BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis bertujuan untuk mengidentifikasi permasalahan-permasalahan yang terdapat pada sistem serta menentukan kebutuhan-kebutuhan dari sistem yang akan dibangun. Analisis tersebut meliputi analisis masalah, analisis kebutuhan perangkat lunak (fungsional) dan analisis non fungsional.

Dalam membangun sebuah aplikasi yang sesuai dengan kebutuhan, tentu dibutuhkan analisis terhadap sistem yang sedang berjalan saat ini atau sistem yang berjalan secara umumnya. Tujuannya adalah agar sistem yang akan dibangun tidak keluar dari sistem inti yang sudah berjalan saat ini.

3.1.1 Analisis Masalah

Mengidentifikasi (mengenal) masalah merupakan langkah pertama yang dilakukan dalam tahap analisis sistem. Masalah dapat diidentifikasikan sebagai suatu pertanyaan yang diinginkan untuk dipecahkan. Masalah inilah yang menyebabkan sasaran dari sistem tidak dicapai. Oleh karena itu, langkah pertama yang harus dilakukan pada tahap ini adalah mengidentifikasi terlebih dahulu masalah-masalah yang terjadi.

Berdasarkan hasil wawancara yang telah dilakukan dengan 10 penjual sembako, dan dari hasil kuesioner yang dilakukan dengan pembeli sembako ditemukan beberapa masalah, yaitu:

1. Penjual sembako kesulitan dalam mempromosikan barang yang dijual hanya dengan melakukan promosi dari mulut ke mulut karena tidak adanya media untuk melakukan promosi sehingga tidak semua calon pembeli mengetahui adanya promosi di toko tersebut.
2. Kualitas pelayanan yang diberikan oleh beberapa toko masih terbilang kurang karena tidak adanya jasa antar barang.

3. Pembeli kesulitan dalam mencari lokasi toko sembako yang berada di sekitar mereka.
4. Pembeli mengalami kesulitan dalam berbelanja sembako secara manual karena tidak adanya pemesanan secara *online* dan informasi daftar harga sehingga pembeli harus mendatangi langsung ke setiap toko sembako.

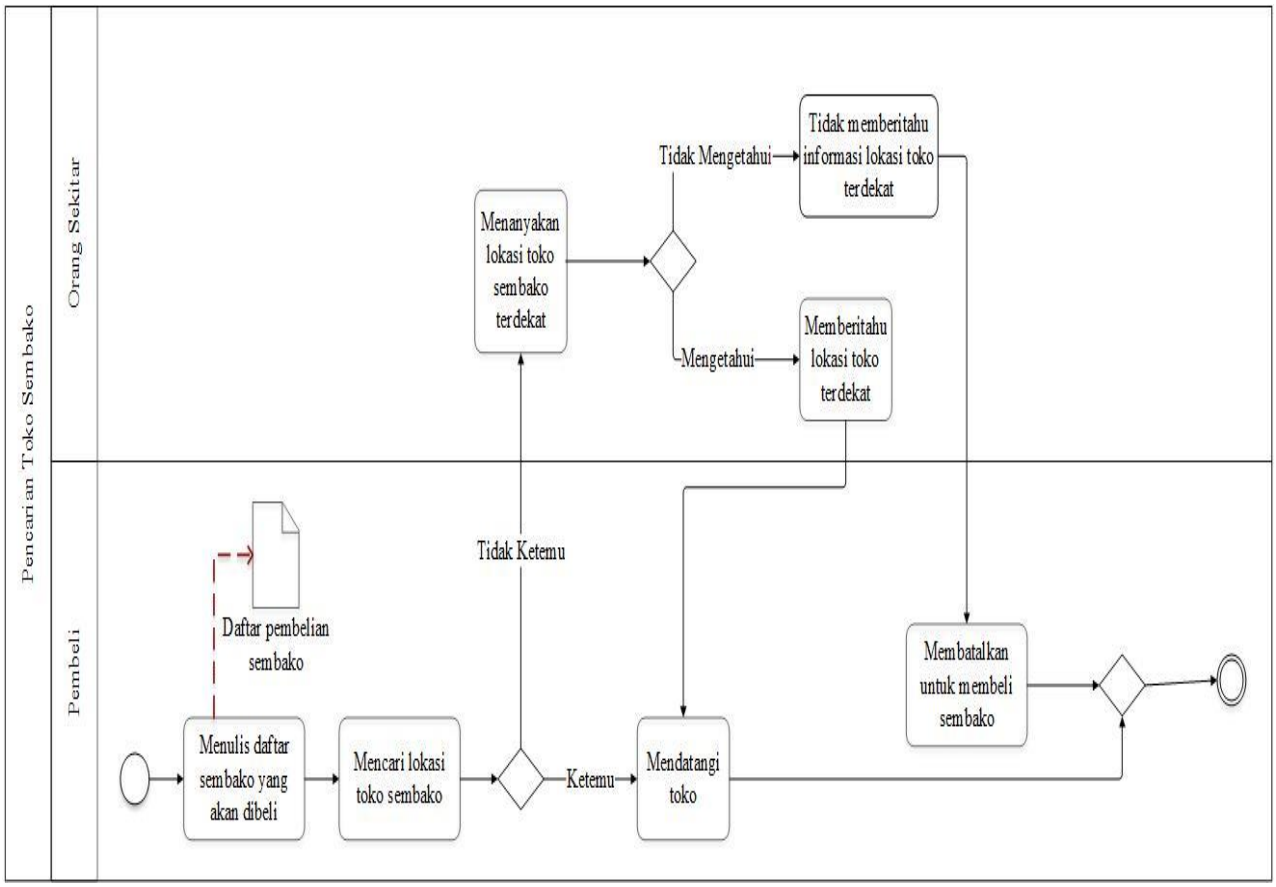
3.1.2 Analisis Prosedur Yang Sedang Berjalan

Analisis prosedur yang sedang berjalan bertujuan untuk mempelajari dan mengevaluasi sistem yang berjalan dan mengetahui seperti apa sistem yang sedang berjalan saat ini:

3.1.2.1 Prosedur Pencarian Toko Sembako

Prosedur pencarian toko sembako yang sedang berjalan adalah sebagai berikut:

1. Pembeli sembako menulis daftar sembako yang akan dibeli.
2. Pembeli sembako mencari toko sembako.
3. Apabila toko tidak ditemukan maka pembeli akan menanyakan lokasi toko sembako terdekat kepada orang di sekitar.
4. Apabila toko ditemukan maka pembeli langsung mendatangi toko.
5. Apabila orang disekitar tidak mengetahui toko sembako terdekat, orang di sekitar tidak memberitahu informasi lokasi toko terdekat.
6. Apabila orang di sekitar mengetahui toko sembako terdekat, orang di sekitar akan memberitahu lokasi toko terdekat dan pembeli akan langsung mendatangi toko.
7. Apabila lokasi toko sembako tidak ditemukan juga maka pembeli akan membatalkan untuk membeli sembako.

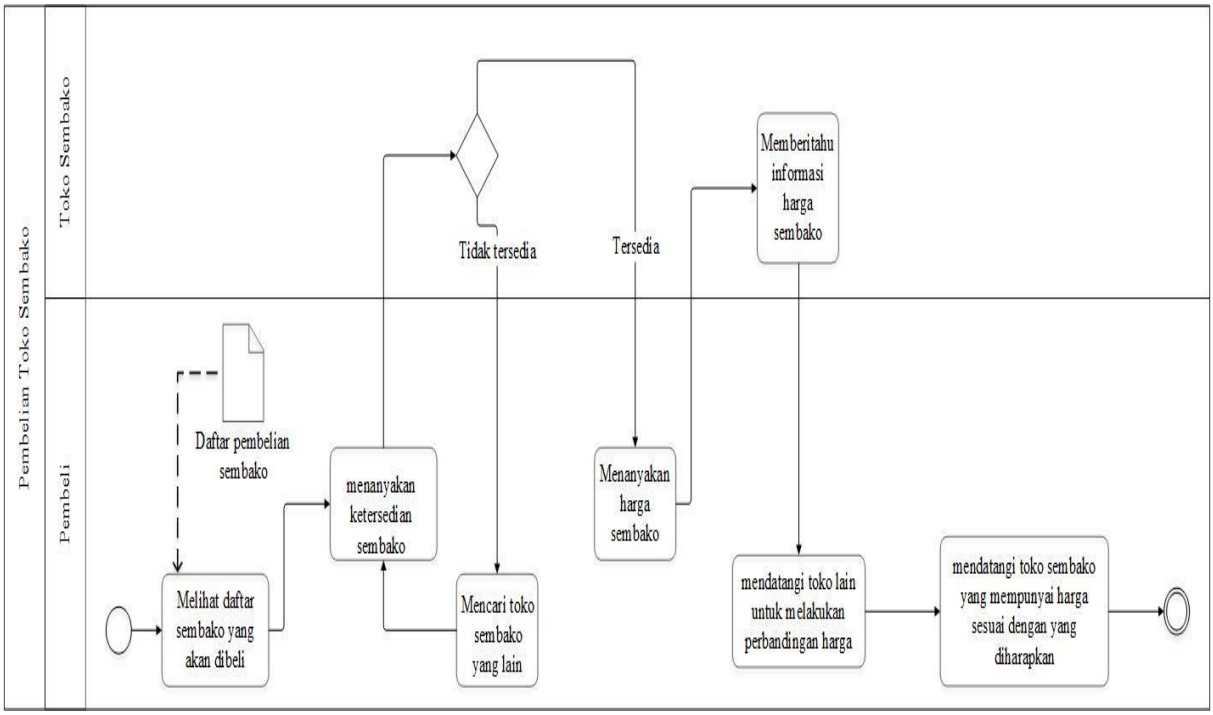


Gambar 3.1 BPMN Pencarian Toko Sembako

3.1.2.2 Prosedur Pembelian Sembako

Prosedur pembelian sembako yang sedang berjalan adalah sebagai berikut:

1. Pembeli melihat daftar sembako yang akan dibeli.
2. Pembeli menanyakan ketersedian sembako yang akan dibeli.
3. Apabila sembako tidak tersedia maka pembeli akan mencari toko sembako yang lain.
4. Apabila sembako tersedia maka pembeli menanyakan harga sembako.
5. Penjual Sembako memberitahu informasi harga sembako yang ada di tokonya.
6. Pembeli mendatangi toko lain untuk melakukan perbandingan harga.
7. Apabila semua harga sembako sudah dibandingkan, maka pembeli akan mendatangi toko sembako yang mempunyai harga sesuai dengan yang diharapkan.



Gambar 3.2 BPMN Pembelian Sembako

3.1.2.3 Prosedur Promosi Toko Sembako

Prosedur promosi toko sembako yang sedang berjalan adalah sebagai berikut:

1. Pembeli mendatangi toko sembako.
2. Penjual sembako menyampaikan secara langsung informasi promosi yang ada di tokonya.

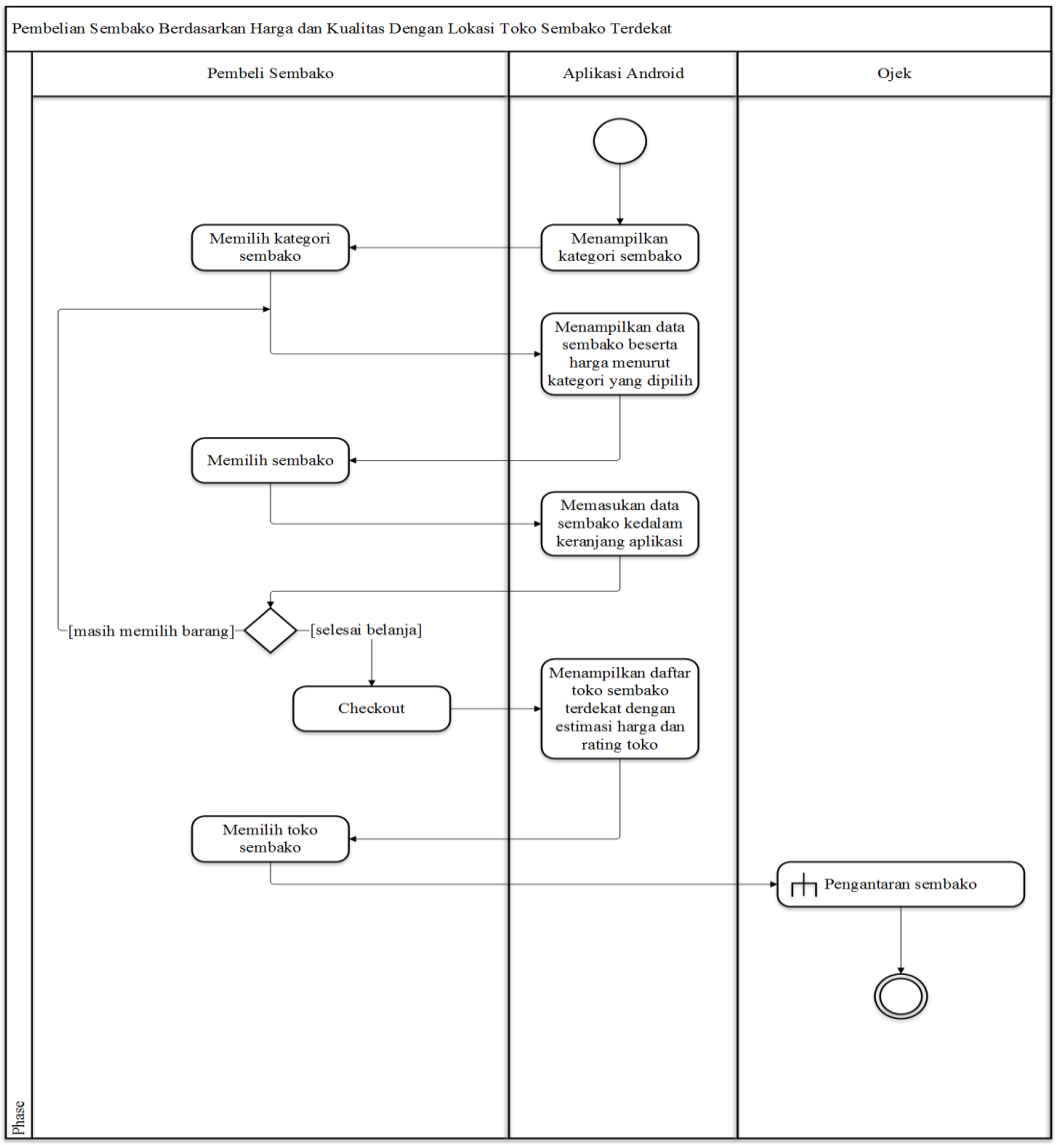


Gambar 3.3 BPMN Promosi Toko Sembako

3.1.3 Analisis Aturan Bisnis

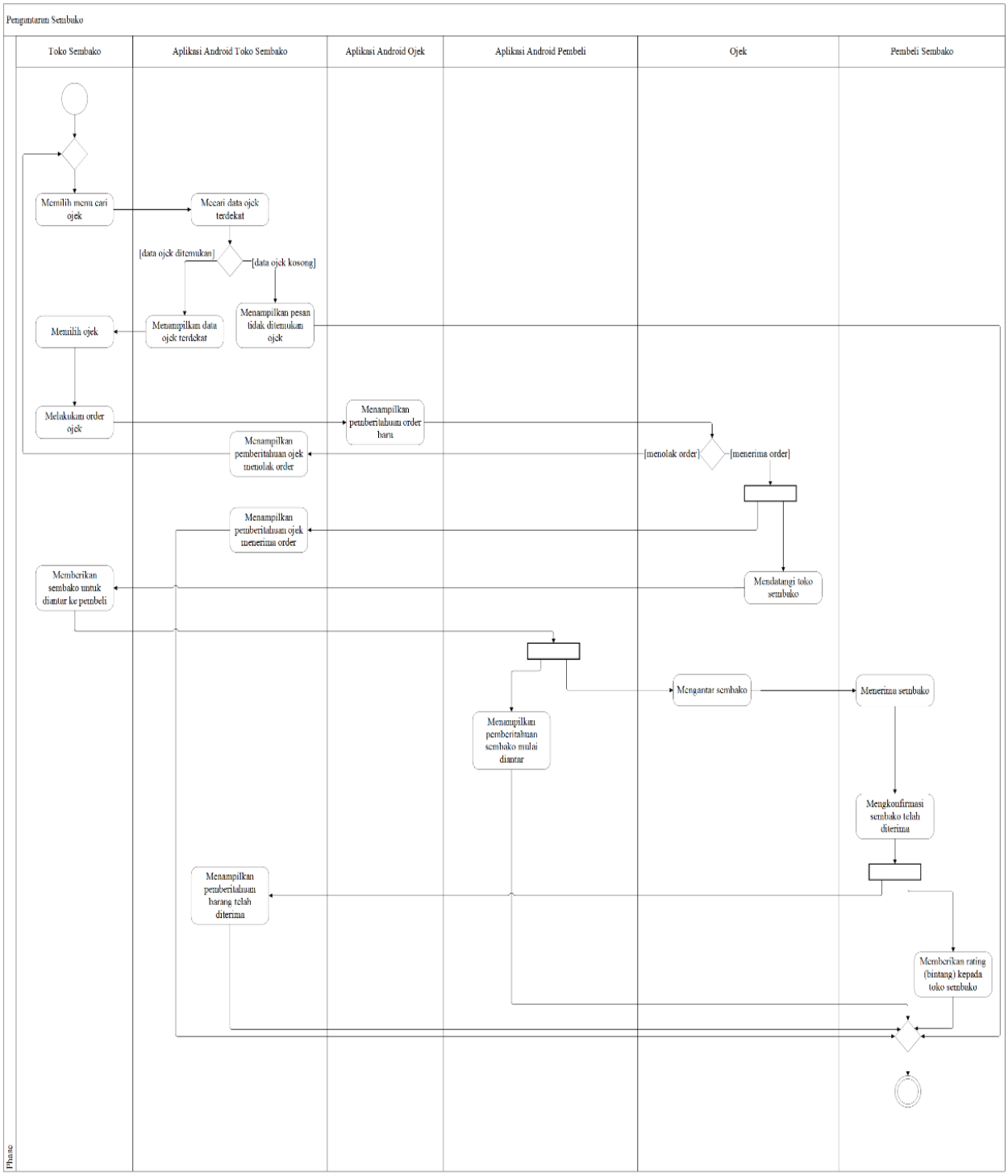
Analisis aturan bisnis yang akan dibangun merupakan penguraian dari suatu sistem yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan-permasalahan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan sebagai perbaikan-perbaikan.

