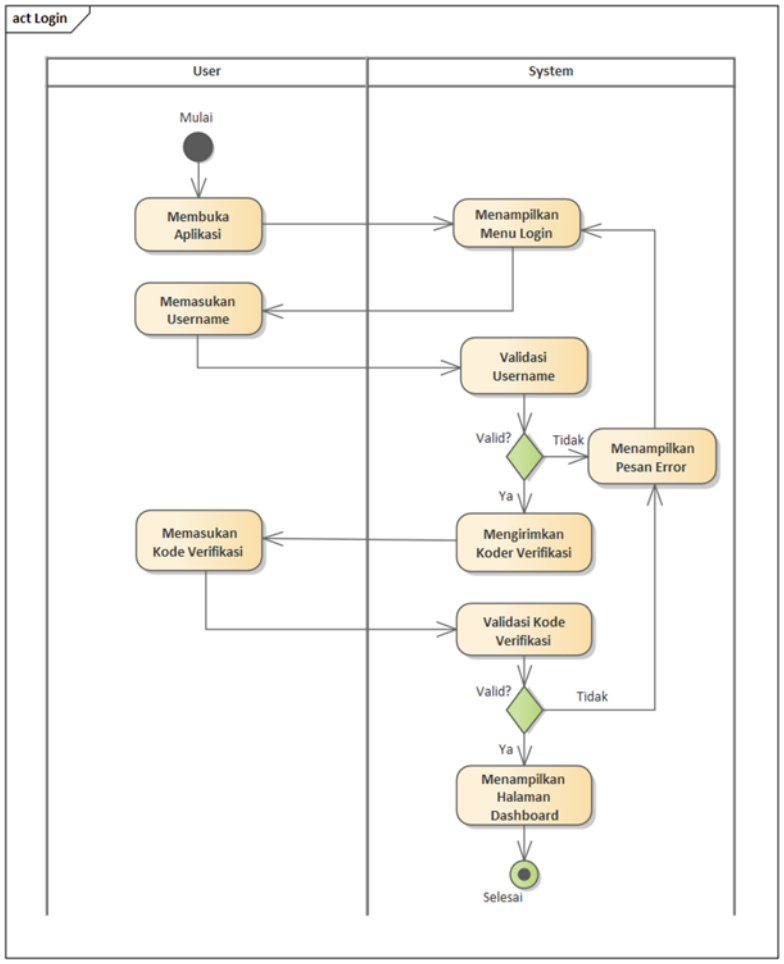
Use Case Name	Mengelola Data Master	
Actor	Administrator	
Description	Use Case menjelaskan proses maintain master user, salesman, dan cabang	
Trigger	Login berhasil diverifikasi	
Pre-Condition	Administrator sudah berhasil mengakses ke dalam aplikasi	
Post-Condition	Menampilkan halaman admin	
Normal Course	**Administrator**	**Sistem**
	1. Melakukan login dengan menggunakan kode verifikasi 2. Memilih button master user	
		3. Menampilkan daftar user
	4. Melakukan CRUD data user	
		5. Update data user 6. Menampilkan pesan data berhasil disimpan
Alternative flows	-	

Tabel 4.5 menjelaskan *use case* setelah *administrator* berhasil *login* ke dalam aplikasi dan memilih tombol *update master user*. *Administrator* bisa melakukan perubahan data pada halaman *master user*.

4.2.5 *Activity* Diagram

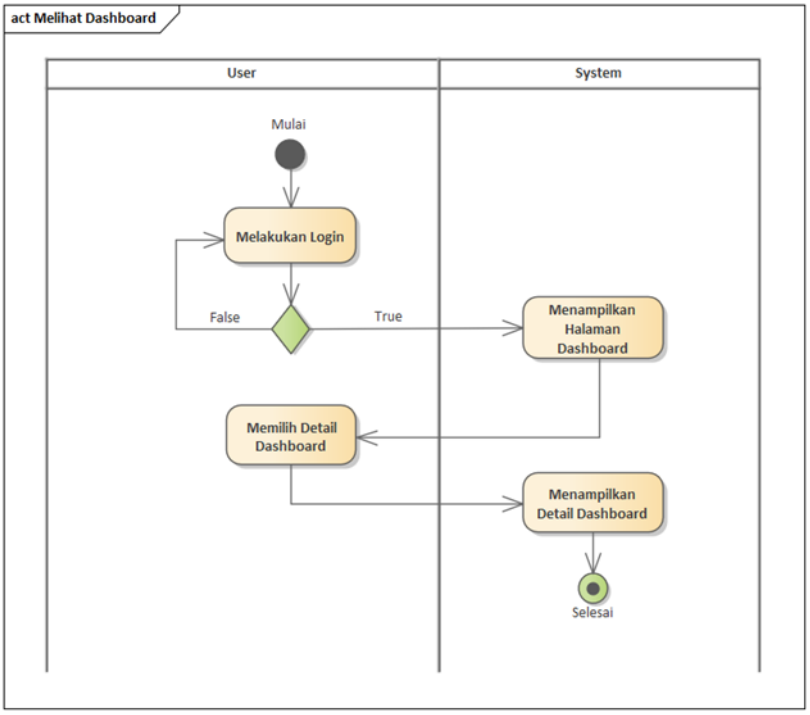
Dalam UML, representasi grafis yang digunakan untuk menunjukkan urutan pekerjaan atau tugas yang dilakukan dalam sistem tertentu disebut *Activity* Diagram. Aktivitas ini menyoroti berbagai kegiatan atau tugas yang terjadi dan urutan mereka dalam suatu proses bisnis atau sistem.

Activity Diagram berguna dalam berbagai tahap pengembangan sistem, termasuk analisis bisnis, perancangan sistem, dan dokumentasi proses. Diagram ini membantu memahami dan mendokumentasikan bagaimana berbagai aktivitas saling berinteraksi dan bagaimana data mengalir di antara mereka.



Gambar 4.6 Activity Diagram Login

Pada Gambar 4.6 menjelasakan aktivitas *login* yang harus dilakukan oleh aktor. Sistem mengecek *username* dan password ke *database* verifikasi data pengguna, jika data sesuai maka sistem akan mengirimkan *one time password* (OTP). Setelah berhasil verifikasi maka akan tampil *dashboard*.



Gambar 4.7 Activity Diagram Melihat Dashboard

Pada Gambar 4.7 menjelaskan saat berhasil *login*, tampilan *dashboard* dan menyesuaikan dengan *role* masing-masing pengguna. *Database* akan membaca *role* tersebut sehingga tampilan *dashboard Salesman* berbeda dengan *Sales Head, Branch Manager, Regional Manager*, dan *National Manager*. *Role* tersebut telah dibuat mengikuti kebutuhan dari setiap pengguna agar data bisa dijadikan sebagai pendukung keputusan.