1. Pembelian Sembako Berdasarkan Harga dan Kualitas Dengan Lokasi Toko Sembako Terdekat.



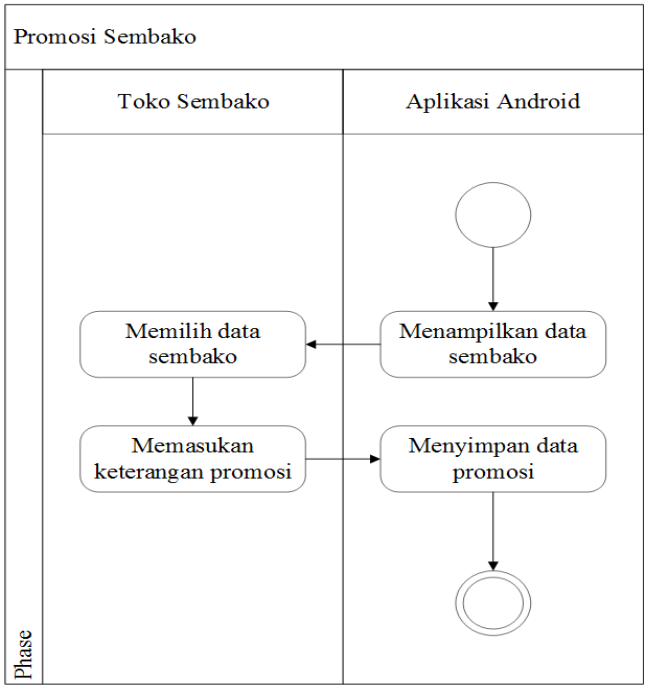
Gambar 3.4 Proses Bisnis Pembelian Sembako Berdasarkan Harga dan Kualitas Dengan Lokasi Toko Sembako Terdekat

2. Pengantaran Sembako



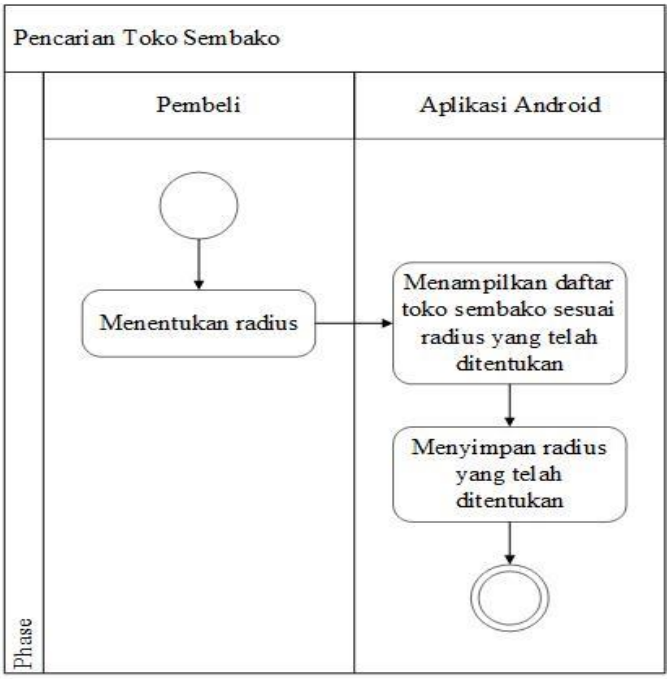
Gambar 3.5 Proses Bisnis Pengantaran Sembako

3. Promosi Sembako



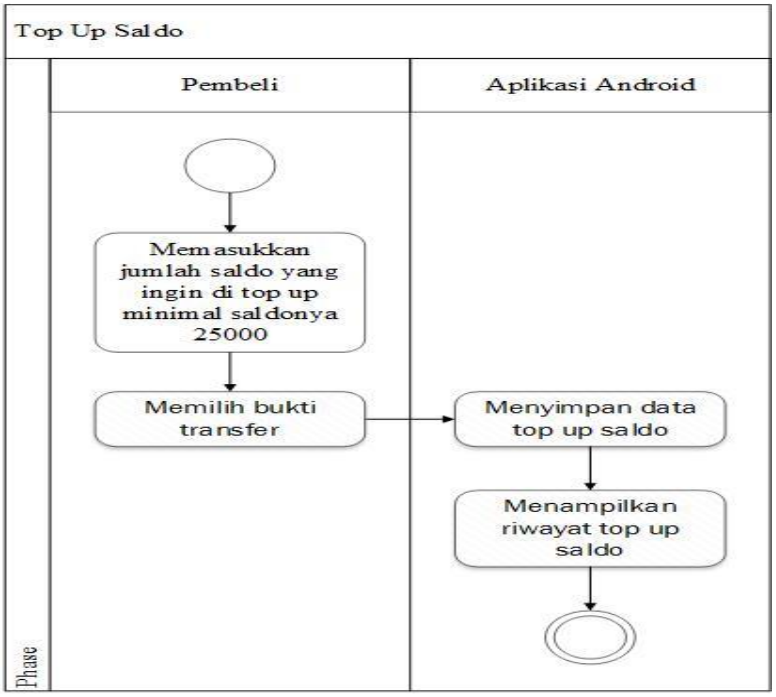
Gambar 3.6 Proses Bisnis Promosi Sembako

4. Pencarian Toko Sembako



Gambar 3.7 Proses Bisnis Pencarian Toko Sembako

5. Top Up Saldo



Gambar 3.8 Proses Bisnis Top Up Saldo

6. Rekomendasi Toko Sembako

Proses bisnis rekomendasi toko sembako yaitu :

a. Pembeli terlebih dahulu menentukan radius.

b. Sistem mencari toko sembako terdekat berdasarkan radius yang telah ditentukan.

c. Kemudian, pembeli memilih kategori sembako yang akan dibeli.

d. Sistem menyimpan daftar sembako yang akan dibeli.

e. Setelah itu, sistem mencari sembako yang ada pada toko sesuai dengan daftar sembako yang akan dibeli.

f. Setelah ditemukan, sistem mengambil data harga lalu menjumlahkannya sesuai dengan total sembako yang ada pada toko tersebut.

Misal : user membeli sembako beras dan jagung

- Toko A harga beras 10000/kg, harga jagung 10000/kg.

10000+10000=20000

Total harga 20000 dengan rating (kualitas) toko 4 bintang.

- Toko B harga beras 8000/kg, harga jagung 8000/kg.

8000+8000=16000

Total harga 16000 dengan rating (kualitas) toko 5 bintang.

Otomatis yang menjadi rekomendasi adalah Toko B karena dilihat dari harga lebih murah dan rating (kualitas) lebih tinggi dari Toko A.

Tetapi Toko A juga akan tetap ditampilkan.

7. Proses Bisnis Pembayaran Ojek

Proses bisnis pembayaran ojek yaitu :

a. Total transaksi pembelian sudah termasuk kedalam pembayaran ojek.

b. Ongkos ojek dihitung sesuai jarak, bila jarak kurang dari 1 km di genapkan menjadi 1 km.

Misal :

- Total pembelian sembako =50.000, jarak pembeli = 3km
- Ongkos ojek menurut jarak = 2.500/km

Jarak*2.500 = 3*2.500 = 7.500

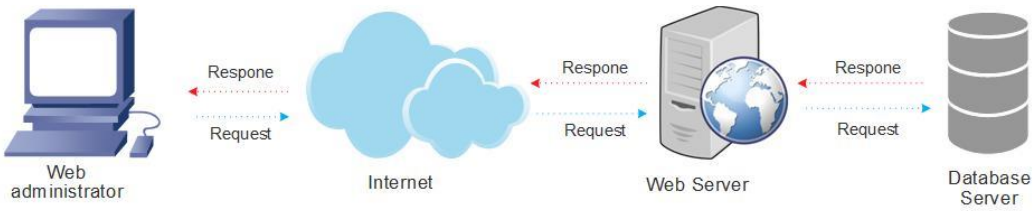
- Jumlah total pembelian 57.500 sudah termasuk ongkos ojek
- Otomatis ongkos ojek masuk ke saldo penjual dan nanti penjual membayar ongkos ojek secara langsung.

3.1.4 Analisis Arsitektur Sistem

Analisis arsitektur sistem bertujuan untuk mengidentifikasi arsitektur yang akan dibangun berdasarkan dua subsistem web dan mobile.

1. *Web*

Platform web adalah salah satu subsistem yang dipilih untuk pembangunan dari perangkat lunak ini. Pengguna perangkat lunak *platform* ini yaitu administrator. Administrator bertugas untuk mengkonfirmasi isi saldo. Berikut adalah Gambar 3.9 Arsitektur Perangkat Lunak pada *Platform Web*. Gambar ini menggambarkan secara keseluruhan arsitektur sistem pada *platform web*.



Gambar 3.9 Arsitektur Perangkat Lunak Pada *Platform Web*

Berikut adalah deskripsi dari Gambar 3.9 Arsitektur Perangkat Lunak pada *Platform Web* :

a. *Web admin* melakukan request data ke *web server*.

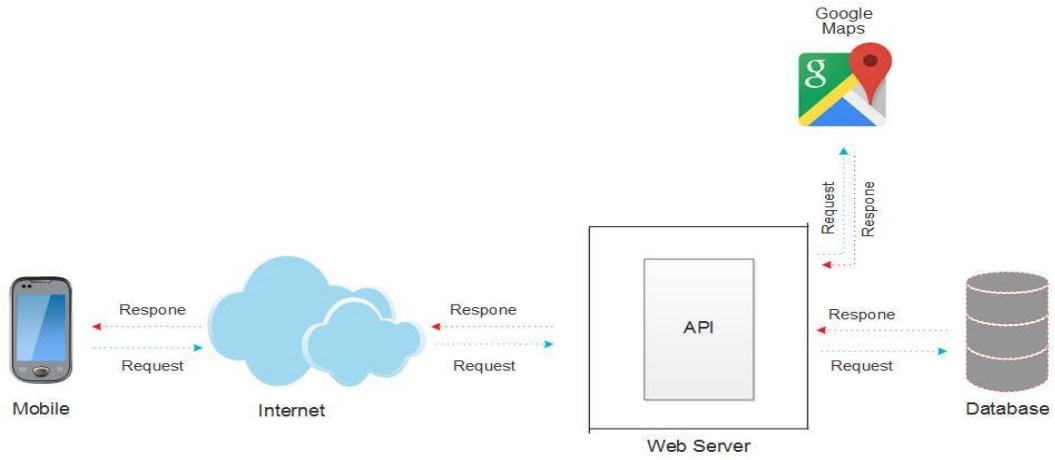
b. *Web server* menerima *request* data dan menentukan jenis *request* yang diminta.

c. Jika *web server* menerima permintaan data maka akan langsung mengambil data yang ada di *database server*.

d. Setelah *web server* menerima data yang diminta, data tersebut akan dikembalikan melalui jaringan internet ke komputer admin.

2. *Android*

Platform mobile adalah salah satu subsistem yang dipilih untuk pembangunan dari perangkat lunak ini. Arsitektur perangkat lunak pada *platform mobile* mengambarkan bagaimana perangkat lunak saling berinteraksi seperti diilustrasikan pada Gambar 3.10 Arsitektur Perangkat Lunak pada *Platform Mobile*. Gambar tersebut menggambarkan keseluruhan arsitektur sistem pada *platform mobile*.



Gambar 3.10 Arsitektur Perangkat Lunak pada Platform Mobile

Berikut adalah deskripsi dari Gambar 3.10 Arsitektur Perangkat Lunak pada *Platform Mobile*:

a. Perangkat *mobile* pengguna melakukan *request* data ke *web server* melalui API.

b. *Web server* menerima *request* data dan menentukan jenis *request* yang diminta.

c. Jika *web server* menerima permintaan lokasi maka *server* memanggil URL *Google Maps* untuk mendapatkan koordinat.

Google Maps mengirim hasil *respone* ke *server*.

e. Jika *web server* menerima permintaan data maka *web server* akan mengambil data yang ada di *database*.

f. Setelah *web server* menerima data yang diminta, data tersebut akan dikembalikan dalam bentuk JSON untuk diproses perangkat *mobile* pengguna.

3.1.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non-fungsional merupakan analisis yang dibutuhkan untuk dapat menentukan spesifikasi dari kebutuhan sistem. Spesifikasi ini meliputi elemen atau perangkat-perangkat yang dibutuhkan untuk sistem yang akan dibangun atau sampai sistem tersebut dapat diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan suatu keluaran yang diinginkan. Kebutuhan non-fungsional terbagi menjadi beberapa analisis yaitu analisis perangkat keras, analisis perangkat lunak dan analisis pengguna.

3.1.5.1 Analisis Kebutuhan Perangkat Keras (*Hardware*)

Analisis kebutuhan perangkat keras merupakan penguraian kebutuhan-kebutuhan non fungsional yang berhubungan dengan spesifikasi perangkat keras dan berhubungan dengan proses pembangunan perangkat lunak.

Tabel 3.1 Spesifikasi Kebutuhan Perangkat Keras Komputer Minimal merupakan spesifikasi perangkat keras komputer minimal yang diperlukan untuk menjalankan perangkat lunak.

Tabel 3.1 Spesifikasi Kebutuhan Perangkat Keras Komputer Minimal

Jenis	Spesifikasi
Prosesor	2 *core* dengan kecepatan 2,3 GHz
RAM	2 GB
Harddisk	120 GB
Monitor	10 Inci

Tabel 3.2 Spesifikasi Kebutuhan Perangkat Keras *Android* Minimal merupakan spesifikasi perangkat keras *android* minimal yang diperlukan untuk menjalankan perangkat lunak.

Tabel 3.2 Spesifikasi Kebutuhan Perangkat Keras *Android* Minimal

Jenis	Spesifikasi
Prosesor	1 *core* dengan kecepatan 830 MHz
RAM	512 GB
Memory internal	2 GB
OS	Android 4.4 Kitkat

3.1.5.2 Analisis Kebutuhan Perangkat Lunak (*Software*)

Tabel 3.3 Spesifikasi Kebutuhan Perangkat Lunak merupakan komponen perangkat lunak yang digunakan untuk membuat aplikasi dan simulasi program.

Tabel 3.3 Spesifikasi Kebutuhan Perangkat Lunak

Jenis	Spesifikasi
Sistem operasi computer	Microsoft Windows 7 Ultimate 64-bit
Sistem operasi android	5.0 Lolipop
Browser	Mozila firefox
Web Server	XAMPP versi 3.2
Java Runtime Environtment (JRE)	Versi 1.8
Java Development Kit (JDK)	Versi 1.8
IntegratedDevelopmentEnvironment (IDE)	Android Studio
TextEditor	Sublime text 3
AndroidDevelopmentTools (ADT)	Versi 23.0.6
SoftwareDevelopmentKit (SDK)	API 19 Kitkat
UML *Tool*	Microsoft visio 2016
MockupTool	Balsamiq Mockup

3.1.5.3 Analisis Pengguna

Suatu aplikasi akan berjalan dengan optimal apabila ditunjang oleh perangkat pikir yang memiliki kemampuan dalam menjalankan aplikasi yang bersangkutan. Perangkat lunak ini akan dugunakan oleh 4 jenis pengguna yaitu penjual sembako, pembeli sembako, ojek dan admin. Karakteristik pengguna yang ada seperti yang ditunjukan Tabel 3.4 Analisis Pengguna Yang Ada pada sebagai berikut:

Tabel 3.4 Analisis Pengguna Yang Ada

No	User	Software yang sering digunakan
1	Penjual Sembako	Aplikasi pengolahan kata (MS Office Word), aplikasi lembar kerja (MS Office Excel), media sosial (WhatsApp, BBM)
2	Pembeli Sembako	Media sosial (WhatsApp, BBM)
3	Ojek	Media sosial (WhatsApp, BBM)
4	Admin	Komputer (Website)

3.1.6 Analisis Kebutuhan Fungsional

Analisis sistem yang dilakukan menggunakan tools UML, adapun tahapan analisis sistem menggunakan UML meliputi *Use Case* Diagram, *Use Case Scenario*, *Activity* Diagram, dan *Class* Diagram. Analisis kebutuhan fungsional pada pembangunan aplikasi pencarian toko sembako ini meliputi 2 *platform* yaitu: *platform mobile android* untuk antar muka ke pengguna penjual sembako, pembeli sembako, dan ojek, dan *platform website* untuk antar muka ke admin.

Berikut ini adalah spesifikasi kebutuhan perangkat lunak fungsional dan non fungsional di *platform mobile android* dan *Platform website*.

Tabel 3.5 Spesifkasi Kebutuhan Perangkat Lunak Fungsional di *Platform Mobile*

Spesifikasi Kebutuhan Perangkat Lunak Fungsional di Platform Mobile	
SKPL-F	**Spesifikasi**
001	Sistem menyediakan fasilitas login
002	Sistem menyediakan fasilitas lupa password
003	Sistem menyediakan fasilitas pendaftaran pembeli
004	Sistem menyediakan fasilitas pendaftaran ojek
005	Sistem menyediakan fasilitas pendaftaran penjual
006	Sistem menyediakan fasilitas untuk mencari lokasi toko sembako terdekat
007	Sistem menyediakan fasilitas untuk order sembako
008	Sistem menyediakan fasilitas untuk isi saldo
009	Sistem menyediakan fasilitas untuk mengubah radius pencarian toko sembako
010	Sistem menyediakan fasilitas untuk pembayaran sembako
011	Sistem menyediakan fasilitas untuuk mengkonfirmasi pemesanan sembako
012	Sistem menyediakan fasilitas untukmencari ojek di sekitar
013	Sistem menyediakan fasilitas pemesanan ojek
014	Sistem menyediakan fasilitas konfirmasi pemesanan ojek
015	Sistem menyediakan fasilitas tambah toko
016	Sistem menyediakan fasilitas ubah toko
017	Sistem menyediakan fasilitas tambah data produk
018	Sistem menyediakan fasilitas ubah data produk
019	Sistem menyediakan fasilitas hapus data produk

Tabel 3.6 Spesifikasi Perangkat Lunak Non Fungsional di *Platform Mobile*

Spesifikasi Kebutuhan Perangkat Lunak Non Fungsional di Platform Mobile	
SKPL-NF	**Keterangan**
001	Sistem dapat diakses selama 24 jam
002	Sistem berbasis android
003	Sistem dapat dijalankan dengan versi minimum android 4.4 (Kitkat)
004	Sistem dapat dijalankan optimal di smartphone yang memiliki fitur GPS

Tabel 3.7 Spesifikasi Perangkat Lunak Fungsional di *Platform Website*

Spesifikasi Kebutuhan Perangkat Lunak Fungsional di Platform Website	
SKPL-F	**Keterangan**
001	Sistem menyediakan fasilitas konfirmasi isi saldo
002	Sistem menyediakan fasilitas konfirmasi pencairan