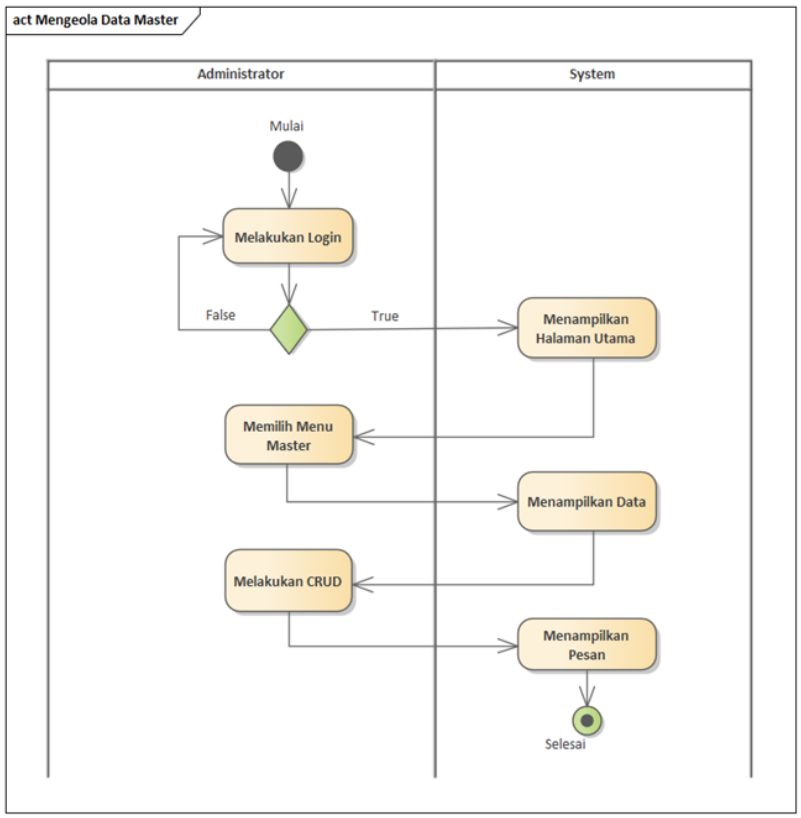
Salesman bisa melihat *detail selling* yang menampilkan penjualan per grup produk. Pada detail c*ustomer* menampilkan jumlah toko yang

sudah ada transaksi *customer*, dan detail *collection* menampilkan pembayaran yang sudah dilakukan oleh *customer*.

Salesman Head dan *Branch Manager* bisa melihat detail *selling* yang menampilkan penjualan per *Salesman*. Pada detail *customer* menampilkan jumlah toko yang sudah ada transaksi penjualan per *Salesman*, dan detail *collection* menampilkan pembayaran yang sudah dilakukan oleh *Salesman*.

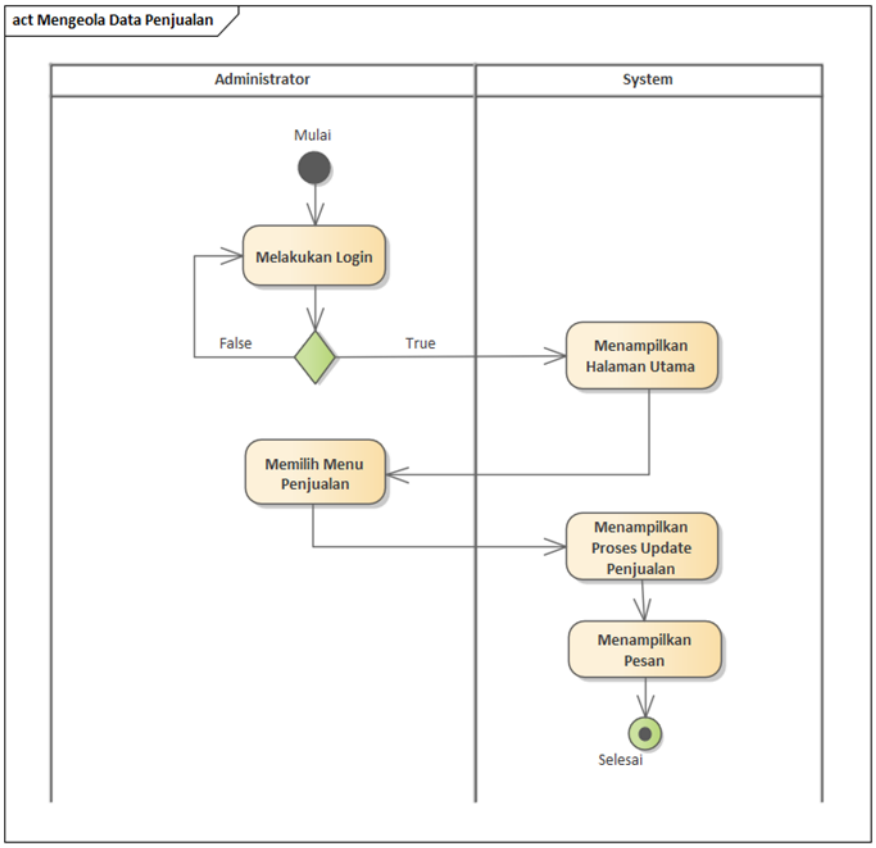
Regional Manager bisa melihat detail *selling* yang menampilkan penjualan per cabang. Pada detail *customer* menampilkan jumlah toko yang sudah ada transaksi penjualan per cabang, dan detail *collection* menampilkan pembayaran yang sudah dilakukan oleh cabang.

National Manager bisa melihat detail *selling* yang menampilkan penjualan per *regional*. Pada detail *customer* menampilkan jumlah toko yang sudah ada transaksi penjualan per *regional*, dan detail *collection* menampilkan pembayaran yang sudah dilakukan oleh *regional*.



Gambar 4.8 Activity Diagram Mengelola Data Master

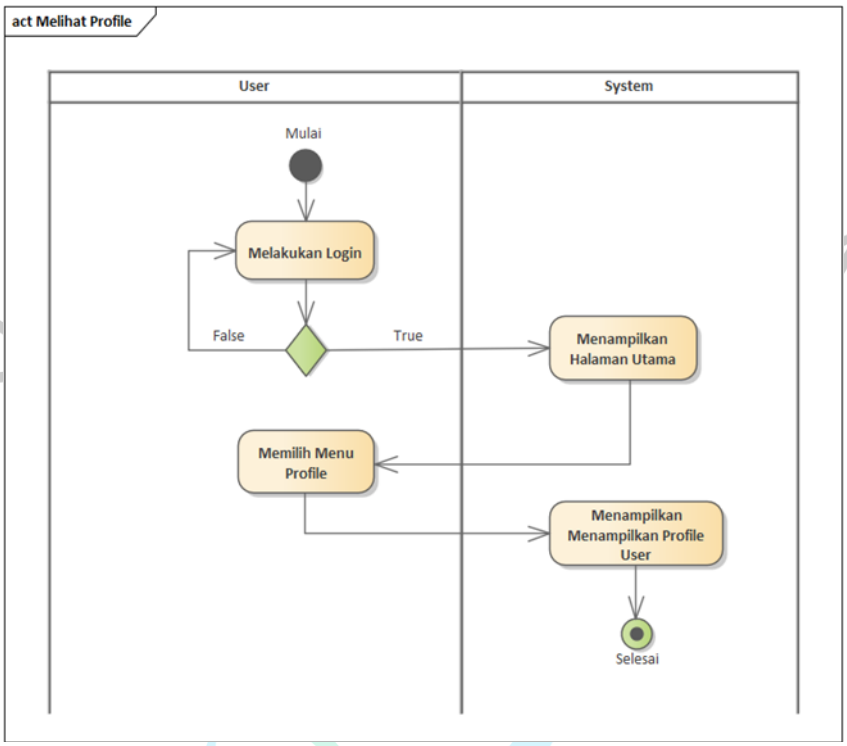
Pada Gambar 4.8 menjelaskan maintain data master hanya bisa dilakukan *Administrator*. Pada tabel *users*, sebagai *Administrator* bisa mendaftarkan dan menonaktifkan pengguna. Hanya pengguna yang sudah didaftarkan pada tabel *users* yang bisa masuk ke dalam aplikasi. Pada tabel *branch*, sebagai *Administrator* bisa menambahkan cabang baru jika ada pemekaran dan maintain regional jika ada perubahan area. Pada tabel *salesman*, sebagai *Administrator* menambahkan data baru dan melakukan perubahan status aktif atau tidak aktif. *Administrator* juga bisa melakukan perubahan saat terjadi rotasi *Salesman*.