Tabel 3.8 Spesifikasi Perangkat Lunak Non Fungsional di *Platform Website*

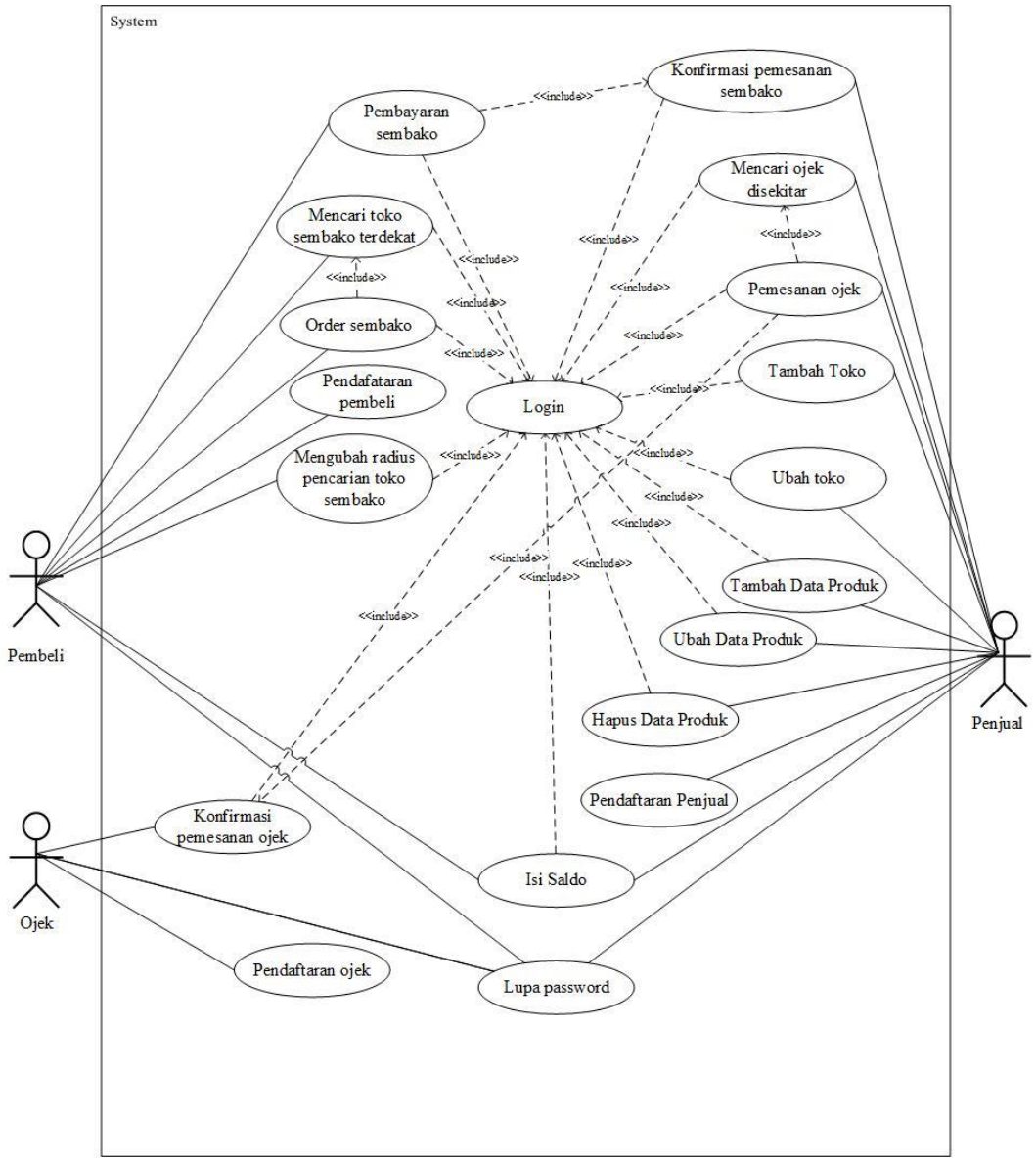
Spesifikasi Kebutuhan Perangkat Lunak Non Fungsional di Platform Website	
SKPL-NF	**Keterangan**
001	Sistem tidak dapat diakses selama 24 jam
002	Sistem berbasis Website
003	Sistem dapat dijalankan dengan prosesor minimum 2 *core* dengan kecepatan 2,3 GHz
004	Sistem dapat dijalankan dengan RAM minimum 2 GB, *Harddisk* minimum 120 GB, Monitor minimum 10 inci

3.1.6.1 Analisis Kebutuhan Fungsional di *Platform Mobile*

Analisis kebutuhan fungsional di platform mobile dilakukan menggunakan *tools* UML, adapun tahapan analisis menggunakan UML meliputi *use case* diagram, *use case scenario*, *activity* diagram, dan *class* diagram. Analisis kebutuhan fungsional di *platform mobile android* akan dijelaskan sebagai berikut.

3.1.6.1.1 *Use Case* Diagram

Use Case Diagram merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktifitas yang terdapat pada sistem. Dari analisis pengguna aplikasi yang ada maka use case diagram di platform mobile dapat dilihat pada Gambar 3.11 *Use Case* Diagram di *Platform Mobile*.



Gambar 3.11 *Use Case* Diagram di *Platform Mobile*

Use case diagram tidak memberikan detail yang cukup untuk menjelaskan bagaimana kebutuhan sistem akan terpenuhi. Untuk itu, *use case scenario* digunakan untuk menunjukan kondisi yang terjdai dan langkah-langkah apa saja yang terlibat pada setiap *usecase*. Use case *scenario* untuk aplikasi yang akan dibangun akan dijelaskan sebagai berikut :

1. *Use Case Scenario* Login

Use case scenario login menggambarkan langkah-langkah aktor untuk masuk kedalam sistem. *Use case scenario* login dapat dilihat pada Tabel 3.9 *Use Case Scenario* Login.

Tabel 3.9 *Use Case Scenario* Login

Use Case Name	**Login**	
Related Requirements	SKPL-F-001	
Goal In Context	Aktor masuk kedalam sistem.	
Precondition	Aktor berada di halaman login.	
Successful End Condition	Aktor berhasil masuk kedalam sistem.	
Failed End Condition	Aktor gagal masuk kedalam sistem.	
Actors	Pembeli, Penjual, ojek	
Trigger	Aktor akan masuk kedalam sistem.	
Main Flow	Step	Action
	1	Aktor mengisi username dan password.
	2	Sistem melakukan validasi data masukan.
	3	Jika data masukan valid, sistem melakukan pemeriksaan akun
	4	Bila akun sudah terdaftar, sistem akan melakukan pemeriksaan hak akses
	5	Sistem menampilkan halaman utama sesuai dengan level yang dipilih
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan kesalahan
	3.1	Sistem menampilkan pesan akun tidak terdaftar

2. *Use Case Scenario* Lupa Password

Use case scenario lupa password menggambarkan langkah-langkah aktor untuk mengubah password apabila aktor lupa password. *Use case scenario* lupa password bisa dilihat pada Tabel 3.10 *Use Case Scenario* Lupa Password.

Tabel 3.10 Use Case Scenario Lupa Password

Use Case Name	**Lupa Password**
Related Requirements	SKPL-F-002
Goal In Context	Sistem mengirim email kepada aktor.
Precondition	Aktor lupa password.
Successful End Condition	Sistem berhasil mengirim email kepada aktor.
Failed End Condition	Sistem gagal mengirim email kepada aktor.

Actors	Pembeli, Penjual, Ojek	
Trigger	Aktor lupa password	
Main Flow	Step	Action
	1	Aktor mengisi email.
	2	Sistem melakukan validasi email.
	3	Jika email valid, sistem melakukan pemeriksaan email
	4	Bila email sudah terdaftar, sistem mengirim email pemulihan akun.
	5	Sistem menampilkan pesan untuk memeriksa email.
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan tidak valid.
	3.1	Sistem menampilkan pesan kesalahan

3. *Use Case Scenario* Pendaftaran Pembeli

Use case scenario pendaftaran pembeli menggambarkan langkah-langkah pendaftaran pembeli baru. *Use case scenario* pendaftaran pembeli bisa dilihat pada Tabel 3.11 *Use Case Scenario* Pendaftaran Pembeli.

Tabel 3.11 Use Case Scenario Pendaftaran Pembeli

Use Case Name	**Pendaftarn Pembeli**	
Related Requirements	SKPL-F-003	
Goal In Context	Aktor berhasil mendaftar di aplikasi	
Precondition	Aktor belum memiliki akun	
Successful End Condition	Aktor berhasil mendaftar di aplikasi.	
Failed End Condition	Aktor gagal mendaftar di aplikasi.	
Actors	Pembeli	
Trigger	Aktor tidak memiliki akun dan akan mendaftar pada aplikasi sebagai pembeli sembako.	
Main Flow	Step	Action
	1	Aktor memasukan data diri.
	2	Sistem melakukan validasi data masukan.
	3	Jika email valid, sistem mengirim email untuk memvalidasi akun.
	4	Bila berhasil, sistem menampilkan pesan untuk memeriksa email.
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan kesalahan.
	3.1	Sistem menampilkan pesan kesalahan.

4. *Use Case Scenario* Pendaftaran Ojek

Use case scenario pendaftaran ojek menggambarkan langkah-langkah pendaftaran pembeli baru. *Use case scenario* pendaftaran ojek bisa dilihat pada Tabel 3.12 *Use Case Scenario* Pendaftaran Ojek.

Tabel 3.12 *Use Case Scenario* Pendaftaran Ojek

Use Case Name	**Pendaftarn Ojek**	
Related Requirements	SKPL-F-004	
Goal In Context	Aktor berhasil mendaftar di aplikasi	
Precondition	Aktor belum memiliki akun	
Successful End Condition	Aktor berhasil mendaftar di aplikasi.	
Failed End Condition	Aktor gagal mendaftar di aplikasi.	
Actors	Ojek	
Trigger	Aktor tidak memiliki akun dan akan mendaftar pada aplikasi sebagai ojek.	
Main Flow	Step	Action
	1	Aktor memasukan data diri.
	2	Sistem melakukan validasi data masukan.
	3	Jika email valid, sistem mengirim email untuk memvalidasi akun.
	4	Bila berhasil, sistem menampilkan pesan untuk memeriksa email.
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan kesalahan.
	3.1	Sistem menampilkan pesan kesalahan.

5. *Use Case Scenario* Pendaftaran Penjual

Use case scenario pendaftaran penjual menggambarkan langkah-langkah pendaftaran penjual baru. *Use case scenario* pendaftaran penjual bisa dilihat pada Tabel 3.13 *Use Case Scenario* Pendaftaran Penjual.

Tabel 3.13 *Use Case Scenario* Pendaftaran Penjual

Use Case Name	**Pendaftarn Pembeli**
Related Requirements	SKPL-F-005
Goal In Context	Aktor berhasil mendaftar di aplikasi
Precondition	Aktor belum memiliki akun
Successful End Condition	Aktor berhasil mendaftar di aplikasi.
Failed End Condition	Aktor gagal mendaftar di aplikasi.
Actors	Penjual

Trigger	Aktor tidak memiliki akun dan akan mendaftar pada aplikasi sebagai pembeli sembako.	
Main Flow	Step	Action
	1	Aktor memasukan data diri.
	2	Sistem melakukan validasi data masukan.
	3	Jika email valid, sistem mengirim email untuk memvalidasi akun.
	4	Bila berhasil, sistem menampilkan pesan untuk memeriksa email.
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan kesalahan.
	3.1	Sistem menampilkan pesan kesalahan.

6. *Use Case Scenario* Mencari Lokasi Toko Sembako Terdekat

Use case scenario mencari lokasi toko sembako terdekat menggambarkan langkah-langkah pencarian lokasi toko sembako terdekat. *Use case scenario* mencari lokasi toko sembako terdekat bisa dilihat pada Tabel 3.14 *Use Case Scenario* Mencari Lokasi Toko Sembako Terdekat.

Tabel 3.14 *Use Case Scenario* Mencari Lokasi Toko Sembako Terdekat

Use Case Name	**Mencari Lokasi Toko Sembako Terdekat**	
Related Requirements	SKPL-F-006	
Goal In Context	Aktor berhasil mencari toko sembako yang terdekat	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan mencari toko sembako terdekat	
Successful End Condition	Aktor berhasil mendapatkan toko sembako yang terdekat	
Failed End Condition	Aktor gagal mendapatkan toko sembako yang terdekat	
Actors	Pembeli	
Trigger	Aktor akan mencari toko sembako terdekat	
Included Case	Login	
Main Flow	Step	Action
	1	Aktor memasukkan data sembako yang akan dibeli
	2	Sistem mencari toko sembako yang menyediakan sembako dengan jarak yang terdekat
	3	Jika ditemukan, sistem menampilkan daftar toko sembako
Extension	Step	Branching Action
	3.1	Sistem tidak menampilkan daftar toko sembako

7. *Use Case Scenario* Order Sembako

Use case scenario order sembako menggambarkan langkah-langkah melakukan order sembako. *Use case scenario* order sembako bisa dilihat pada Tabel 3.15 *Use Case Scenario* Order Sembako.

Tabel 3.15 *Use Case Scenario* Order Sembako

Use Case Name	**Order Sembako**	
Related Requirements	SKPL-F-007	
Goal In Context	Aktor berhasil melakukan pemesanan sembako	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan melakukan pemesanan sembako	
Successful End Condition	Aktor berhasil melakukan pemesanan sembako	
Failed End Condition	Aktor gagal melakukan pemesanan sembako	
Actors	Pembeli	
Trigger	Aktor akan melakukan pemesanan sembako	
Included Case	Mencari Lokasi Toko Sembako Terdekat	
Main Flow	Step	Action
	1 **Include::Mencari Lokasi Toko Sembako Terdekat**	Aktor berhasil mencari toko sembako yang terdekat
	2	Aktor memilih toko sembako
	3	Aktor melakukan pemesanan sembako
	4	Sistem mengkonfirmasi pemesanan
	5	Jika berhasil, sistem menampilkan pesan berhasil
Extension	Step	Branching Action
	4.1	Sistem menampilkan pesan kesalahan

8. *Use Case Scenario* Isi Saldo

Use case scenario isi saldo menggambarkan langkah-langkah melakukan pengisian saldo. *Use case scenario* isi saldo bisa dilihat pada Tabel 3.16 *Use Case Scenario* Isi Saldo.

Tabel 3.16 *Use Case Scenario* Isi Saldo

Use Case Name	**Isi Saldo**
Related Requirements	SKPL-F-008
Goal In Context	Aktor berhasil melakukan pengisian saldo
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan melakukan pengisian saldo
Successful End Condition	Aktor berhasil melakukan pengisian saldo

Failed End Condition	Aktor gagal melakukan pengisian saldo	
Actors	Pembeli, Penjual	
Trigger	Aktor akan melakukan pengisian saldo	
Included Case	-	
Main Flow	Step	Action
	1	Aktor memasukan jumlah saldo yang akan diisi
	2	Sistem melakukan validasi masukan jumlah saldo
	3	Jika valid, sistem menampilkan nomor rekening untuk melakukan transfer
	4	Sistem mengirim data pengisian saldo
	5	Jika berhasil, sistem menampilkan pesan berhasil
Extension	Step	Branching Action
	2.1	Menampilkan pesan saldo yang dimasukkan tidak valid
	4.1	Sistem menampilkan pesan kesalahan

9. *Use Case Scenario* Mengubah Radius Pencarian Toko Sembako

Use case scenario mengubah radius pencarian toko sembako menggambarkan langkah-langkah untuk mengubah radius pencarian toko sembako. *Use case scenario* mengubah radius pencarian toko sembako bisa dilihat pada Tabel 3.17 *Use Case Scenario* Mengubah Radius Pencarian Toko Sembako.

Tabel 3.17 *Use Case Scenario* Mengubah Radius Pencarian Toko Sembako

Use Case Name	**Mengubah Radius Pencarian Toko Sembako**	
Related Requirements	SKPL-F-009	
Goal In Context	Aktor berhasil mengubah radius pencarian toko sembako	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan mengubah radius pencarion toko sembako	
Successful End Condition	Aktor berhasil mengubah radius pencarian toko sembako	
Failed End Condition	Aktor gagal mengubah radius pencarion toko sembako	
Actors	Pembeli	
Trigger	Aktor akan mengubah radius pencarian toko sembako	
Included Case	Login	
Main Flow	Step	Action
	1	Aktor memasukan radius pencarian toko sembako
	2	Sistem menyimpan radius pencarian toko sembako
Extension	Step	Branching Action
	-	-

10. *Use Scenario* Pembayaran Sembako

Use case scenario pembayaran sembako menggambarkan langkah-langkah untuk melakukan pembayaran sembako. *Use case scenario* pembayaran sembako bisa dilihat pada Tabel 3.18 *Use Case Scenario* Pembayaran Sembako.

Tabel 3.18 *Use Case Scenario* Pembayaran Sembako

Use Case Name	**Pembayaran Sembako**	
Related Requirements	SKPL-F-010	
Goal In Context	Aktor berhasil melakukan pembayaran	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan melakukan pembayaran sembako	
Successful End Condition	Aktor berhasil melakukan pembayaran sembako	
Failed End Condition	Aktor gagal melakukan pembayaran sembako	
Actors	Pembeli	
Trigger	Aktor akan melakukan pembayaran sembako	
Included Case	Konfirmasi Pemesanan Sembako	
Main Flow	Step	Action
	1 **Include::Konfirmasi Pemesanan Sembako**	Penjual sembako menerima pemesanan
	2	Aktor mengkonfirmasi pemesanan
	3	Sistem mengirimi konfirmasi pemesanan
Extension	Step	Branching Action
	-	-

11. *Use Case Scenario* Konfirmasi Pemesanan Sembako

Use case scenario konfirmasi pemesanan sembako menggambarkan langkah-langkah untuk melakukan konfirmasi pemesanan sembako. *Use case scenario* konfirmasi pemesanan sembako bisa dilihat pada Tabel 3.19 *Use Case Scenario* Konfirmasi Pemesanan Sembako.

Tabel 3.19 *Use Case Scenario* Konfirmasi Pemesanan Sembako

Use Case Name	**Konfirmasi Pemesanan Sembako**
Related Requirements	SKPL-F-011
Goal In Context	Aktor berhasil mengkonfirmasi pemesanan sembako
Precondition	1. Aktor sudah melakukan login dan masuk kedalam system 2. Aktor menerima pemberitahuan pemesanan baru
Successful End Condition	Aktor berhasil mengkonfirmasi pemesanan sembako
Failed End Condition	Aktor gagal mengkonfirmasi pemesanan sembako
Actors	Penjual

Trigger	Aktor akan mengkonformasi pemesanan	
Included Case	-	
Main Flow	Step	Action
	1	Aktor konfirmasi pemesanan
	2	Sistem menampilkan data sembako yang sudah dibeli
	3	Sistem mengkonfirmasi pemesanan
Extension	Step	Branching Action
	-	-

12. *Use Case Scenario* Mencari Ojek Di Sekitar

Use case scenario mencari ojek di sekitar menggambarkan langkah-langkah untuk mencari ojek di sekitar. *Use case scenario* mencari ojek di sekitar bisa dilihat pada Tabel 3.20 *Use Case Scenario* Mencari Ojek Sekitar.

Tabel 3.20 *Use Case Scenario* Mencari Ojek Di Sekitar

Use Case Name	**Mencari Ojek di Sekitar**	
Related Requirements	SKPL-F-012	
Goal In Context	Aktor berhasil mencari ojek di sekitar	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan mengirim sembako kepada pembeli	
Successful End Condition	Aktor berhasil mencari ojek di sekitar	
Failed End Condition	Aktor gagal mencari oejk di sekitar	
Actors	Penjual	
Trigger	Aktor akan mencari ojek disekitar	
Included Case	-	
Main Flow	Step	Action
	1	Aktor memilih pesanan sembako
	2	Sistem mencari daftar ojek yang berada disekitar
	3	Jika ditemukan, sistem menampilkan daftar ojek yang berada di sekitar
Extension	Step	Branching Action
	2.1	Sistem tidak menampilkan daftar ojek yang berada di sekitar

13. *Use Case Scenario* Pemesanan Ojek

Use case scenario pemesanan ojek menggambarkan langkah-langka pemesanan ojek. *Use case scenario* pemesanan ojek bisa dilihat pada Tabel 3.21 *Use Case Scenario* Pemesanan Ojek.