Gambar 4.9 Activity Diagram Mengelola Data Penjualan

Pada Gambar 4.9 menjelaskan proses maintain data penjualan hanya bisa dilakukan oleh *Administrator*. Sebagai *Administrator* bisa melakukan *update* data penjualan yang sumbernya dari sistem ERP. Data penjualan dari

sistem ERP akan diproses secara *background* karena datanya transaksi sangat besar. Hal ini dilakukan agar proses data tidak mengganggu transaksi operasional. Data tersebut akan di*upload* oleh *Administrator* ke dalam *database* aplikasi tersebut.



Gambar 4.10 Activity Diagram Melihat Profile

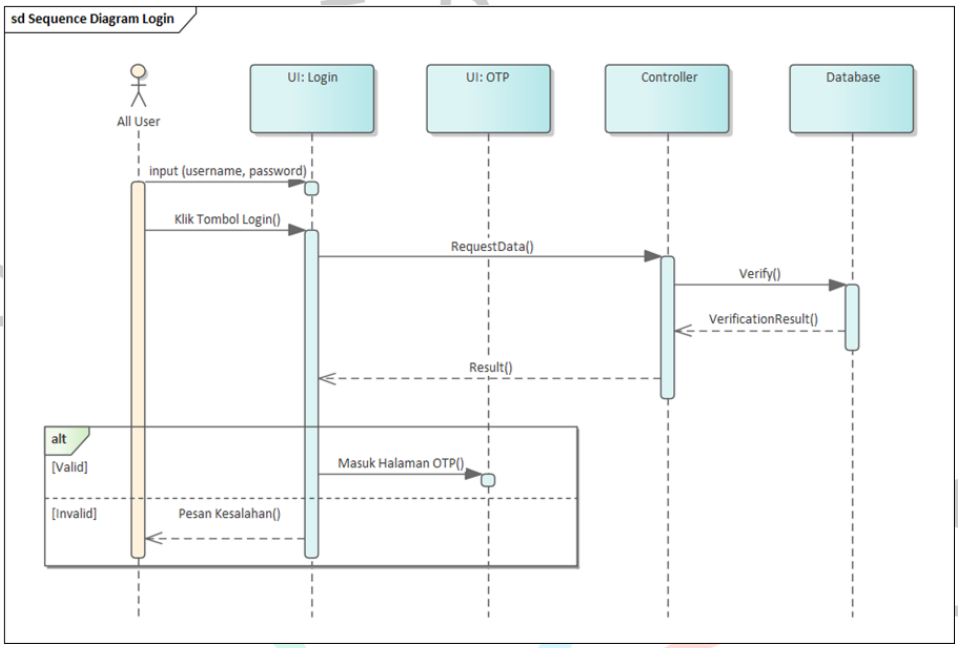
Pada Gambar 4.10 menjelaskan proses melihat halaman *profile* yang bisa dilakukan oleh semua pengguna. Halaman *profile* akan menampilkan kode, nama, cabang, regional dari masing-masing pengguna. Data *profile* penting untuk memastikan atasan yang bersangkutan sudah sesuai dengan kondisi real.

4.2.6 *Sequence* Diagram

Sequence Diagram termasuk dalam klasifikasi UML yang berperan penting dalam mendeskripsikan cara objek atau entitas dalam sistem tertentu berinteraksi satu sama lain melalui urutan pesan di seluruh dimensi temporal. Jenis diagram ini menunjukkan urutan pesan yang dikirim dari

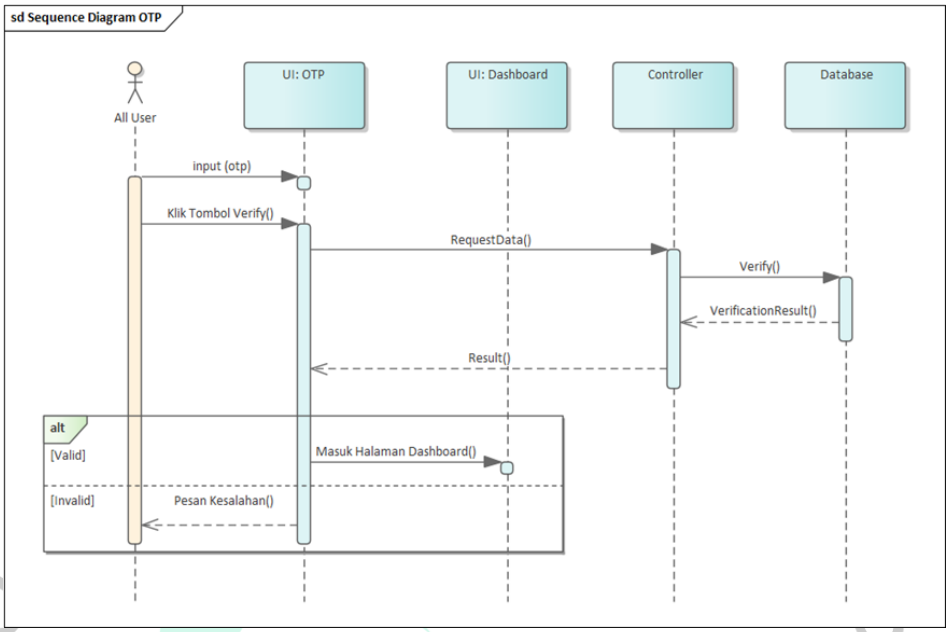
objek ke objek lain untuk memenuhi suatu fungsionalitas atau kasus penggunaan tertentu.

Sequence Diagram sangat berguna dalam perancangan sistem untuk memastikan bahwa semua interaksi antara komponen terdefinisi dengan jelas dan dapat dipahami oleh semua pemangku kepentingan.



Gambar 4.11 Sequence Diagram Login

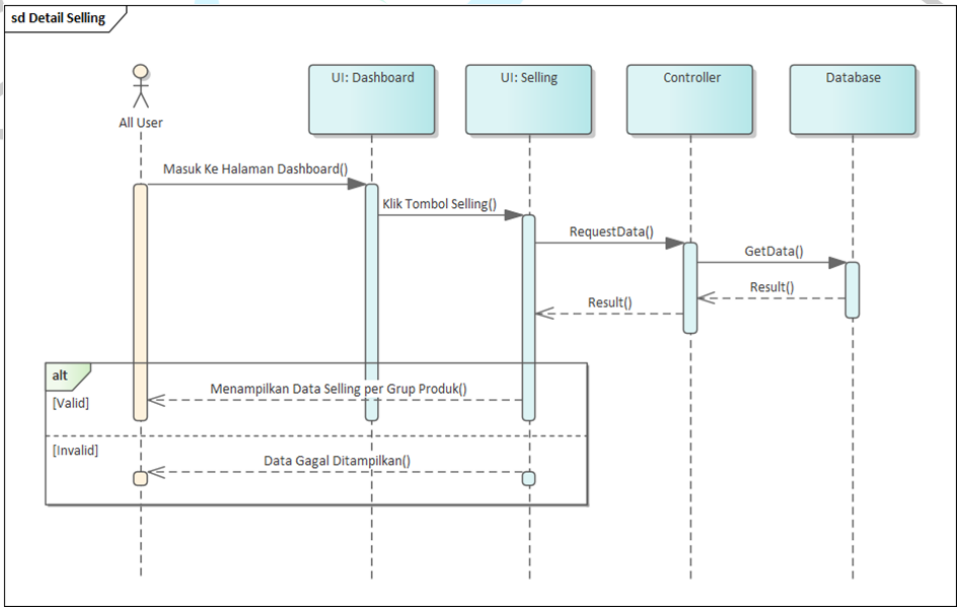
Pada Gambar 4.11 mengilustrasikan interaksi *user* dan sistem melalui *User Interface* (UI) yang mencakup *Login* dengan verifikasi *One Time Password* (OTP). Selanjutnya, sistem memverifikasi ke *database*. Jika gagal diverifikasi, pengguna akan masuk lagi ke halaman *Login*.



Gambar 4.12 Sequence Diagram OTP

Pada Gambar 4.12 mengilustrasikan interaksi *user* dengan sistem melalui UI yang mencakup OTP dan *Dashboard*.

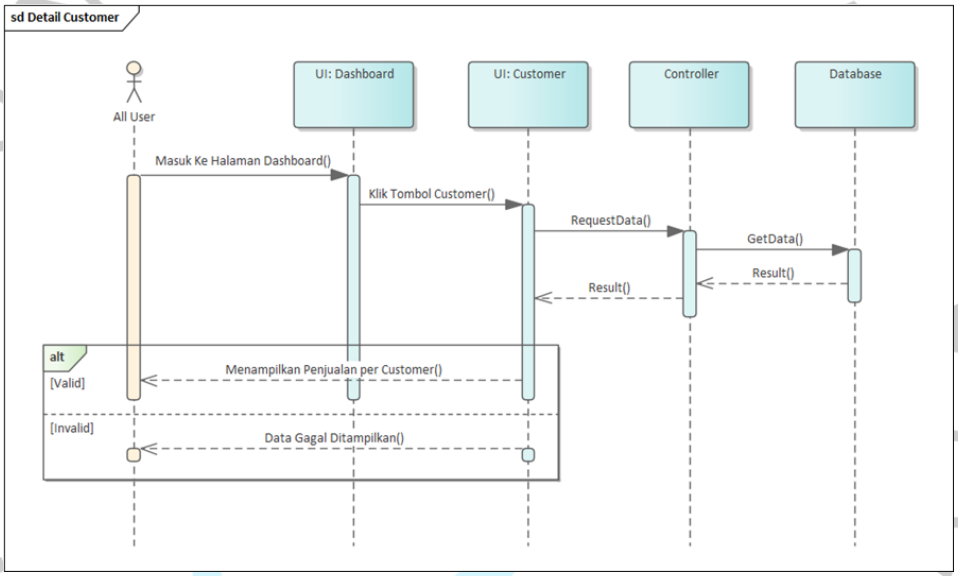
Selanjutnya, setelah *username* dan *password* berhasil diverifikasi, sistem akan merespon dengan mengirimkan OTP. Pengguna akan masuk ke halaman OTP dan harus menginput OTP agar masuk ke dalam aplikasi. Jika OTP gagal verifikasi maka pengguna akan kembali ke halaman OTP.



Gambar 4.13 Sequence Diagram Detail Selling

Pada Gambar 4.13 mengilustrasikan interaksi *user* dan sistem melalui UI yang mencakup *Dashboard* dan detail *Selling*. Awalnya, *user* memulai interaksi dengan masuk halaman *Dashboard*.

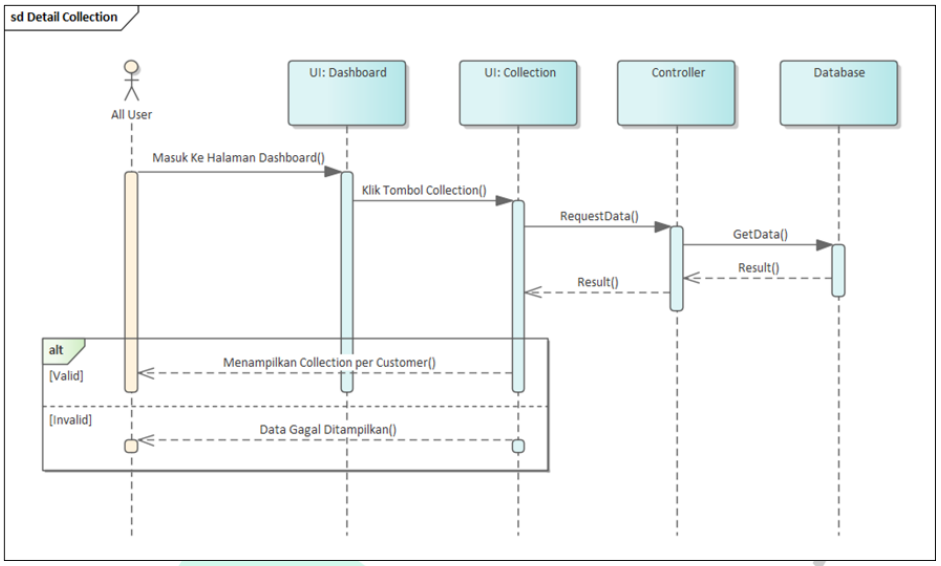
Selanjutnya, *User* diarahkan ke halaman detail *Selling* dan sistem akan merespon dengan menampilkan data *Selling*. Halaman ini berfungsi agar *User* mengetahui kontribusi *customer* yang sudah melakukan transaksi pembelian.



Gambar 4.14 Sequence Diagram Detail Customer

Pada Gambar 4.14 mengilustrasikan interaksi *User* dan sistem melalui UI yang mencakup *Dashboard* dan detail *Customer*. Awalnya, *User* memulai interaksi dengan masuk halaman *Dashboard*.

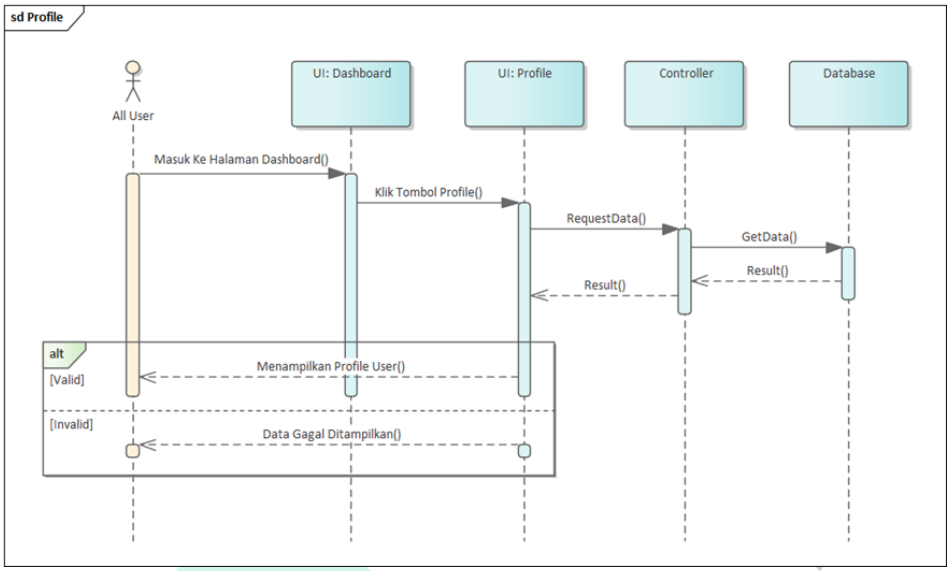
Selanjutnya, *User* diarahkan ke halaman detail *Customer* dan sistem akan merespon dengan menampilkan data *customer*. Halaman ini berfungsi agar *User* mengetahui jumlah *customer* yang sudah melakukan transaksi pembelian. Selain itu, *User* bisa mengetahui *customer* mana saja yang harus di*push* agar terjadi transaksi pembelian.



Gambar 4.15 Sequence Diagram Detail Collection

Pada Gambar 4.15 mengilustrasikan interaksi *User* dan sistem melalui UI yang mencakup *Dashboard* dan detail *Collection*. Awalnya, *User* memulai interaksi dengan masuk halaman *Dashboard*.