Tabel 3.21 *Use Case Scenario* Pemesanan Ojek

Use Case Name	**Pemesanan Ojek**

Related Requirements	SKPL-F-013	
Goal In Context	Aktor berhasil mencari ojek di sekitar	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan mengirim sembako kepada pembeli	
Successful End Condition	Aktor berhasil mencari ojek di sekitar	
Failed End Condition	Aktor gagal mencari oejk di sekitar	
Actors	Penjual	
Trigger	Aktor akan mencari ojek disekitar	
Included Case	Mencari Ojek di Sekitar, Konfirmasi Pemesanan Ojek	
Main Flow	Step	Action
	1	Aktor memilih ojek yang akan dipesan
	2 **Include::Mencari Ojek Di Sekitar**	Sistem berhasil menemukan ojek disekitar
	3	Sistem mengirim pemberitahuan pemesanan ojek
	4 **Include::Konfirmasi Pemesanan Ojek**	Ojek menerima pemesanan
Extension	Step	Branching Action
	-	-

14. *Use Case Scenario* Konfirmasi Pemesanan Ojek

Use case scenario konfirmasi pemesanan ojek menggambarkan langkah-langka konfirmasi pemesanan ojek. *Use case scenario* konfirmasi pemesanan ojek bisa dilihat pada Tabel 3.22 *Use Case Scenario* Konfirmasi Pemesanan Ojek.

Tabel 3.22 *Use Case Scenario* Konfirmasi Pemesanan Ojek

Use Case Name	**Konfirmasi Pemesanan Ojek**	
Related Requirements	SKPL-F-014	
Goal In Context	Aktor berhasil mengkonfirmasi pemesanan ojek	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor menerima pemberitahuan pemesanan baru	
Successful End Condition	Aktor berhasil mengkonfirmasi pemesanan ojek	
Failed End Condition	Aktor gagal mengkonfirmasi pemesanan ojek	
Actors	Ojek	
Trigger	Aktor akan mengkonformasi pemesanan	
Included Case	-	
Main Flow	Step	Action
	1	Aktor konfirmasi pemesanan
	2	Sistem menampilkan data sembako yang akan diantar
	3	Sistem mengirim konfirmasi pemesanan
Extension	Step	Branching Action

	-	-

15. *Use Case Scenario* Tambah Toko

Use case scenario tambah toko menggambarkan langkah-langkah tambah toko. *Use case scenario* tambah toko bisa dilihat pada Tabel 3.23 *Use Case Scenario* Tambah Toko.

Tabel 3.23 *Use Case Scenario* Tambah Toko

Use Case Name	**Tambah Toko**	
Related Requirements	SKPL-F-015	
Goal In Context	Aktor berhasil menambah toko	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan melakukan tambah toko	
Successful End Condition	Aktor berhasil melakukan tambah toko	
Failed End Condition	Aktor gagal melakukan tambah toko	
Actors	Penjual	
Trigger	Aktor akan melakukan tambah toko	
Included Case	Tambah toko	
Main Flow	Step	Action
	1	Aktor memasukkan data toko
	2	Sistem melakukan validasi data masukan
	3	Jika data masukan valid, sistem mengirim data toko ke database
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan data masukan tidak valid

16. *Use Case Scenario* Ubah Toko

Use case scenario ubah toko menggambarkan langkah-langkah mengubah toko. *Use case scenario* ubah toko dapat dilihat pada Tabel 3.24 *Use Case Scenario* Ubah Toko.

Tabel 3.24 *Use Case Scenario* Ubah Toko

Use Case Name	**Ubah Toko**
Related Requirements	SKPL-F-016
Goal In Context	Aktor berhasil mengubah toko
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan melakukan ubah toko
Successful End Condition	Aktor berhasil ubah toko
Failed End Condition	Aktor gagal ubah toko
Actors	Penjual

Trigger	Aktor akan melakukan ubah toko	
Included Case	Ubah toko	
Main Flow	Step	Action
	1	Aktor memasukkan data toko yang akan diubah
	2	Sistem melakukan validasi data masukan
	3	Jika berhasil, sistem mengirim data toko ke database
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan data masukan tidak valid

17. *Use Case Scenario* Tambah Data Produk

Use case scenario tambah data produk menggambarkan langkah-langkah menambahkan data produk. *Use case scenario* tambah data produk dapat dilihat pada Tabel 3.25 *Use Case Scenario* Data Produk.

Tabel 3.25 *Use Case Scenario* Tambah Data Produk

Use Case Name	**Tambah Data Produk**	
Related Requirements	SKPL-F-017	
Goal In Context	Aktor berhasil tambah data produk	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan tambah data produk	
Successful End Condition	Aktor berhasil tambah data produk	
Failed End Condition	Aktor gagal tambah data produk	
Actors	Penjual	
Trigger	Aktor akan tambah data produk	
Included Case	Tambah data produk	
Main Flow	Step	Action
	1	Aktor memasukkan data produk
	2	Sistem melakukan validasi data masukan
	3	Jika valid, sistem mengirim data produk ke database
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan data masukan tidak valid

18. *Use Case Scenario* Ubah Data Produk

Use case scenario ubah data produk menggambarkan langkah-langkah mengubah data produk. *Use case scenario* ubah data produk dapat dilihat pada Tabel 3.26 *Use Case Scenario* Ubah Data Produk.

Tabel 3.26 *Use Case Scenario* Ubah Data Produk

Use Case Name	**Ubah Data Produk**

Related Requirements	SKPL-F-018	
Goal In Context	Aktor berhasil ubah data produk	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan ubah data produk	
Successful End Condition	Aktor berhasil ubah data produk	
Failed End Condition	Aktor gagal ubah data produk	
Actors	Penjual	
Trigger	Aktor akan ubah data produk	
Included Case	Ubah data produk	
Main Flow	Step	Action
	1	Aktor memasukkan data produk yang akan diubah
	2	Sistem melakukan validasi data masukan
	3	Jika valid, sistemm mengirim data produk ke database
	4	Sistem menampilkan pesan berhasil
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan data masukan tidak valid

19. *Use Case Scenario* Hapus Data Produk

Use case scenario hapus data produk menggambarkan langkah-langkah menghapus data produk. *Use case scenario* hapus data produk dapat dilihat pada Tabel 3.27 *Use Case Scenario* Hapus Data Produk.

Tabel 3.27 *Use Case Scenario* Hapus Data Produk

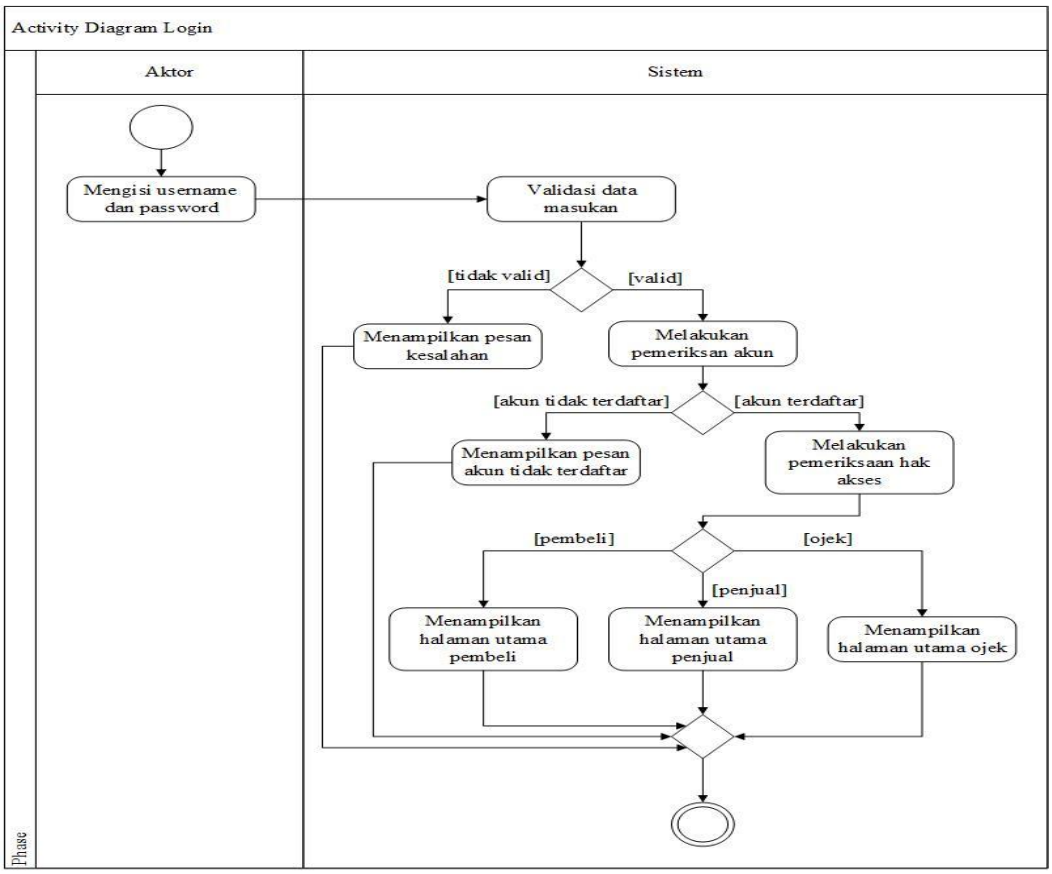
Use Case Name	**Hapus Data Produk**	
Related Requirements	SKPL-F-019	
Goal In Context	Aktor berhasil hapus data produk	
Precondition	1. Aktor sudah melakukan login dan masuk kedalam sistem 2. Aktor akan hapus data produk	
Successful End Condition	Aktor berhasil hapus data produk	
Failed End Condition	Aktor gagal hapus data produk	
Actors	Penjual	
Trigger	Aktor akan hapus data produk	
Included Case	Hapus data produk	
Main Flow	Step	Action
	1	Aktor mengkonfirmasi penghapusan produk
	2	Sistem melakukan validasi data masukan
	3	Jika valid, sistem menghapus data produk
Extension	Step	Branching Action
	2.1	Sistem menampilkan pesan data masukan tidak valid

3.1.6.1.2 *Activity* Diagram

Activity Diagram menggambarkan proses bisnis dan urutan aktifitas dalam sebuah proses, dan digunakan pada proses business modeling untuk memperlihatkan urutan aktifitas proses bisnis. *Activity* Diagram sanagat bermanfaat untuk memahami proses dari sistem secara keseluruhan. *Activity* Diagram dibuat berdasarkan sebuah atau beberapa *use case* pada *use case* diagram. Berikut ini penjelasan dari masing-masing *activity* diagram.

1. *Activity* Diagram Login

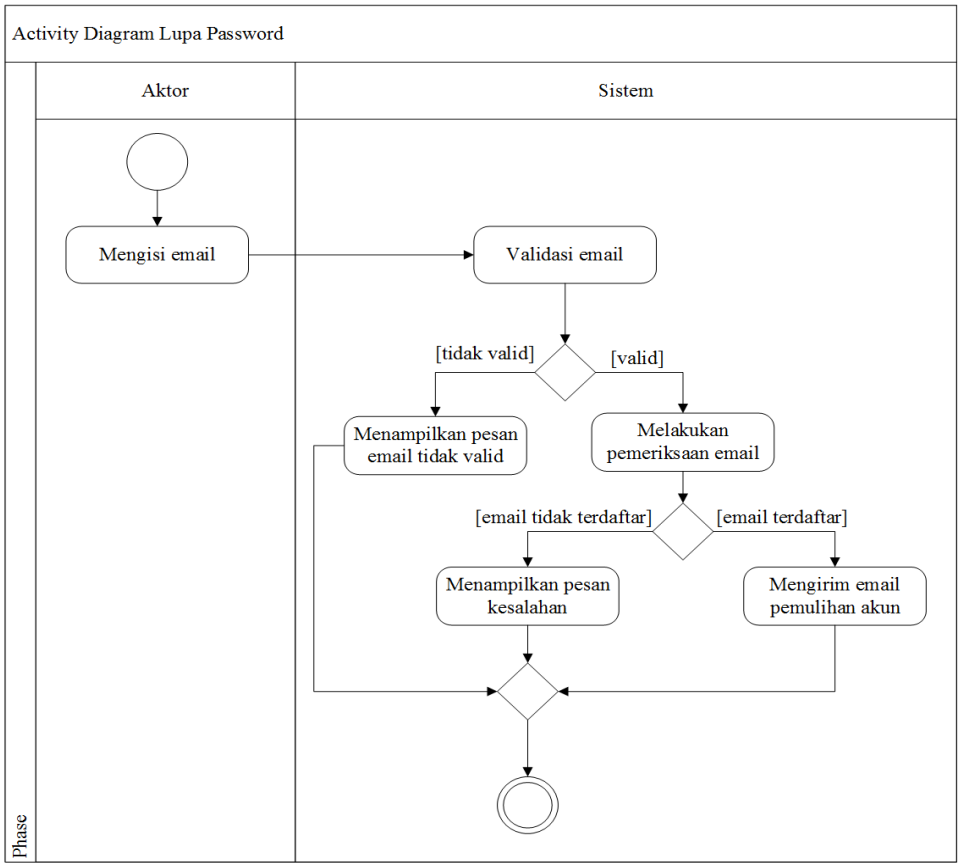
Proses login pada aplikasi dapat dilihat pada Gambar 3.12 *Activity* Diagram Login.



Gambar 3.12 *Activity* Diagram Login

2. *Activity* Diagram Lupa Password

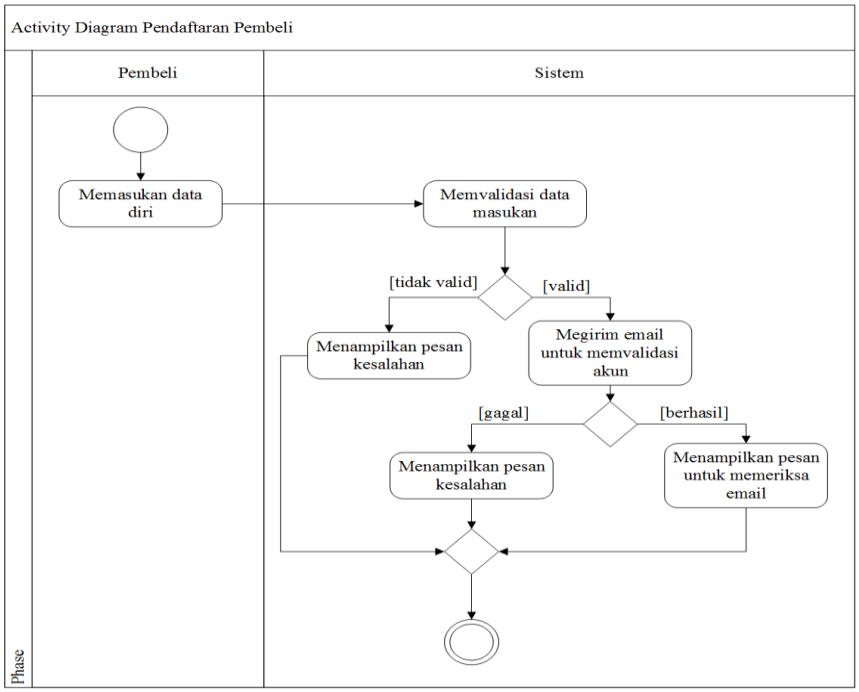
Proses lupa password pada aplikasi dapat dilihat pada Gambar 3.13 *Activity* Diagram Lupa Password.



Gambar 3.13 *Activity* Diagram Lupa Password

3. Activity Diagram Pendaftaran Pembeli

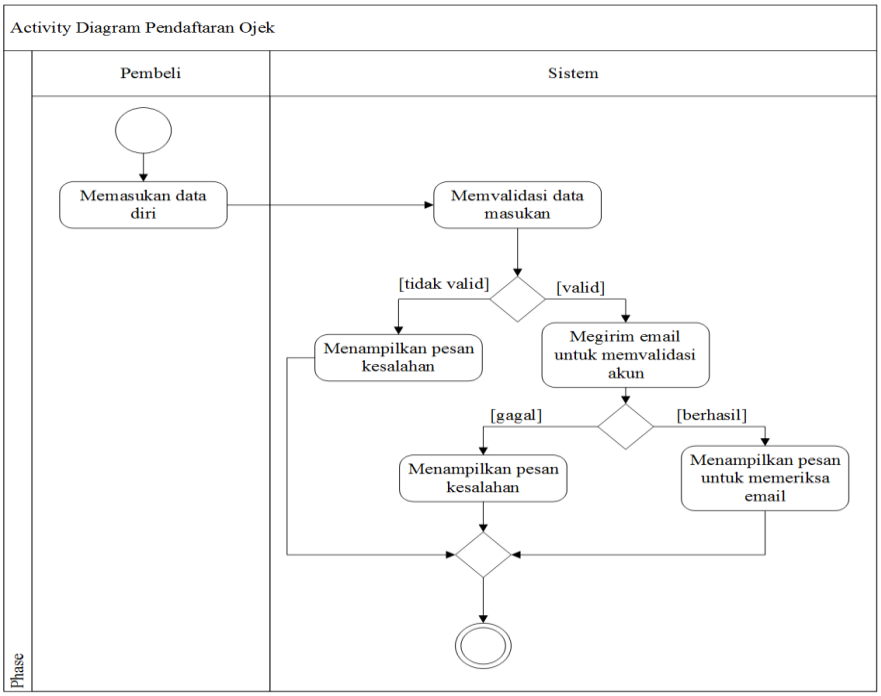
Proses pendaftaran pembeli pada aplikasi dapat dilihat pada Gambar 3.14 *Activity* Diagram Pendaftaran Pembeli.



Gambar 3.14 *Activity* Diagram Pendaftaran Pembeli

4. *Activity* Diagram Pendaftaran Ojek

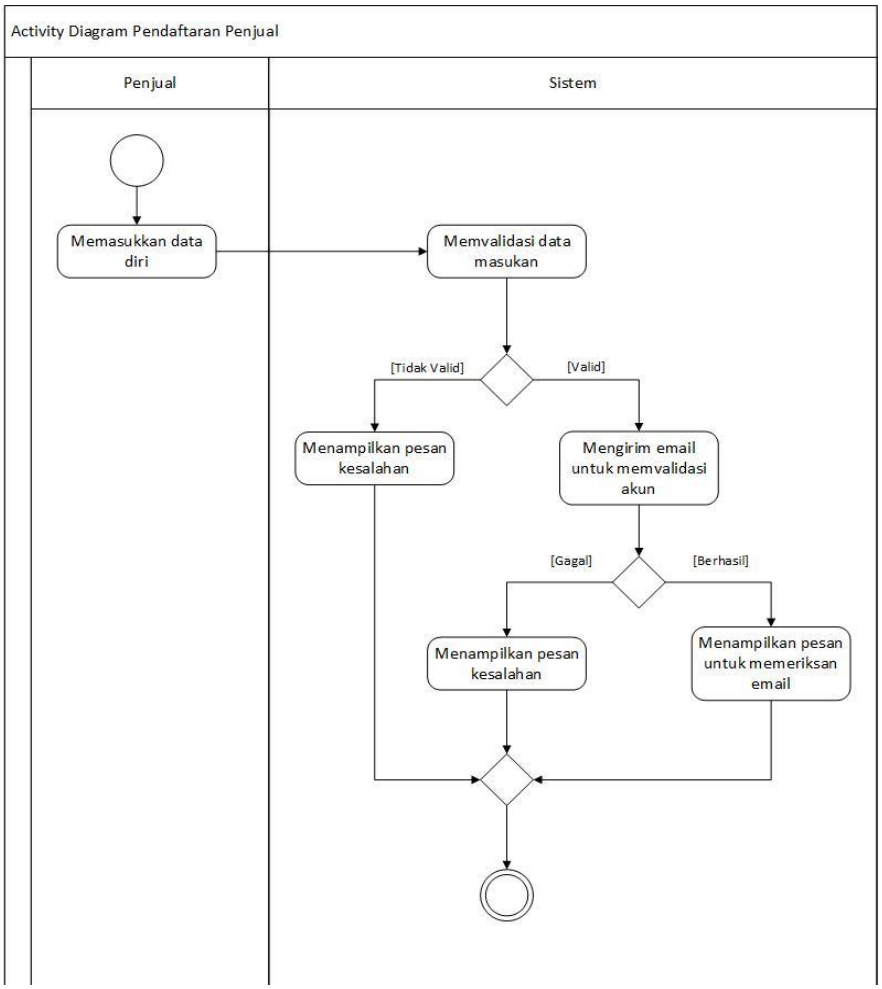
Proses pendaftaran ojek pada aplikasi dapat dilihat pada Gambar 3.15 *Activity* Diagram Pendaftaran Ojek.