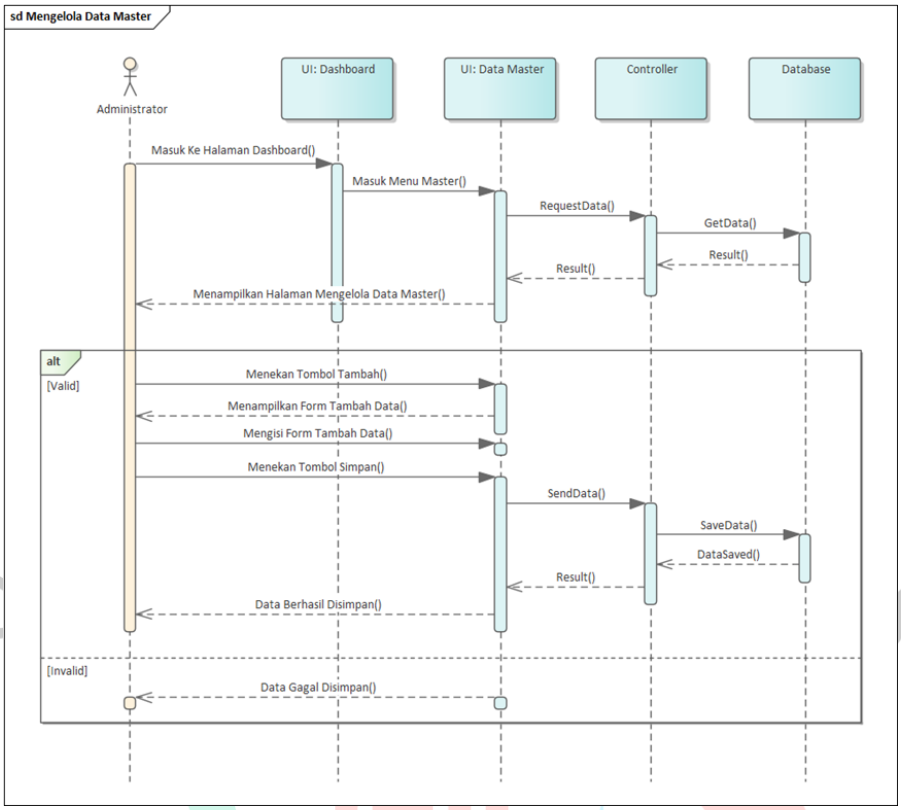
Selanjutnya, *User* diarahkan ke halaman detail *Collection* dan sistem akan merespon dengan menampilkan data pembayaran dari *customer*. Halaman ini berfungsi agar *User* mengetahui *customer* yang sudah melakukan transaksi pembayaran. Selain itu, *User* bisa mengetahui *customer* mana saja yang harus di*push* agar segera melakukan pembayaran terutama untuk faktur yang sudah jatuh tempo.



Gambar 4.16 Sequence Diagram Profile

Pada Gambar 4.16 mengilustrasikan interaksi *User* dengan sistem melalui UI yang mencakup *Dashboard* dan *Profile*. Awalnya, *User* memulai interaksi dengan masuk halaman *Dashboard*.

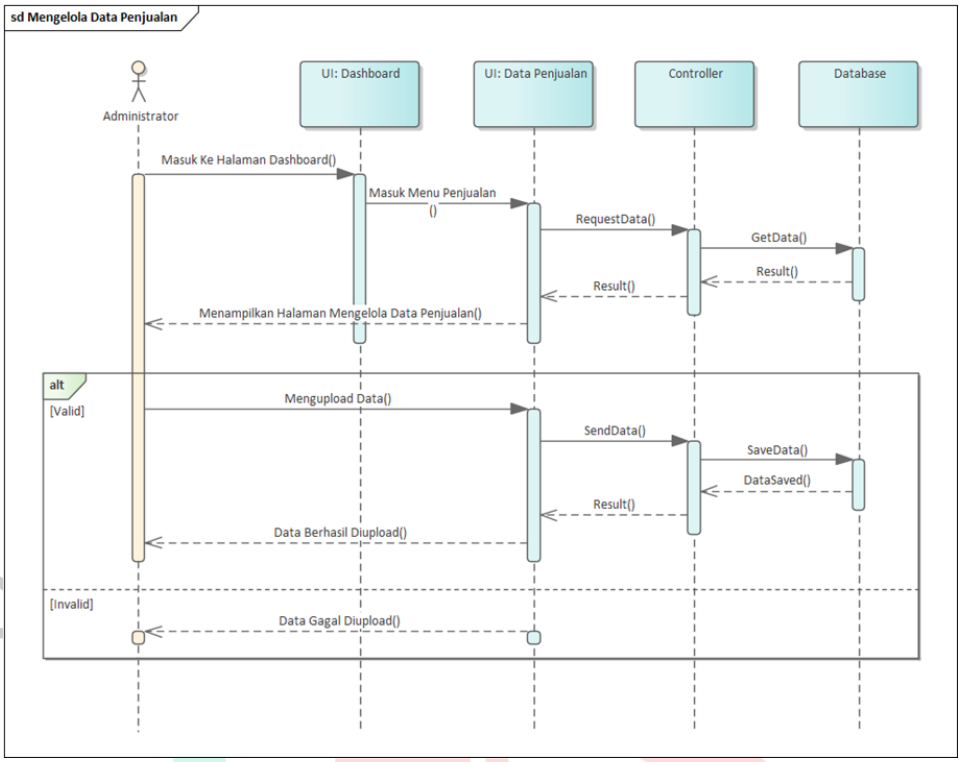
Selanjutnya, *User* diarahkan ke halaman *Profile* dan sistem akan merespon dengan menampilkan data *profile User*. Halaman ini berfungsi untuk memastikan bahwa *User* sudah terdaftar pada struktur *sales team* yang benar.



Gambar 4.17 Sequence Diagram Mengelola Data Master

Pada Gambar 4.17 mengilustrasikan interaksi *Administrator* dan sistem melalui UI yaitu Data *Master*. Awalnya, *Administrator* memulai interaksi dengan masuk halaman *Dashboard*.

Selanjutnya, *Administrator* diarahkan ke halaman Data *Master* dan sistem akan merespon dengan menampilkan data master. Halaman ini berfungsi *Administrator* dalam maintain master *Users*, *Salesman*, dan *Branch*.



Gambar 4.18 Sequence Diagram Mengelola Data Penjualan

Pada Gambar 4.18 mengilustrasikan interaksi *Administrator* dan sistem melalui UI yaitu Data Penjualan. Awalnya, *Administrator* memulai interaksi dengan masuk halaman *Dashboard*.

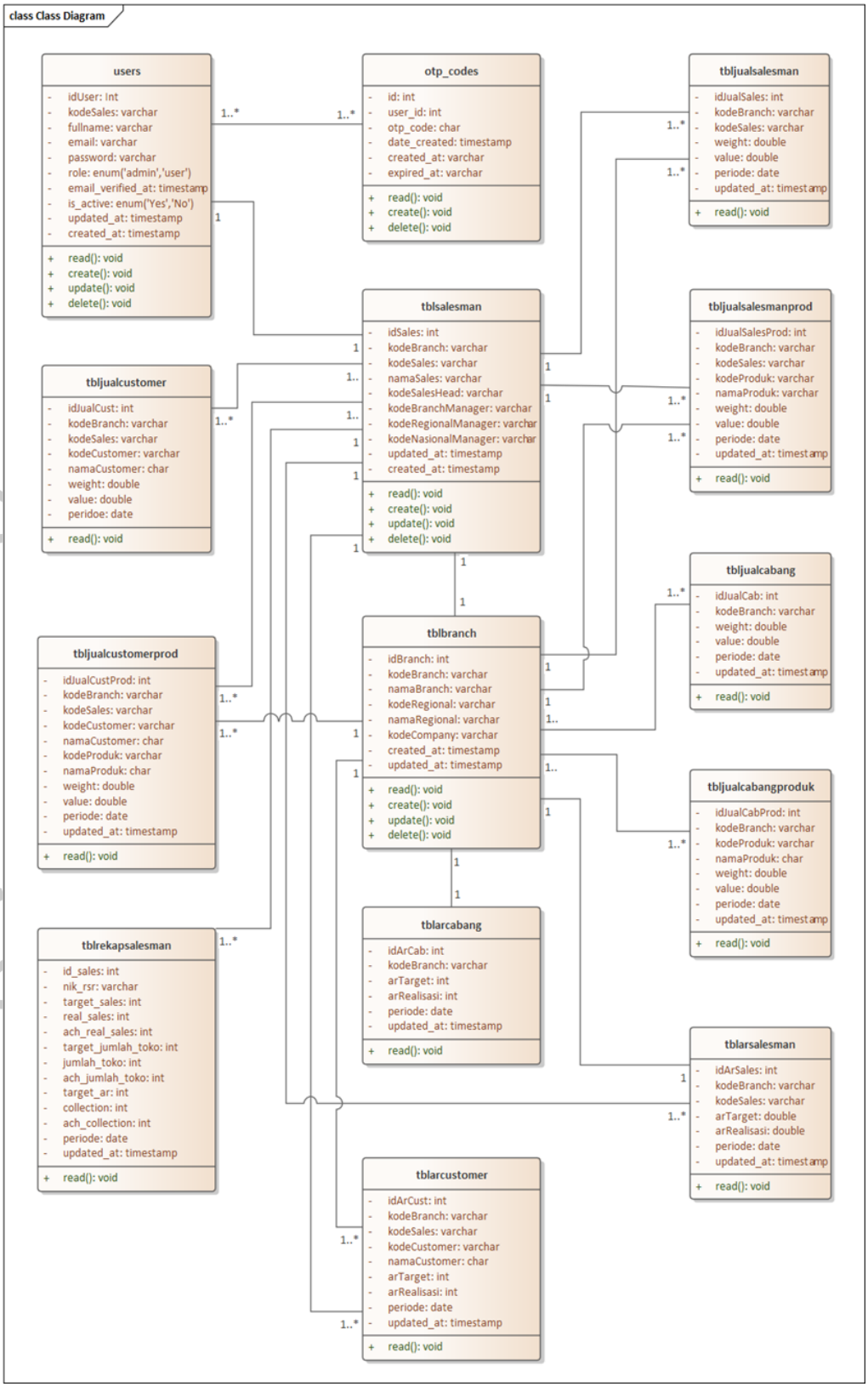
Selanjutnya, *Administrator* diarahkan ke halaman Data Penjualan dan sistem merespon dengan menampilkan menu *upload* data penjualan. Halaman ini berfungsi *Administrator* dalam melakukan *update* data penjualan.

4.2.7 *Class* Diagram

Class Diagram dianggap penting dalam ranah UML, mewakili kategori diagram yang signifikan. Diagram ini berfungsi untuk menggambarkan struktur sistem yang tidak berubah dengan menggambarkan berbagai kelas yang ada di dalamnya, selain menentukan atribut, metode, dan interkoneksi mereka. Dalam bidang pengembangan perangkat lunak, *Class* Diagram telah muncul sebagai instrumen yang

sangat bermanfaat untuk desain dan dokumentasi sistem. *Class* Diagram memainkan peran penting dalam berbagai tahap pengembangan perangkat lunak, sebagai berikut :

1. Membantu pengembang dalam merancang struktur sistem secara menyeluruh sebelum menulis kode. Dengan mendefinisikan class, atribut, dan metode serta hubungan antar *class*, pengembang dapat memahami bagaimana sistem harus dibangun dan berinteraksi.
2. Berfungsi sebagai alat komunikasi yang efektif antar anggota tim pengembang. *Class Diagram* menyediakan visualisasi yang jelas mengenai struktur sistem, sehingga memudahkan anggota tim untuk memahami dan berkolaborasi dalam pengembangan sistem.
3. Sebagai dokumentasi yang membantu pemeliharaan dan pengembangan. Struktur sistem jelas sehingga pengembang baru dapat dengan mudah memahaminya dan melanjutkan perbaikan atau pengembangan.
4. Membantu dalam mengidentifikasi kebutuhan sistem dan merancang solusi yang efektif. Dengan memahami hubungan antar *class* dan bagaimana mereka berinteraksi, pengembang dapat merancang sistem yang lebih modular dan mudah dipelihara.



Gambar 4.19 Class Diagram

4.2.8 Spesifikasi *Table Database*

Spesifikasi *tabel database* adalah dokumen yang mendetail tentang struktur sebuah tabel dalam basis data. Spesifikasi ini sangat penting dalam perancangan sistem informasi karena memberikan panduan tentang bagaimana data disimpan, diorganisir, dan diakses. Dalam perancangan tabel harus menjelaskan nama tabel, deskripsi tabel, nama setiap *field*, *size* ukuran data, *type* data, keterangan, dan kolom yang menjadi *primary key (PK)* atau *foreign key (FK)*.

Nama tabel adalah identifikasi unik yang digunakan untuk membedakan satu tabel dari tabel lainnya dalam *database*. Nama tabel biasanya mencerminkan jenis data yang disimpan di dalamnya. Deskripsi tabel adalah penjelasan singkat tentang tujuan tabel tersebut dan jenis data yang disimpannya. Nama *field* adalah nama dari setiap kolom dalam tabel. Nama-nama ini harus jelas dan deskriptif sehingga memudahkan pengelolaan dan penggunaan data. *Type* data menentukan jenis data yang bisa disimpan dalam suatu kolom. Pemilihan *type* data yang tepat sangat penting untuk efisiensi penyimpanan dan kecepatan akses.

Primary key adalah kolom atau sekumpulan kolom yang secara unik mengidentifikasi setiap baris dalam tabel, harus berbeda dan tidak boleh memiliki nilai NULL. *Foreign key* mengacu pada *primary key* di tabel lain, membantu menjaga integritas referensial dalam database.

1. Tabel *User*

   Nama : *users*

   Deskripsi : Tabel untuk menyimpan daftar user

   PK : idUser

   FK : kodeSales

Nama Field	Type	Size	Keterangan
idUser	int	11	Auto increment
kodeSales	varchar	8	
fullname	varchar	64	

Nama Field	Type	Size	Keterangan
email	varchar	64	
password	varchar	255	
role	enum('admin','user')		
email_verified_at	timestamp		
is_active	enum('Yes','No')		
updated_at	timestamp		
created_at	timestamp		

2. Tabel *One-Time Password* (OTP)

   Nama : otp_codes

   Deskripsi : Tabel untuk menyimpan data OTP

   PK : id

   FK : user_id

Nama Field	Type	Size	Keterangan
id	int	11	Auto increment
user_id	int	11	
otp_code	char	6	
date_created	timestamp	4	
created_at	varchar	16	
expired_at	varchar	16	

3. Tabel *Branch*

   Nama : tblbranch

   Deskripsi : Tabel untuk menyimpan data cabang

   PK : idBranch

   FK : kodeBranch

Nama Field	Type	Size	Keterangan
idBranch	int	11	Auto increment
kodeBranch	varchar	4	
namaBranch	varchar	64	
kodeRegional	varchar	4	
namaRegional	varchar	64	
kodeCompany	varchar	4	