Gambar 3.15 *Activity* Diagram Pendaftaran Ojek

5. *Activity* Diagram Pendaftaran Penjual

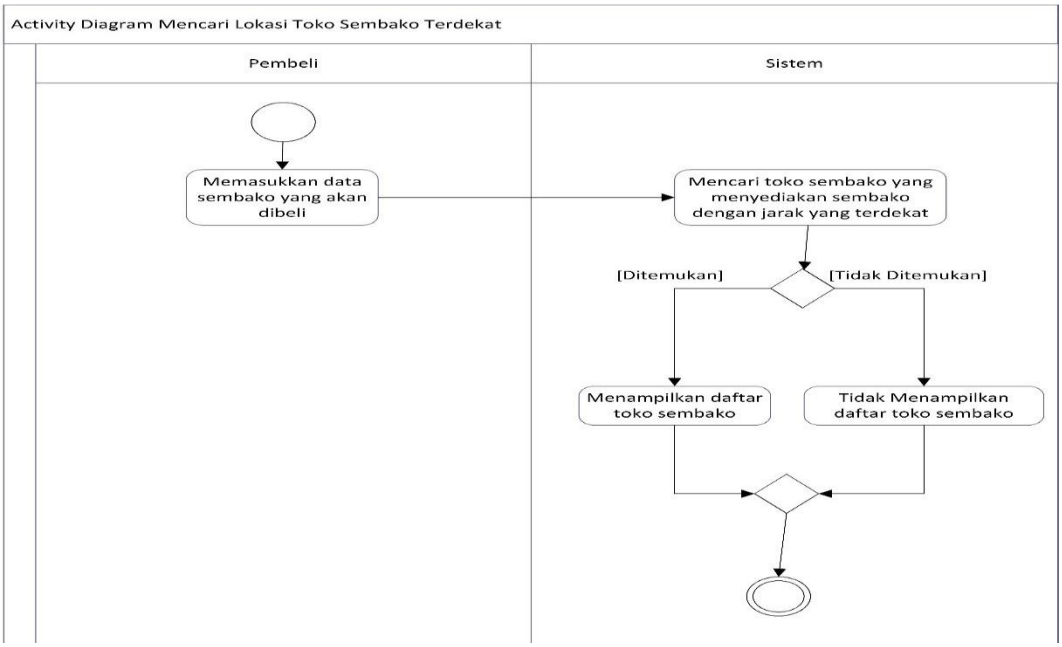
Proses pendaftaran penjual pada aplikasi dapat dilihat pada Gambar 3.16 *Activity* Diagram Pendaftaran Penjual.



Gambar 3.16 *Activity* Diagram Pendaftaran Penjual

6. *Activity* Diagram Mencari Lokasi Toko Sembako Terdekat

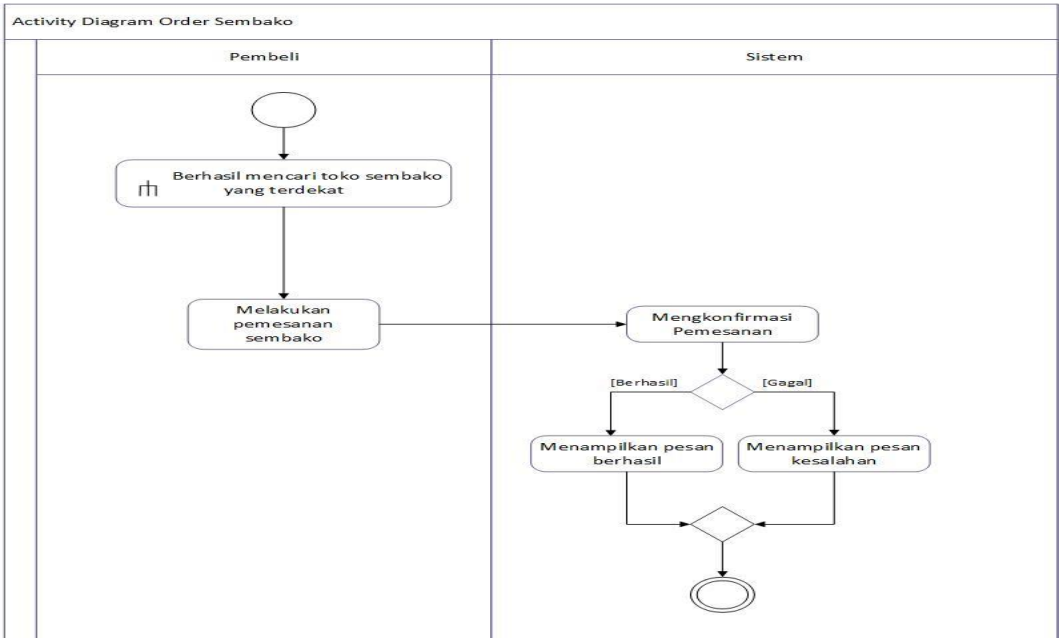
Proses pencarian lokasi toko sembako terdekat dapat dilihat pada Gambar 3.17 *Activity* Diagram Mencari Lokasi Toko Sembako Terdekat.



Gambar 3.17 *Activity* Diagram Mencari Lokasi Toko Sembako Terdekat

7. Activity Diagram Order Sembako

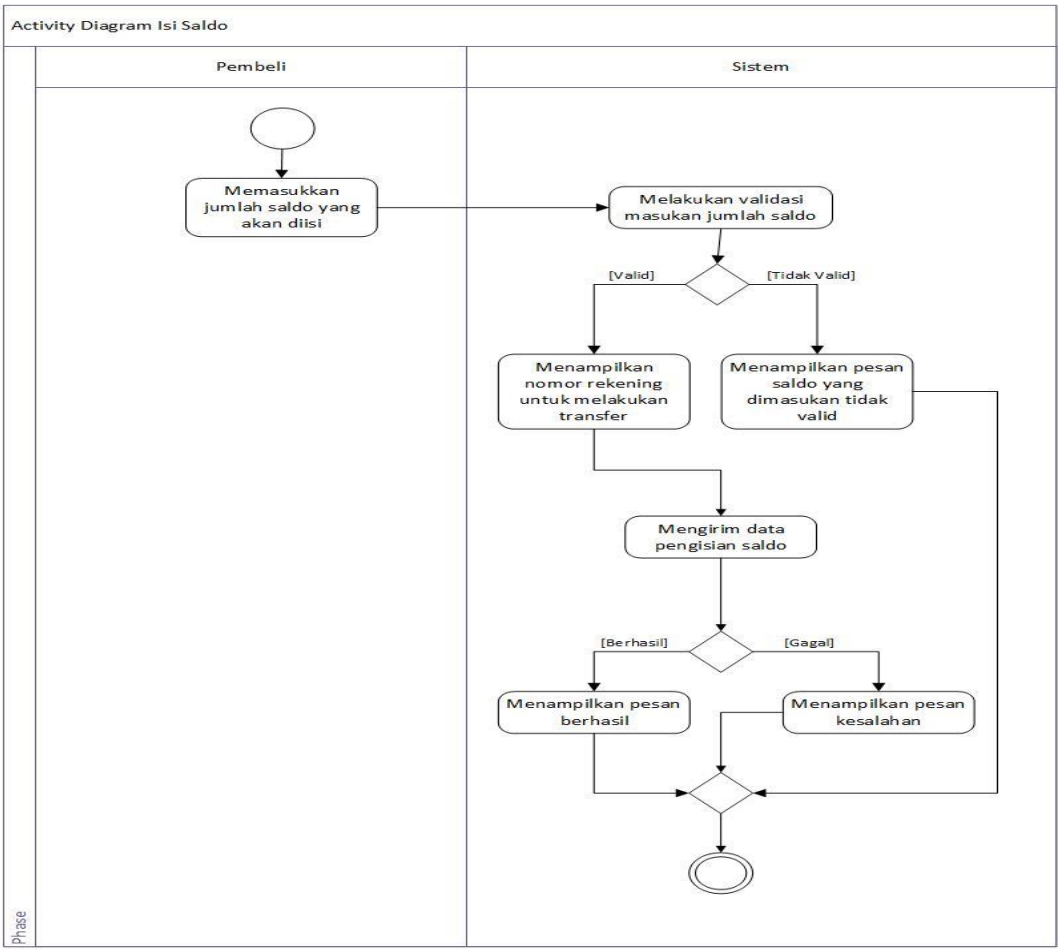
Proses order sembako dapat dilihat pada Gambar 3.18 *Activity* Diagram Order Sembako.



Gambar 3.18 *Activity* Diagram Order Sembako

8. Activity Diagram Isi Saldo

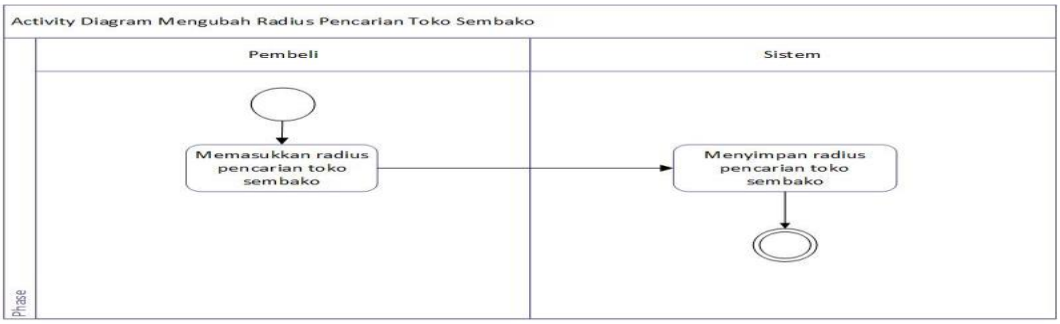
Proses isi saldo dapat dilihat pada Gambar 3.19 *Activity* Diagram Isi Saldo.



Gambar 3.19 *Activity* Diagram Isi Saldo

9. *Activity* Diagram Mengubah Radius Pencarian

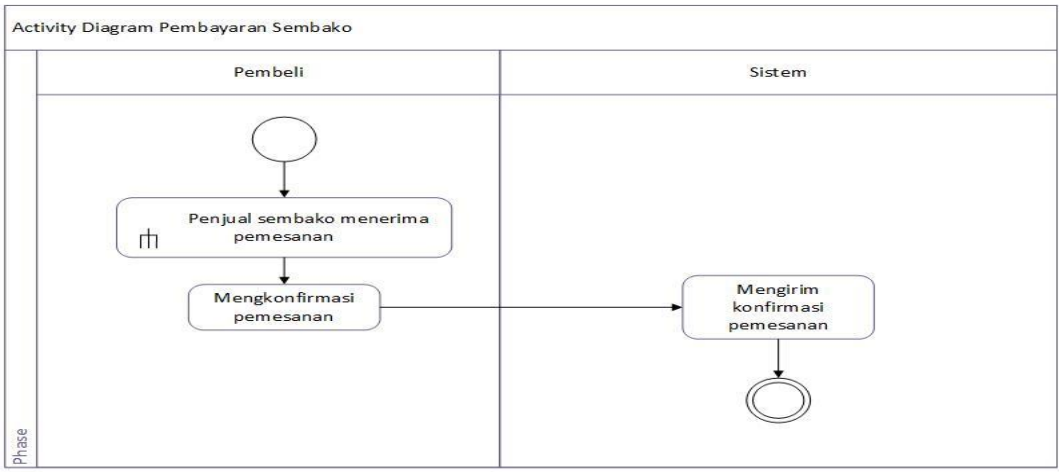
Proses mengubah radius pencarian dapat dilihat pada Gambar 3.20 *Activity* Diagram Mengubah Radius Pencarian Toko Sembako.



Gambar 3.20 *Activity* Diagram Mengubah Radius Pencarian Toko Sembako

10. *Activity* Diagram Pembayaran Sembako

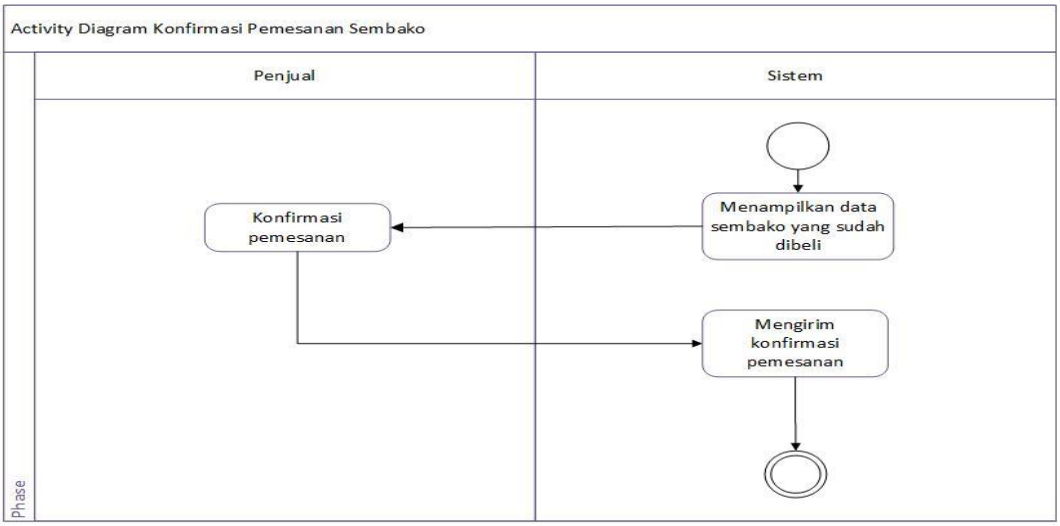
Proses pembayaran sembako dapat dilihat pada Gambar 3.21 *Activity* Diagram Pembayaran Sembako.



Gambar 3.21 *Activity* Diagram Pembayaran Sembako

11. *Activity* Diagram Konfirmasi Pemesanan Sembako

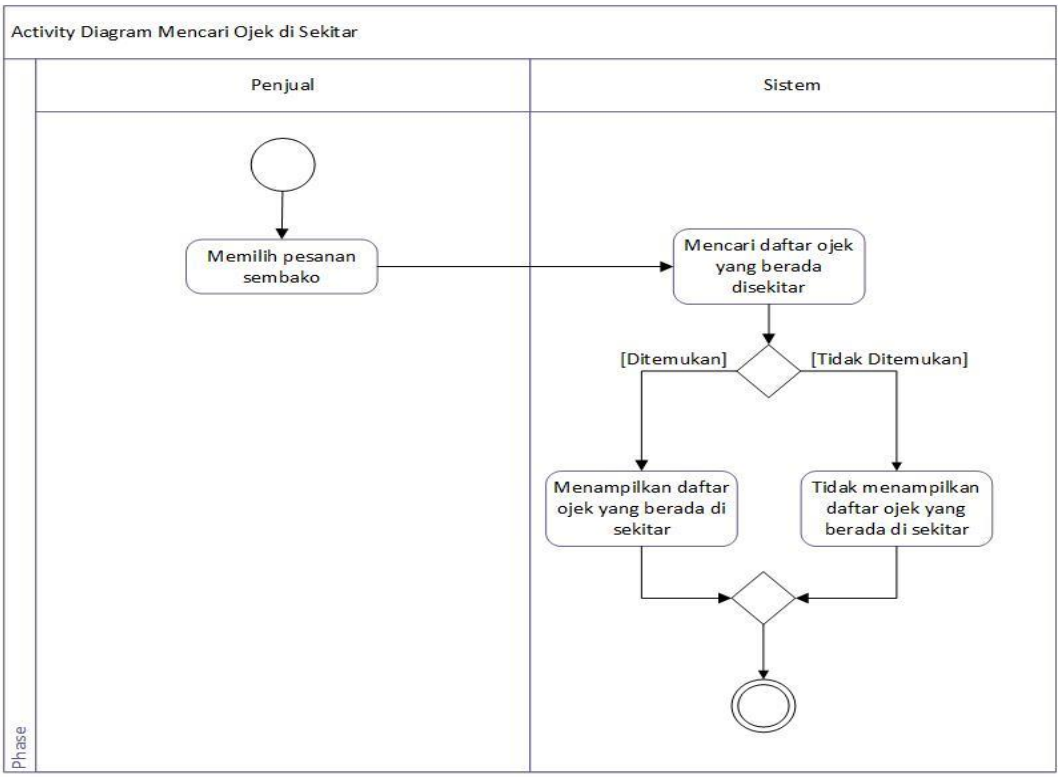
Proses konfirmasi pemesanan sembako dapat dilihat pada Gambar 3.22 *Activity* Diagram Konfirmasi Pemesanan Sembako.



Gambar 3.22 *Activity* Diagram Pemesanan Sembako

12. *Activity* Diagram Mencari Ojek Di Sekitar

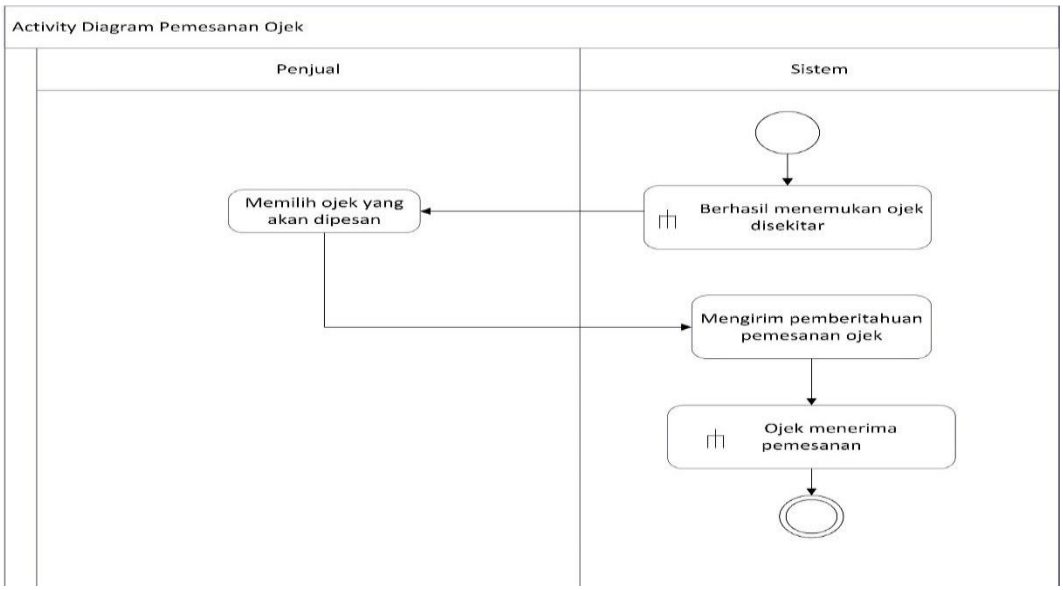
Proses mencari ojek di sekitar dapat dilihat pada Gambar 3.23 *Activity* Diagram Mencari Ojek di Sekitar.