Nama Field	Type	Size	Keterangan
created_at	timestamp		
updated_at	timestamp		

4. Tabel *Salesman*

   Nama : tblsalesman

   Deskripsi : Tabel untuk menyimpan data salesman

   PK : idSales

   FK : kodeSales

Nama Field	Type	Size	Keterangan
idSales	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
namaSales	varchar	64	
kodeSalesHead	varchar	8	
kodeBranchManager	varchar	8	
kodeRegionalManager	varchar	8	
kodeNasionalManager	varchar	8	
updated_at	timestamp		
created_at	timestamp		

5. Tabel Penjualan Cabang

   Nama : tbljualcabang

   Deskripsi : Tabel untuk menyimpan data penjualan cabang

   PK : idJualCab

   FK : kodeBranch

Nama Field	Type	Size	Keterangan
idJualCab	int	11	Auto increment
kodeBranch	varchar	4	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

6. Tabel Penjualan Cabang per Produk

Nama : tbljualcabangproduk

Deskripsi : Tabel untuk menyimpan data penjualan cabang per produk

PK : idJualCabProd

FK : kodeBranch

Nama Field	Type	Size	Keterangan
idJualCabProd	int	11	Auto increment
kodeBranch	varchar	4	
kodeProduk	varchar	6	
namaProduk	char	64	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

7. Tabel Penjualan Customer

Nama : tbljualcustomer

Deskripsi : Tabel untuk menyimpan data penjualan customer

PK : idJualCust

FK : kodeBranch

Nama Field	Type	Size	Keterangan
idJualCust	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
kodeCustomer	varchar	10	
namaCustomer	char	100	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

8. Tabel Penjualan Customer per Produk

   Nama : tbljualcustomerprod

   Deskripsi : Tabel untuk menyimpan data penjualan customer per produk

   PK : idJualCust

   FK : kodeBranch

Nama Field	Type	Size	Keterangan
idJualCustProd	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
kodeCustomer	varchar	10	
namaCustomer	char	100	
kodeProduk	varchar	6	
namaProduk	char	100	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

9. Tabel Penjualan Salesman

   Nama : tbljualsalesman

   Deskripsi : Tabel untuk menyimpan data penjualan salesman

   PK : idJualSales

   FK : kodeBranch, kodeSales

Nama Field	Type	Size	Keterangan
idJualSales	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

10. Tabel Penjualan Salesman per Produk

Nama : tbljualsalesmanprod

Deskripsi : Tabel untuk menyimpan data penjualan salesman per produk

PK : idJualSalesProd

FK : kodeBranch, kodeSales

Nama Field	Type	Size	Keterangan
idJualSalesProd	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
kodeProduk	varchar	6	
namaProduk	char	64	
weight	double		
value	double		
periode	date		
updated_at	timestamp		

11. Tabel Rekap Salesman

Nama : tblrekapsalesman

Deskripsi : Tabel untuk menyimpan data penjualan rekap salesman

PK : id_sales

FK : nik_rsr

Nama Field	Type	Size	Keterangan
id_sales	int	11	Auto increment
nik_rsr	varchar	8	
nama_rsr	varchar	64	
target_sales	int	11	
real_sales	int	11	
ach_real_sales	int	11	
target_jumlah_toko	int	11	
jumlah_toko	int	11	
ach_jumlah_toko	int	11	
target_ar	int	11	
collection	int	11	

Nama Field	Type	Size	Keterangan
ach_collection	int	11	
periode	date		
updated_at	timestamp		

12. Tabel A/R Cabang

Nama : tblarcabang

Deskripsi : Tabel untuk menyimpan data A/R cabang

PK : idArCab

FK : kodeBranch

Nama Field	Type	Size	Keterangan
idArCab	int	11	Auto increment
kodeBranch	varchar	4	
arTarget	double		
arRealisasi	double		
periode	date		
updated_at	timestamp		

13. Tabel A/R Salesman

Nama : tblarsalesman

Deskripsi : Tabel untuk menyimpan data A/R salesman

PK : idArSales

FK : kodeBranch, kodeSales

Nama Field	Type	Size	Keterangan
idArSales	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
arTarget	double		
arRealisasi	double		
periode	date		
updated_at	timestamp		

14. Tabel A/R Customer

Nama : tblarcustomer

Deskripsi : Tabel untuk menyimpan data A/R Customer

PK : idArCust

FK : kodeBranch, kodeSales

Nama Field	Type	Size	Keterangan
idArCust	int	11	Auto increment
kodeBranch	varchar	4	
kodeSales	varchar	8	
kodeCustomer	varchar	10	
namaCustomer	char	100	
arTarget	double		
arRealisasi	double		
periode	date		
updated_at	timestamp		

4.3 Mengembangkan *Prototype*

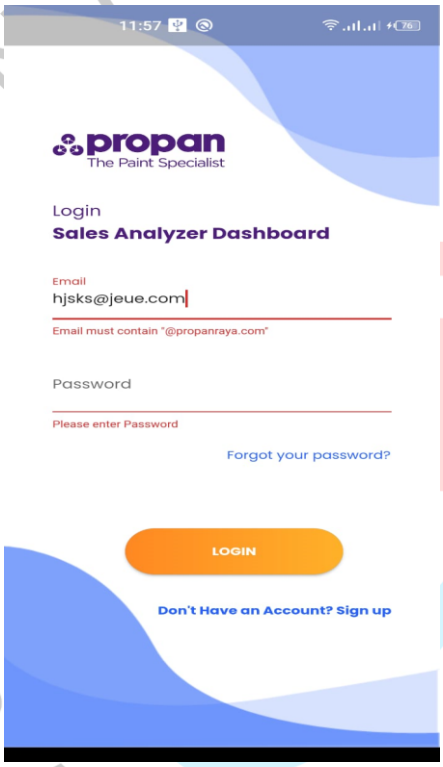
Setelah fitur dan fungsi-fungsi disepakati oleh pengguna, pada tahap ini peneliti mulai mengembangkan *prototype* (*Develop Prototype*). Antarmuka untuk aplikasi berbasis *mobile* adalah proses yang kompleks dan integral dalam pengembangan aplikasi. Desain antarmuka atau *User Interface* (UI) yang baik harus menggabungkan estetika visual dengan fungsionalitas yang intuitif, memastikan pengalaman pengguna atau *User Experience* (UX) yang optimal.

Peneliti menggunakan menggunakan figma untuk mendesain *User Interface* dan *User Experience* (UI/UX). *Dashboard* penjualan dikembangkan menggunakan Flutter karena mendukung pengembangan *platform* Android dan iOS dengan hanya satu basis kode.

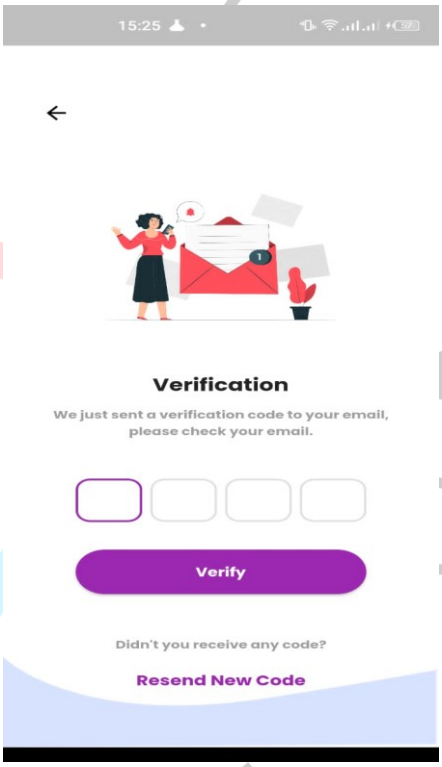
Setelah mendapatkan umpan balik dari pengguna, peneliti mulai mengembangkan antarmuka untuk aplikasi berbasis mobile adalah proses yang kompleks dan integral dalam pengembangan aplikasi. Desain antarmuka atau *User Interface* (UI) yang baik harus menggabungkan

estetika visual dengan fungsionalitas yang intuitif, memastikan pengalaman pengguna atau *User Experience* (UX) yang optimal.

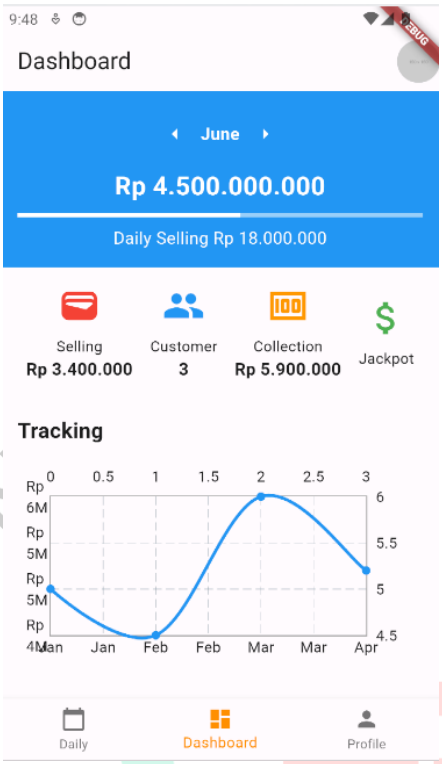
Device mobile termasuk *smartphone* mempunyai ukuran layar yang beragam menuntut desain antarmuka tetap fungsional dan estetis di berbagai perangkat. Desain antarmuka harus mengakomodasi kedua orientasi *portrait* dan *landscape*, perubahan tata letak harus memastikan elemen tetap terlihat dan dapat diakses dengan mudah.



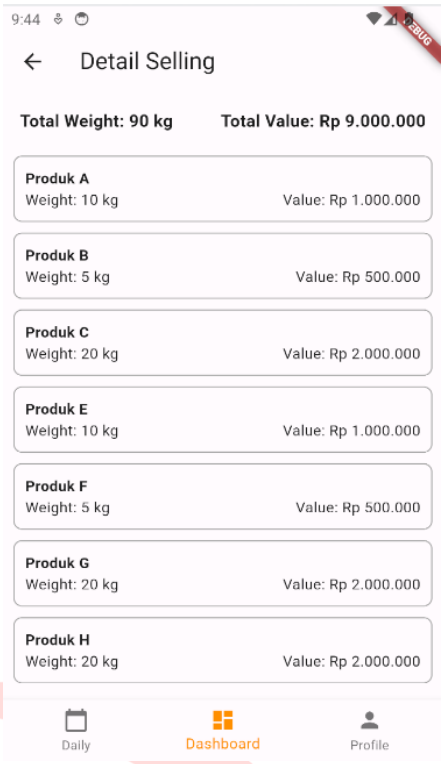
Gambar 4.20 UI Login



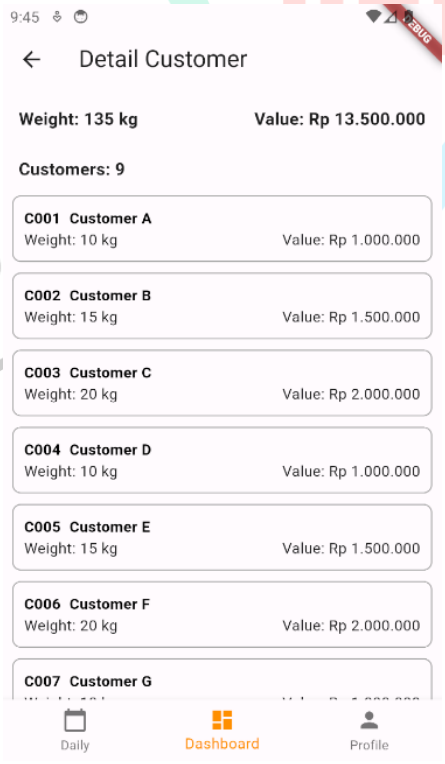
Gambar 4.21 UI OTP



Gambar 4.22 UI Dashboard



Gambar 4.23 UI Detail Selling



Gambar 4.24 UI Detail Customer

Gambar 4.25 UI Detail Collection

Gambar 4.26 UI Pending Order

Gambar 4.27 UI Surat Jalan

4.4 Mengevaluasi *Prototype*

Tahap ini bertujuan untuk mengumpulkan umpan balik dari pengguna untuk menilai dan kegunaan *prototype* yang sudah dikembangkan. *Evaluate prototype* dilakukan dilakukan untuk memastikan apakah *prototype* tersebut telah berhasil memenuhi persyaratan pengguna dan beroperasi dengan benar.

Peneliti meminta beberapa pengguna untuk melakukan pengujian (*Testing*) pada aplikasi *dashboard*. Pengujian mulai dari fitur register, *login*, menu, dan perpindahan dari satu menu ke menu lainnya.

Tabel 4.6 *Daftar Hasil Pengujian*

Masukan (*Input*)	**Proses (*Process*)**	**Keluaran (*Output*)**	**Kesimpulan (*Resume*)**
Klik link text "Sign Up"	Muncul halaman "Sign Up"	Muncul halaman "Sign Up", berisi Full Name, Email, Password	Sesuai
Klik tombol "Sign Up"	Muncul halaman pengisian OTP	Muncul halaman pengisian OTP yang terdiri dari 4 angka	Sesuai
Membuka aplikasi	Muncul halaman login	Muncul halaman login	Sesuai
Input salah "Email" atau "Password"	Muncul warning "Invalid Password"	Muncul warning "Invalid Password"	Sesuai
Input "Email" dan "Password" dengan benar	Muncul halaman dashboard	Muncul halaman dashboard	Sesuai
Klik ikon "Selling"	Menampilkan daftar penjualan produk	Menampilkan daftar penjualan produk	Sesuai
Klik ikon "Customer"	Menampilkan daftar customer	Menampilkan daftar customer	Sesuai
Klik ikon "Collection"	Menampilkan daftar customer yang sudah melakukan pembayaran	Menampilkan daftar customer yang sudah melakukan pembayaran	Sesuai

Masukan (*Input*)	Proses (*Process*)	Keluaran (*Output*)	Kesimpulan (*Resume*)
Klik ikon "Daily"	Muncul dua tab "Pending Order" dan "Surat Jalan"	Muncul dua tab "Pending Order" dan "Surat Jalan"	Sesuai
Klik tab "Pending Order"	Menampilkan daftar pesanan yang belum dikirim	Menampilkan daftar pesanan yang belum dikirim	Sesuai
Klik tab "Surat Jalan"	Menampilkan daftar surat jalan pengiriman hari tersebut	Menampilkan daftar surat jalan pengiriman hari tersebut	Sesuai
Klik ikon "Dashboard"	Muncul halaman dashboard	Muncul halaman dashboard	Sesuai
Klik ikon profil kanan atas	Menampilkan informasi nama, email, menu "Profile", "Change Password", dan "Logout"	Menampilkan informasi nama, email, menu "Profile", "Change Password", dan "Logout"	Sesuai
Klik menu "Change Password"	Muncul halaman "Change Password"	Muncul halaman "Change Password" yang berisi "Old Password", "New Password", "Confirm Passwor""	Sesuai
Klik menu "Change Password" dan "Old Password" salah	Menampilkan warning "Old Password Is Wrong"	Muncul warning "Old Password Is Wrong"	Sesuai
Klik menu "Change Password" dan "Old Password" benar	Muncul warning "Successfully Change Password"	Muncul warning "Successfully Change Password"	Sesuai
Klik menu "Logout"	Keluar dari aplikasi	Keluar dari aplikasi	Sesuai