Gambar 3.23 *Activity* Diagram Mencari Ojek Di Sekitar

13. *Acitivity* Diagram Pemesanan Ojek

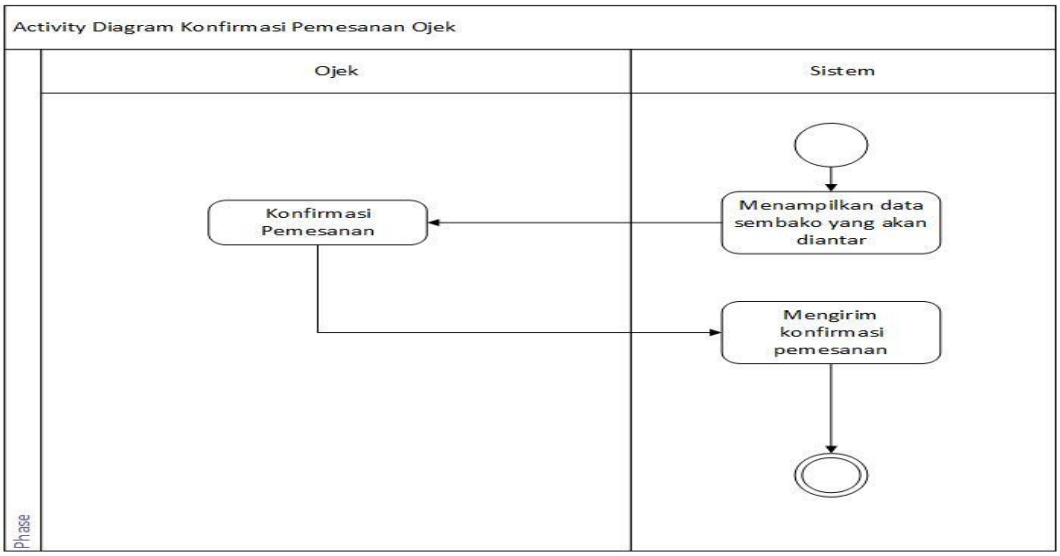
Proses pemesanan ojek dapat dilihat pada Gambar 3.24 *Activity* Diagram Pemesanan Ojek.



Gambar 3.24 *Activity* Diagram Pemesanan Ojek

14. *Activity* Diagram Konfirmasi Pemesanan Ojek

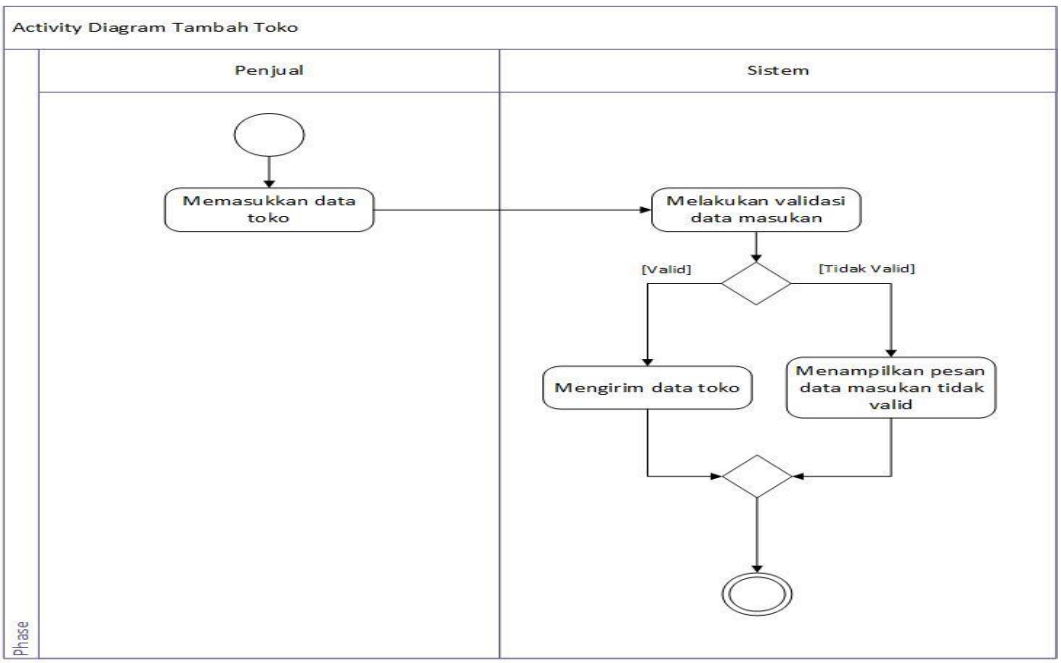
Proses konfirmasi pemesanan ojek dapat dilihat pada Gambar 3.25 *Activity* Diagram Konfirmasi Pemesanan Ojek.



Gambar 3.25 *Activity* Diagram Konfirmasi Pemesanan Ojek

15. *Activity* Diagram Tambah Toko

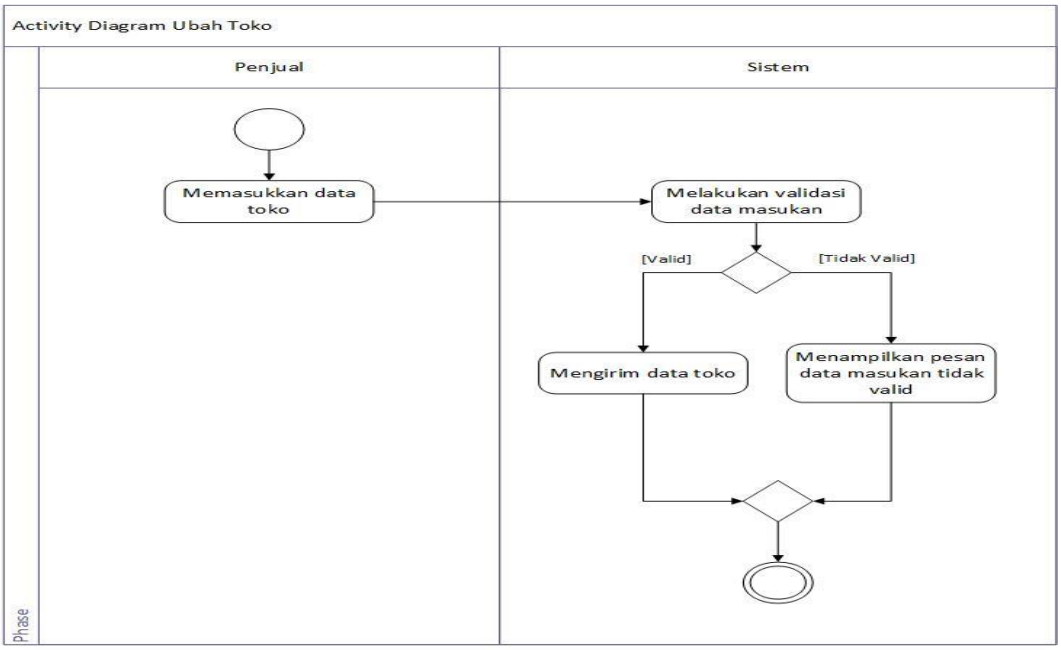
Proses tambah toko dapat dilihat pada Gambar 3.26 *Activity* Diagram Tambah Toko.



Gambar 3.26 *Activity* Diagram Tambah Toko

16. *Activity* Diagram Ubah Toko

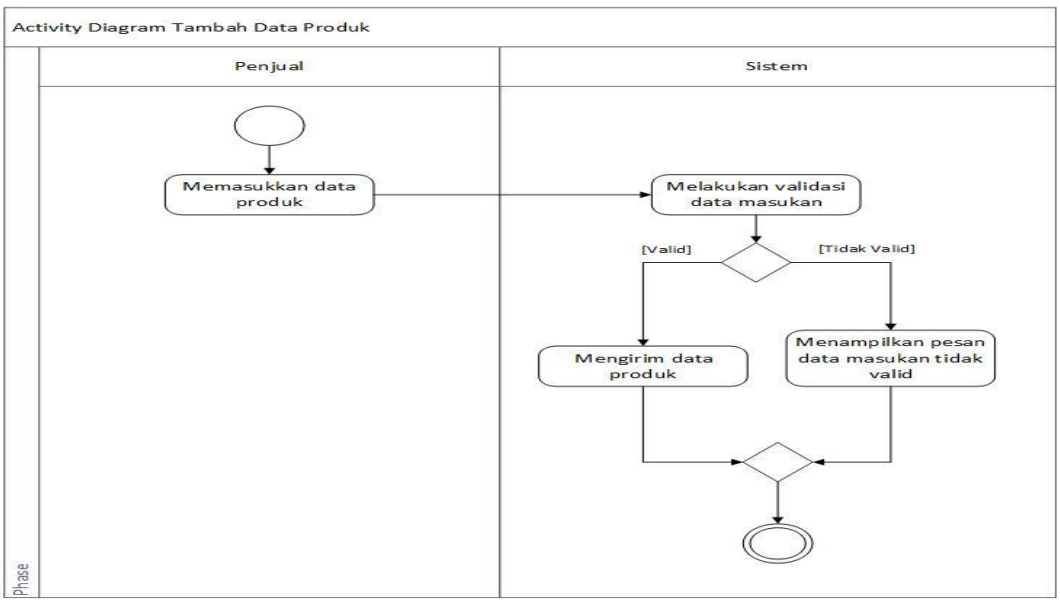
Proses ubah toko dapat dilihat pada Gambar 3.27 *Activity* Diagram Ubah Toko.



Gambar 3.27 *Activity* Diagram Ubah Toko

17. *Activity* Diagram Tambah Data Produk

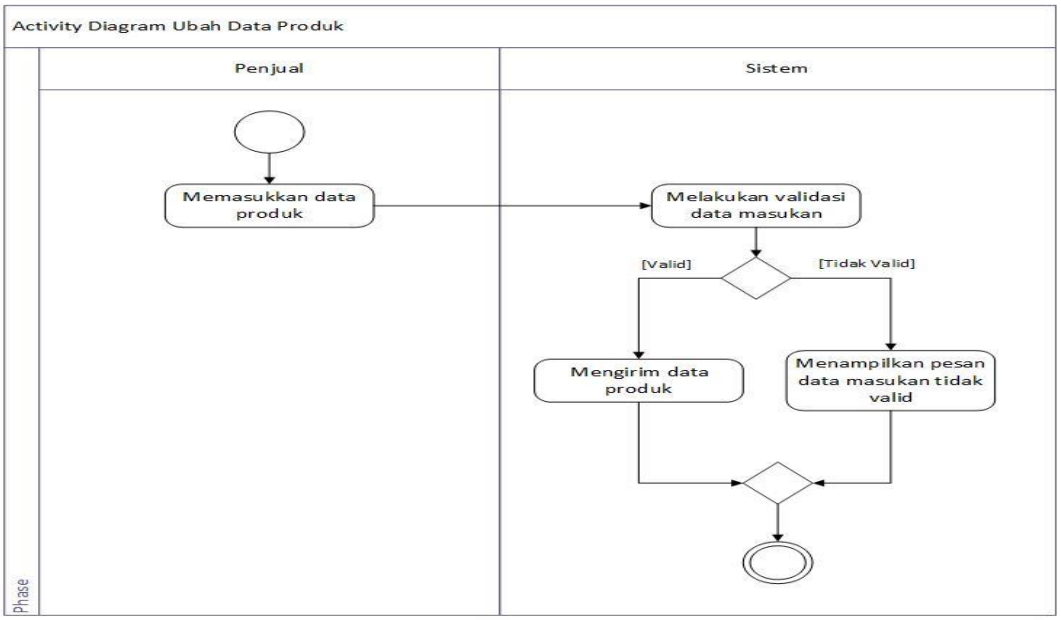
Proses tambah data produk dapat dilihat pada Gambar 3.28 *Activity* Diagram Tambah Data Produk.



Gambar 3.28 *Activity* Diagram Tambah Data Produk

18. *Activity* Diagram Ubah Data Produk

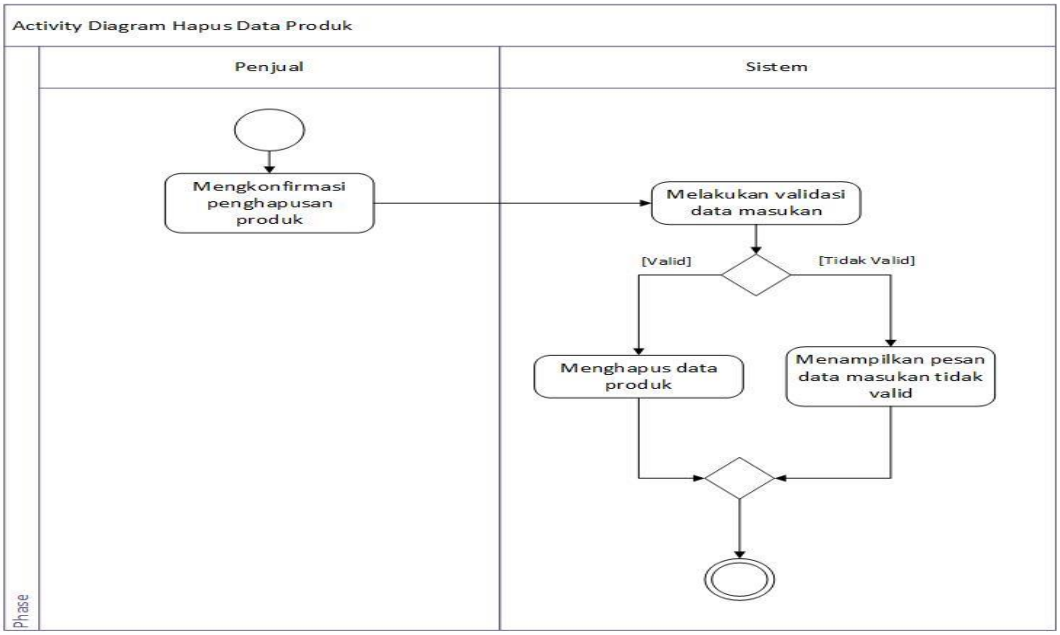
Proses ubah data produk dapat dilihat pada Gambar 3.29 *Activity* Diagram Ubah Data Produk.



Gambar 3.29 *Activity* Diagram Ubah Data Produk

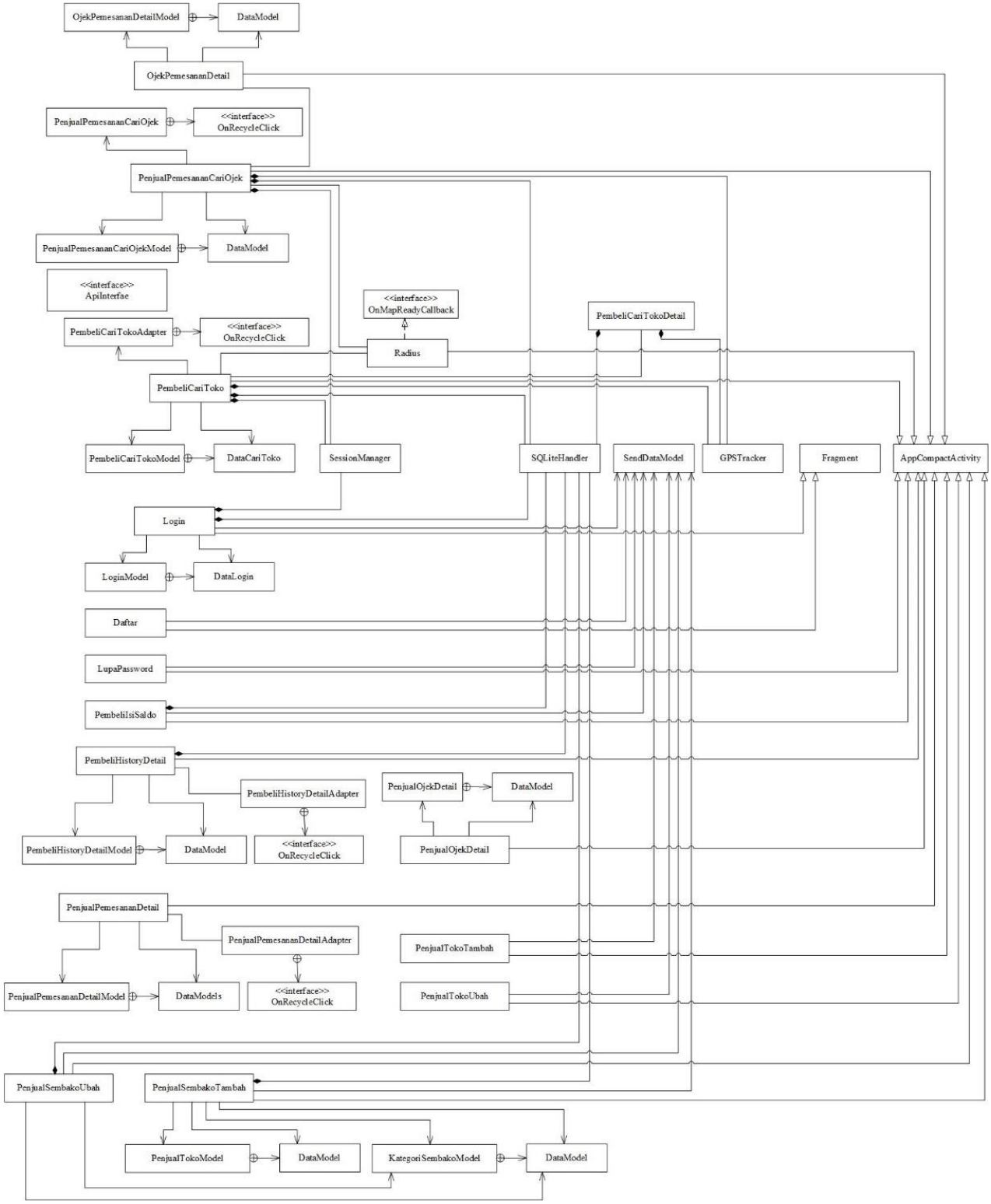
19. *Activity* Diagram Hapus Data Produk

Proses hapus data produk dapat dilihat pada Gambar 3.30 *Activity* Diagram Hapus Data Produk.



Gambar 3.30 *Activity* Diagram Hapus Data Produk

3.1.6.1.3 *Class* Diagram



Gambar 3.31 *Class* Diagram

3.1.6.1.4 Deskripsi *Class* Diagram

<<Interface>>
ApiInterfae

+ getDataLogin(token: String, email: String, password: String, type: String) : Call<LoginModel>
+ getRegisterResponse(token: String, type:String, nama: String, alamat: String, lat: String, lng: String, telepon: String, email: String, password: String, image_name: String, encode_string: String): Call<SendDataModel>
+ getkategoriSembako(token: String): Call<PembeliKategoriSembakoModel>
+ getPenjualSembakoKategori(token: String, idPenjual: String): Call<PembeliKategoriSembakoModel>
+ getDaftarSembakoPerkategori(token: String., id: String): Call<PembeliDaftarSembakoPerkategoriModel>
+ getSembakoTokoTerdekatDaftar(token: String, id: String, idToko: String): Call<PembeliDaftarSembakoPerkategoriModel>
+ getPembeliTokoTerdekat(token: String, distance: String, latitude: String, longitude: String): Call<PembeliTokoTerdekatModel>
+ getTokoTerdekatkategoriSembako(token: Srting, idToko: String): Call<PembeliKategoriSembakoModel>
+ getCariToko(token: ArrayList<String>, namaSembako: ArrayList<String>, jumlah: ArrayList<String>, distance: String, latitude: String, longitude: String,): Call<PembeliCariTokoModel>
+ getPenjualToko(token: String, idPenjual: String): Call<PenjualTokoModel>
+ sendPenjualSembakoTambah(token: String, idToko: String, idKategori: String, namaSembako: String, kualitas: String, harga: String, stok: String, minimum: String, imagerName: String, encodedString: String): Call<SendDataModel>
+ getKategoriSembako(token: String): Call<KategoriSembakoModel>
+ getBank(token: String): Call<PenjualBankModel>
+ getSembakoTambahOrder(token: String, jumlahPembelian: ArrayList<String>, idSembako: ArrayList<String>, subTotal: ArrayList<String>, total: String, idPembeli: String, jarak: String, latitude: String, longitude: String): Call<SendDataModel>
+ getDataTopUp(token: String, idPembeli: String): Call<PembeliDataTopUpModel>
+ getPenjualRekening(token: String, idPenjual: String): Call<PenjualRekeningModel>
+ getPembeliHistory(token: String, idPembembeli: String): Call<PembeliHistoryModel>
+ getPembeliHistoryDetail(token: String, idPembelian: String): Call<PembeliHistoryDetailModel>
+ sendPembeliOrderKonfirmasi(token: String, idPembelian: String, status: String, idOrder: String, idOjek: String): Call<SendDataModel>
+ sendPenjualOrderKonfirmasi(token: String, idPembelian: String, status: String, idOjek: String): Call<SendDataModel>
+ sendPenjualOjekKonfirmasi(token: String, idOrder:String, idOjek:String, statusOrder:String): Call<SendDataModel>
+ sendPembeliOrderPembayaran(token: String, idPembelian: String, idPembeli: String): Call<SendDataModel>
+ sendPembeliTambahSaldo(token: String, idPembeli:String, jumlahTopUp:String, imageName:String, encodedString: String): Call<SendDataModel>
+ sendPenjualTokoTambah(token, idPenjual, namaToko, alamat, lat, lng, imageName, encodeString): Call<SendDataModel>
+ getPenjualOrderData(token: String, idPenjual: String): Call<PenjualPemesananModel>
+ getPenjualOjekData(token: String, idPenjual: String): Call<PenjualOjekModel>
+ getOjekPemesanan(token: String, idPenjual: String): Call<OjekPemesananModel>
+ getPenjualPemesananDetail(token: String, idPembelian: String): Call<PenjualPemesananDetailModel>
+ getPenjualOjekDetail(token: String, idOrderOjek: String): Call<PenjualOjekDetailModel>
+ getOjekPemesananDetail(token: String, idOrderOjek: String): Call<OjekPemesananDetailModel>
+ getPenjualPemesananDetail(token: String, distance: String, latitude: String, longitude: String): Call<PenjualPemesananCariOjekModel>
+ sendPenjualTambahOrderOjek(token: String, idOjek: String, idPenjual: String, idPembelian: String, totalBayar: String): Call<SendDataModel>
+ sendTambahRekening(token: String, idPenjual: String, idBank: String, pemilikRekening: String, nomorRekening: String): Call<SendDataModel>
+ sendLupaPassword(token: String, email: String): Call<SendDataModel>
+ getDataPencairan(token: String, idPenjual: String): Call<PenjualPencairanModel>

LoginModel
- dataLoginPembeli: DataLogin - status: String - message: String
+ getDataLogin(): DataLogin + setDataLogin(dataLoginPembeli: DataLogin): void + getStatus(): String + setStatus(status: String): void + getMessage(): String + setMessage(message: String): void

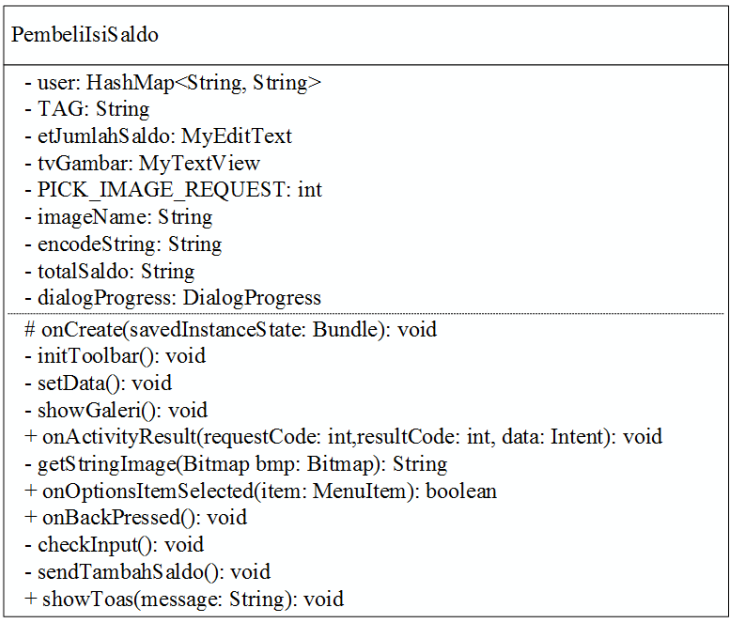
DataLogin
- type: String - id: String - alamat: String - telepon: String - latitude: String - longitude: String - email: String - createdAt: String - totalSaldo: String - gambar: String
+ DataLogin(type: String, id: String, nama: String, telepon: String, latitude: String, logngitude: String, email: String, createdAt: String, totalSaldo: String, gambar:String) + getType(): String + setType(type: String): void + getId(): String + setId(id: String): void + getAlamat(): String + setAlamat(alamat: String): void + getTelepon(): String + setTelepon(telepon: String): void + getLatitude(): String + setLatitude(latitude: String): void + getLongitude(): String + setLongitude(longitude: String): void + getEmail(): String + setEmail(email: String): void + getCreatedAt(): String + setCreatedAt(createdAt: String): void + getTotalSaldo(): String + setTotalSaldo(totalSaldo: String): void + getGambar(): String + setGambar(gambar: String): void

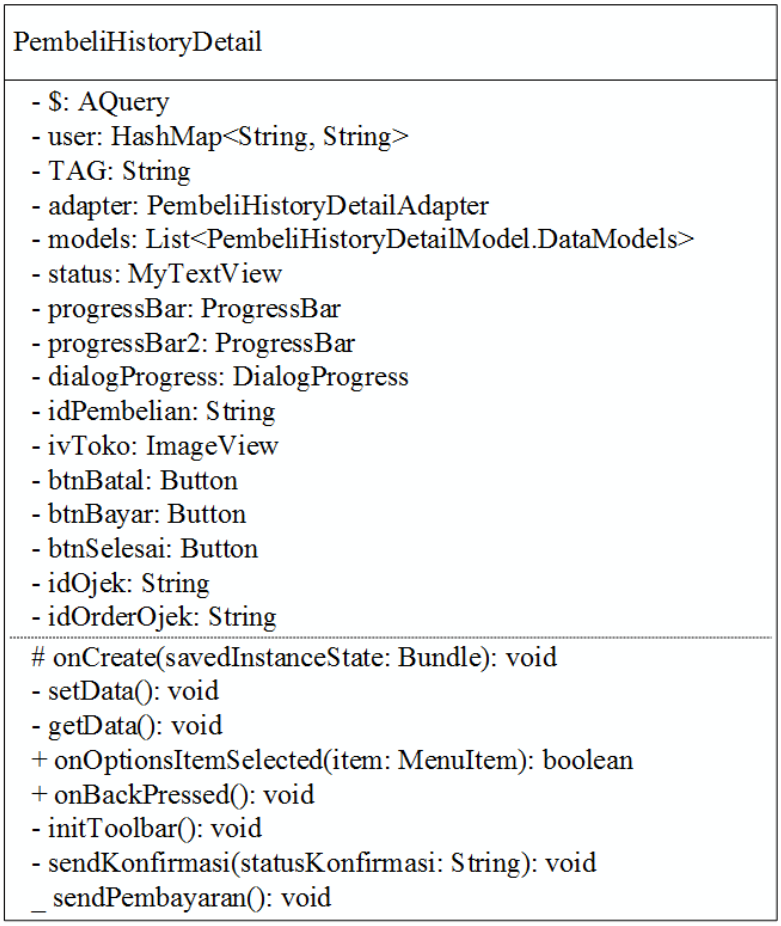
Daftar
- TAG: String - PICK_IMAGE_REQUEST: int - PLACE_PICKER_REQUEST: int - ivPhoto: ImageView - tvPembeli: MyTextView - tvOjek: MyTextView - tvPenjual: MyTextView - etNama: MyEditText - etAlamat: MyEditText - etTelepon: MyEditText - etEmail: MyEditText - etPassword: MyEditText - etPasswordKonfirmasi: MyEditText - btnDaftar: LinearLayout - type: String - dialogProgress: DialogProgress - nama: String - alamatVal: String - latitude: String - longitude: String - telepon: String - email: String - password: String - imageName: String - encodeString: String
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState:Bundle): View - setData(): void - showGaleri(): void + onActivityResult(requestCode: int,resultCode: int, data: Intent): void - getStringImage(Bitmap bmp: Bitmap): String - checkInput(): void - sendData(): void + showDialogCustom(message: String, status: int): void + showToast(message: String): void + isValidEmailAddress(email: String): boolean

Login
- TAG: String - session: SessionManager - db: SQLiteHandler - etEmail: MyEditText - etPassword: MyEditText - tvTampilkan: MyTextView - rbPembeli: RadioButton - rbPenjual: RadioButton - rbOjek: RadioButton - dialogProgress: DialogProgress - models: LoginModel.DataLogin - email: String - password: String - akses: String - showPass: boolean
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState:Bundle): View - checkData(): void + isValidEmailAddress(email: String): boolean - sendData(): void - saveData(): void - showHide(): void

Daftar
- TAG: String - PICK_IMAGE_REQUEST: int - PLACE_PICKER_REQUEST: int - ivPhoto: ImageView - tvPembeli: MyTextView - tvOjek: MyTextView - tvPenjual: MyTextView - etNama: MyEditText - etAlamat: MyEditText - etTelepon: MyEditText - etEmail: MyEditText - etPassword: MyEditText - etPasswordKonfirmasi: MyEditText - btnDaftar: LinearLayout - type: String - dialogProgress: DialogProgress - nama: String - alamatVal: String - latitude: String - longitude: String - telepon: String - email: String - password: String - imageName: String - encodeString: String
+ onCreateView(inflater: LayoutInflater, container: ViewGroup, savedInstanceState:Bundle): View - setData(): void - showGaleri(): void + onActivityResult(requestCode: int,resultCode: int, data: Intent): void - getStringImage(Bitmap bmp: Bitmap): String - checkInput(): void - sendData(): void + showDialogCustom(message: String, status: int): void + showToast(message: String): void + isValidEmailAddress(email: String): boolean

LupaPassword
- TAG: String - etEmail: MyEditText - email: String - dialogProgress: DialogProgress
onCreate(savedInstanceState: Bundle): void - checkInput(): void + isValidEmailAddress(email: String): boolean + sendData(): void


PembeliIsiSaldo
- user: HashMap<String, String>
- TAG: String
- etJumlahSaldo: MyEditText
- tvGambar: MyTextView
- PICK_IMAGE_REQUEST: int
- imageName: String
- encodeString: String
- totalSaldo: String
- dialogProgress: DialogProgress
onCreate(savedInstanceState: Bundle): void
- initToolbar(): void
- setData(): void
- showGaleri(): void
+ onActivityResult(requestCode: int,resultCode: int, data: Intent): void
- getStringImage(Bitmap bmp: Bitmap): String
+ onOptionsItemSelected(item: MenuItem): boolean
+ onBackPressed(): void
- checkInput(): void
- sendTambahSaldo(): void
+ showToas(message: String): void


PembeliHistoryDetail
- $: AQuery
- user: HashMap<String, String>
- TAG: String
- adapter: PembeliHistoryDetailAdapter
- models: List<PembeliHistoryDetailModel.DataModels>
- status: MyTextView
- progressBar: ProgressBar
- progressBar2: ProgressBar
- dialogProgress: DialogProgress
- idPembelian: String
- ivToko: ImageView
- btnBatal: Button
- btnBayar: Button
- btnSelesai: Button
- idOjek: String
- idOrderOjek: String
onCreate(savedInstanceState: Bundle): void
- setData(): void
- getData(): void
+ onOptionsItemSelected(item: MenuItem): boolean
+ onBackPressed(): void
- initToolbar(): void
- sendKonfirmasi(statusKonfirmasi: String): void
_ sendPembayaran(): void

PembeliHistoryDetailAdapter

- models: List<PembeliHistoryDetailModel.DataModels>
- context: Context
- listener: OnRecycleClick

+ PembeliHistoryDetailAdapter(context: Context, models: List<PembeliHistoryDetailModel.DataModels>, listener: OnRecycleClick)
+ onCreateViewHolder(parent: ViewGroup, viewType: int): ViewHolder
+ setFilter(model: List<PembeliHistoryDetailModel.DataModels>): void
+ onBindViewHolder(holder: ViewHolder, position: int): void
+ getItemCount(): int

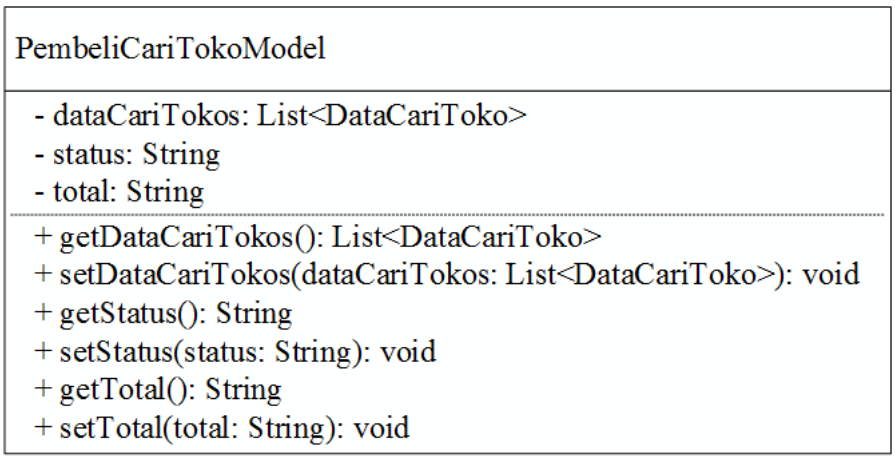
PembeliHistoryDetailModel

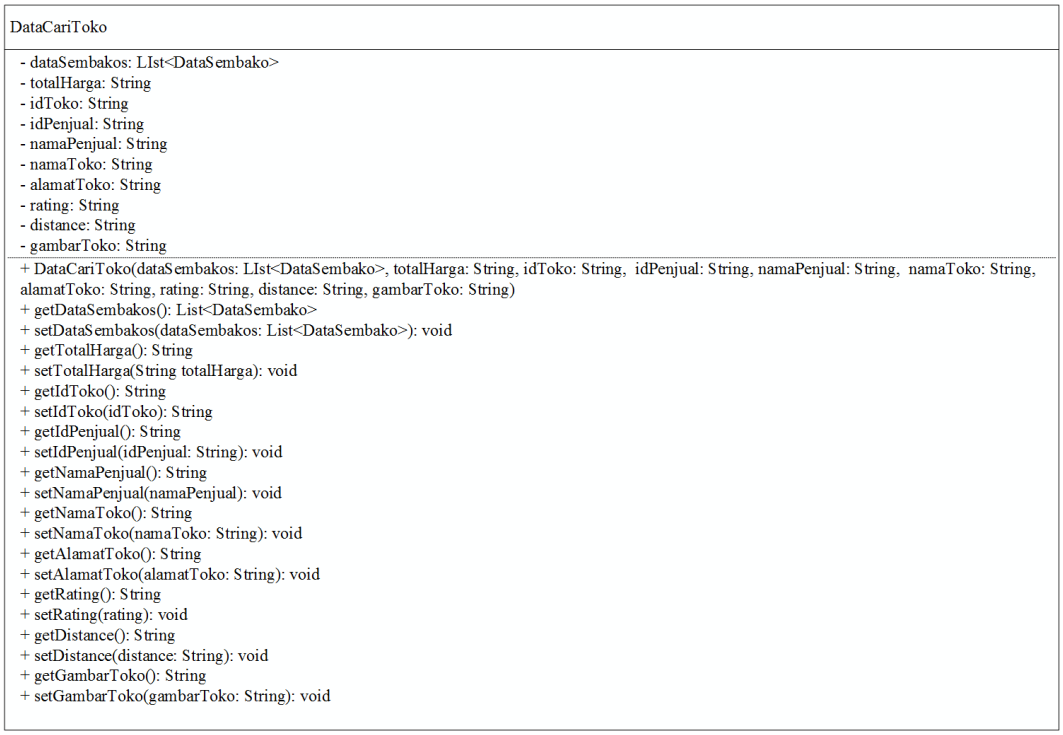
- idToko: String
- namaToko: String
- alamatToko: String
- gambarToko: String
- ongkos: String
- totalPembelian: String
- statusPembelian: String
- statusPembayaran: String
- idOrderOjek: String
- idOjek: String
- namaOjek: String
- statusOrderOjek: String
- status: String
- total: String
- dataModels : List<DataModels>

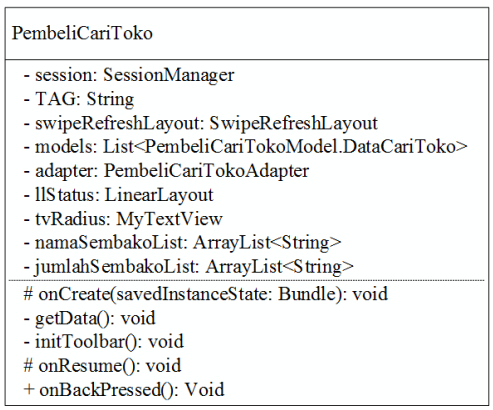
+ getIdToko(): String
+ setIdToko(idToko: String): void
+ getNamaToko(): String
+ setNamaToko(namaToko: String): void
+ getAlamatToko(): String
+ setAlamatToko(alamatToko: String): void
+ getGambarToko(): String
+ setGambarToko(gambarToko: String): void
+ getOngkos(): String
+ setOngkos(ongkos: String): void
+ getTotalPembelian(): String
+ setTotalPembelian(totalPembelian: String)
+ getStatusPembelian(): String
+ setStatusPembelian(statusPembelian: String): void
+ getStatusPembayaran(): String
+ setStatusPembayaran(statusPembayaran: String): void
+ getIdOrderOjek(): String
+ setIdOrderOjek(idOrderOjek: String): void
+ getIdOjek(): String
+ setIdOjek(idOjek: String): void
+ getNamaOjek(): String
+ setNamaOjek(namaOjek: String): void
+ getStatusOrderOjek(): String
+ setStatusOrderOjek(statusOrderOjek: String): void
+ String getStatus(): String
+ void setStatus(status: String): void
+ getTotal(): String
+ setTotal(total: String)
+ getDataModels(): List<DataModels>
+ setDataModels(dataModels: List<DataModels>): void

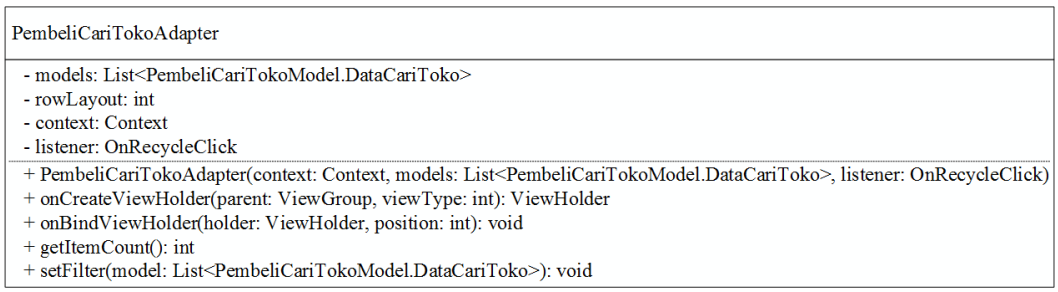

DataModel
- idSembako: String
- nama: String
- gambar: String
- kualitas: String
- harga: String
- jumlah: String
- satuan: String
- subtotal: String
+ DataModels(idSembako: String, nama: String, gambar: String, kualitas: String, harga: String, jumlah: String, satuan: String, subtotal: String)
+ getIdSembako(): String
+ setIdSembako(idSembako: String): void
+ getNama(): String
+ setNama(nama: String): void
+ getGambar(): String
+ setGambar(gambar: String): void
+ getKualitas(): String
+ setKualitas(kualitas: String): void
+ getHarga(): String
+ setHarga(harga: String): void
+ getJumlah(): String
+ setJumlah(jumlah: String): void
+ getSatuan(): String
+ setSatuan(satuan: String): void
+ getSubtotal(): String
+ setSubtotal(subtotal: String): Void

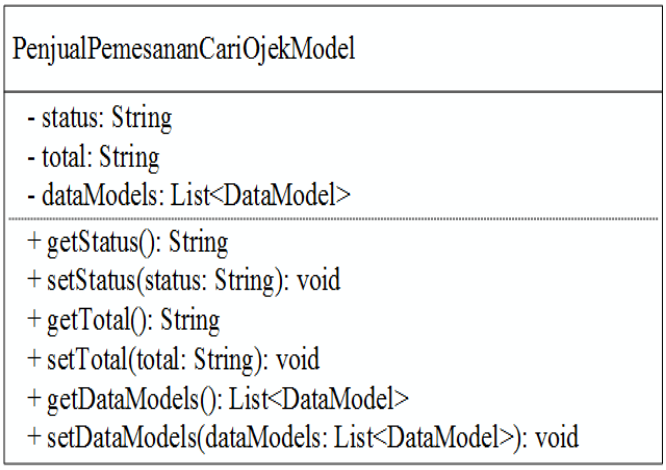

<<Interface>>
OnRecycleClick
+ onItemClick(id)

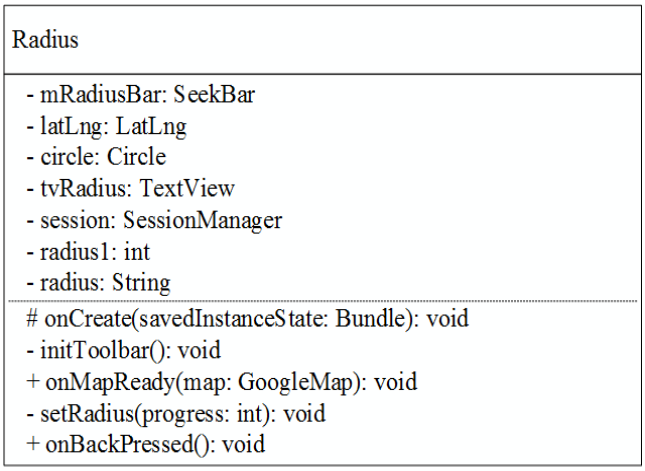

PembeliCariTokoModel
- dataCariTokos: List<DataCariToko>
- status: String
- total: String
+ getDataCariTokos(): List<DataCariToko>
+ setDataCariTokos(dataCariTokos: List<DataCariToko>): void
+ getStatus(): String
+ setStatus(status: String): void
+ getTotal(): String
+ setTotal(total: String): void

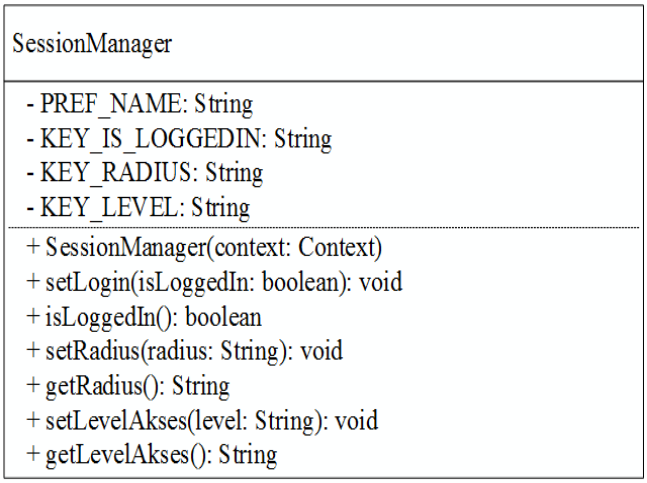

DataCariToko
- dataSembakos: LIst<DataSembako>
- totalHarga: String
- idToko: String
- idPenjual: String
- namaPenjual: String
- namaToko: String
- alamatToko: String
- rating: String
- distance: String
- gambarToko: String
+ DataCariToko(dataSembakos: LIst<DataSembako>, totalHarga: String, idToko: String, idPenjual: String, namaPenjual: String, namaToko: String, alamatToko: String, rating: String, distance: String, gambarToko: String)
+ getDataSembakos(): List<DataSembako>
+ setDataSembakos(dataSembakos: List<DataSembako>): void
+ getTotalHarga(): String
+ setTotalHarga(String totalHarga): void
+ getIdToko(): String
+ setIdToko(idToko): String
+ getIdPenjual(): String
+ setIdPenjual(idPenjual: String): void
+ getNamaPenjual(): String
+ setNamaPenjual(namaPenjual): void
+ getNamaToko(): String
+ setNamaToko(namaToko: String): void
+ getAlamatToko(): String
+ setAlamatToko(alamatToko: String): void
+ getRating(): String
+ setRating(rating): void
+ getDistance(): String
+ setDistance(distance: String): void
+ getGambarToko(): String
+ setGambarToko(gambarToko: String): void

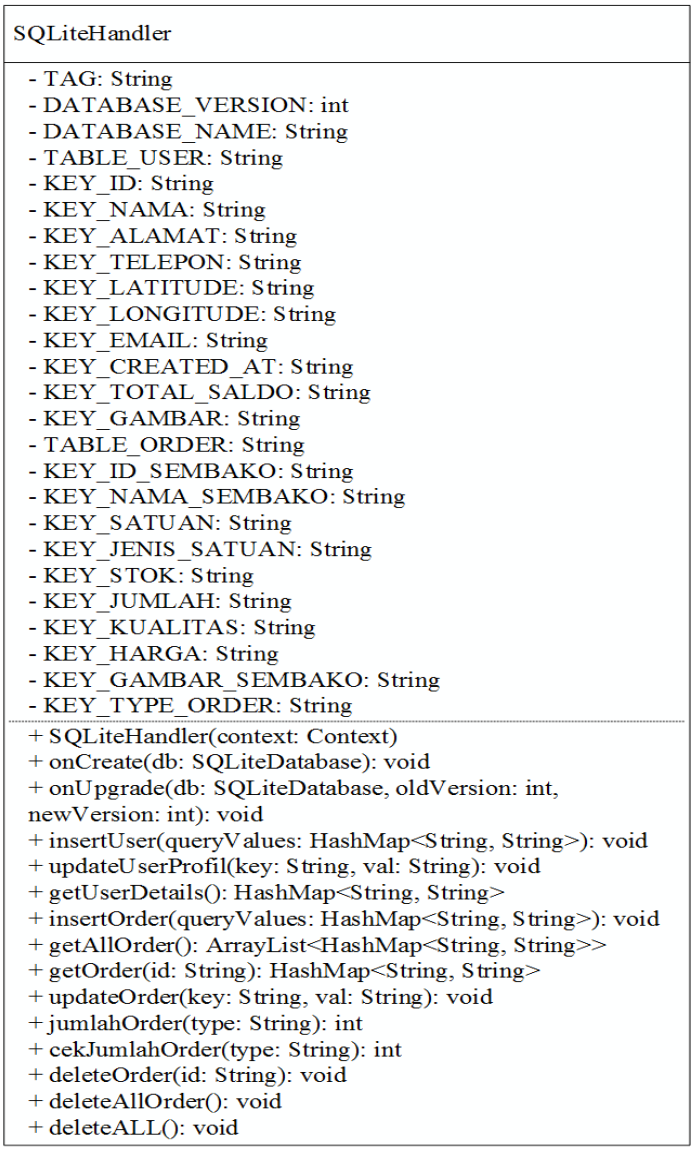

PembeliCariToko
- session: SessionManager
- TAG: String
- swipeRefreshLayout: SwipeRefreshLayout
- models: List<PembeliCariTokoModel.DataCariToko>
- adapter: PembeliCariTokoAdapter
- llStatus: LinearLayout
- tvRadius: MyTextView
- namaSembakoList: ArrayList<String>
- jumlahSembakoList: ArrayList<String>
onCreate(savedInstanceState: Bundle): void
- getData(): void
- initToolbar(): void
onResume(): void
+ onBackPressed(): Void

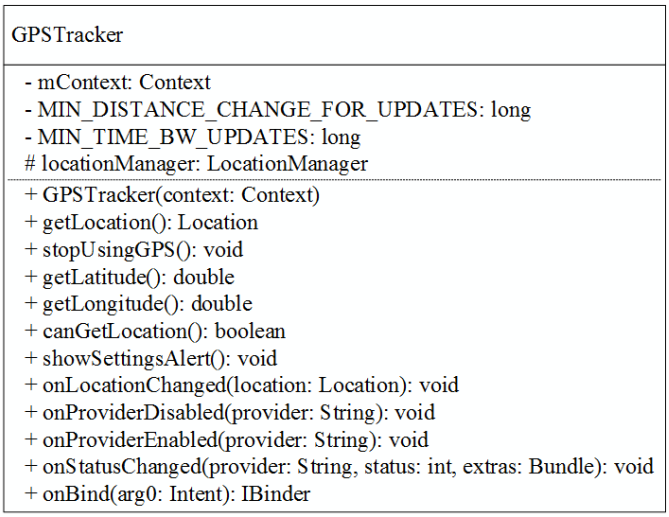

PembeliCariTokoAdapter
- models: List<PembeliCariTokoModel.DataCariToko>
- rowLayout: int
- context: Context
- listener: OnRecycleClick
+ PembeliCariTokoAdapter(context: Context, models: List<PembeliCariTokoModel.DataCariToko>, listener: OnRecycleClick)
+ onCreateViewHolder(parent: ViewGroup, viewType: int): ViewHolder
+ onBindViewHolder(holder: ViewHolder, position: int): void
+ getItemCount(): int
+ setFilter(model: List<PembeliCariTokoModel.DataCariToko>): void

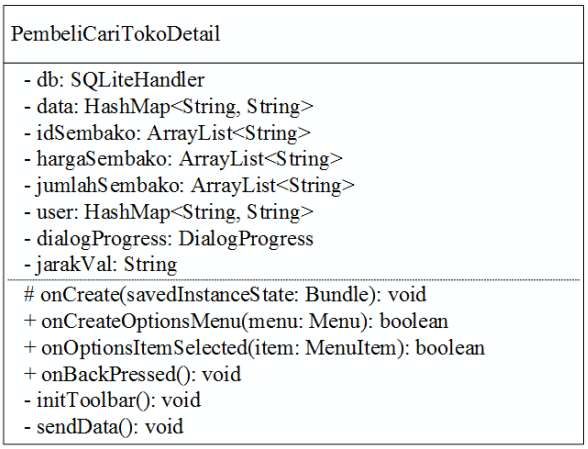

PenjualPemesananCariOjekModel
- status: String
- total: String
- dataModels: List<DataModel>
+ getStatus(): String
+ setStatus(status: String): void
+ getTotal(): String
+ setTotal(total: String): void
+ getDataModels(): List<DataModel>
+ setDataModels(dataModels: List<DataModel>): void

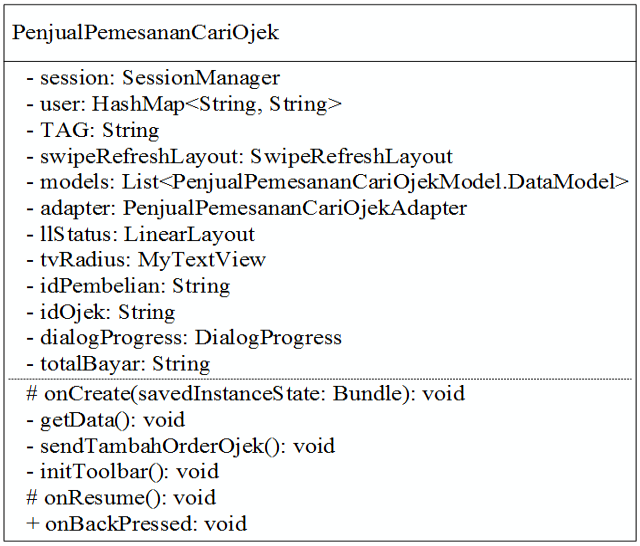

Radius
- mRadiusBar: SeekBar
- latLng: LatLng
- circle: Circle
- tvRadius: TextView
- session: SessionManager
- radius1: int
- radius: String
onCreate(savedInstanceState: Bundle): void
- initToolbar(): void
+ onMapReady(map: GoogleMap): void
- setRadius(progress: int): void
+ onBackPressed(): void

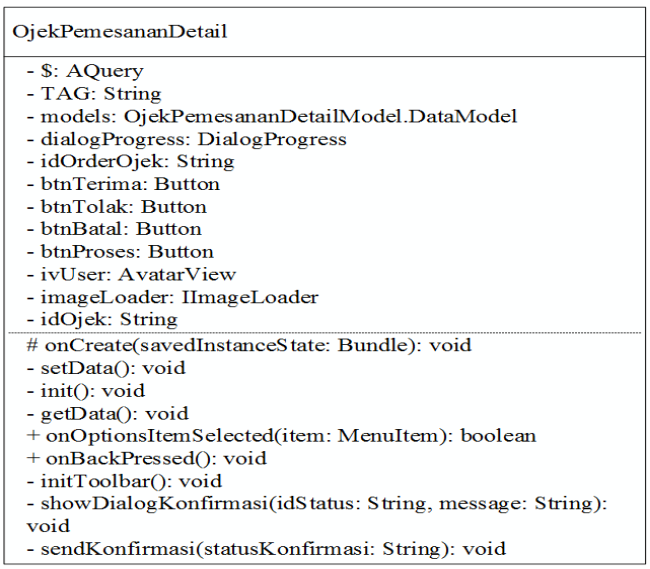

SessionManager
- PREF_NAME: String
- KEY_IS_LOGGEDIN: String
- KEY_RADIUS: String
- KEY_LEVEL: String
+ SessionManager(context: Context)
+ setLogin(isLoggedIn: boolean): void
+ isLoggedIn(): boolean
+ setRadius(radius: String): void
+ getRadius(): String
+ setLevelAkses(level: String): void
+ getLevelAkses(): String

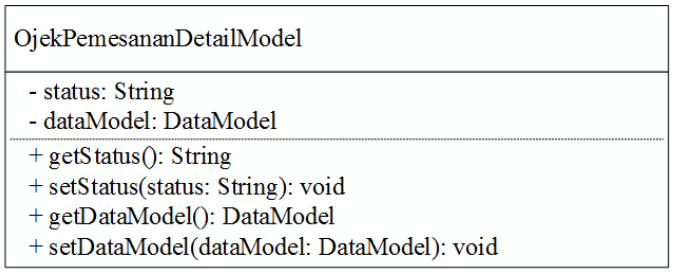

SQLiteHandler
- TAG: String
- DATABASE_VERSION: int
- DATABASE_NAME: String
- TABLE_USER: String
- KEY_ID: String
- KEY_NAMA: String
- KEY_ALAMAT: String
- KEY_TELEPON: String
- KEY_LATITUDE: String
- KEY_LONGITUDE: String
- KEY_EMAIL: String
- KEY_CREATED_AT: String
- KEY_TOTAL_SALDO: String
- KEY_GAMBAR: String
- TABLE_ORDER: String
- KEY_ID_SEMBAKO: String
- KEY_NAMA_SEMBAKO: String
- KEY_SATUAN: String
- KEY_JENIS_SATUAN: String
- KEY_STOK: String
- KEY_JUMLAH: String
- KEY_KUALITAS: String
- KEY_HARGA: String
- KEY_GAMBAR_SEMBAKO: String
- KEY_TYPE_ORDER: String
+ SQLiteHandler(context: Context)
+ onCreate(db: SQLiteDatabase): void
+ onUpgrade(db: SQLiteDatabase, oldVersion: int, newVersion: int): void
+ insertUser(queryValues: HashMap<String, String>): void
+ updateUserProfil(key: String, val: String): void
+ getUserDetails(): HashMap<String, String>
+ insertOrder(queryValues: HashMap<String, String>): void
+ getAllOrder(): ArrayList<HashMap<String, String>>
+ getOrder(id: String): HashMap<String, String>
+ updateOrder(key: String, val: String): void
+ jumlahOrder(type: String): int
+ cekJumlahOrder(type: String): int
+ deleteOrder(id: String): void
+ deleteAllOrder(): void
+ deleteALL(): void

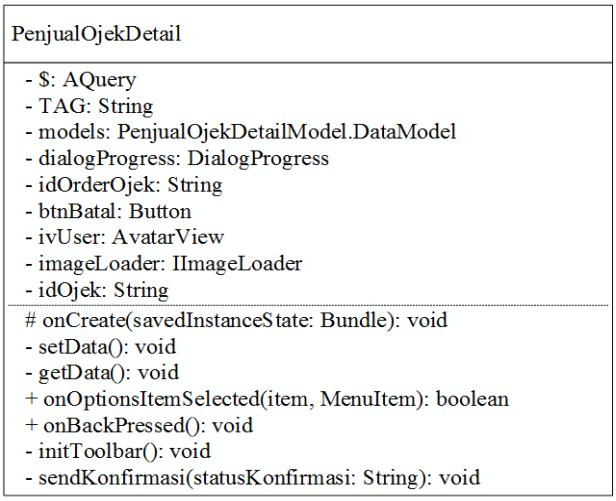

GPSTracker
- mContext: Context
- MIN_DISTANCE_CHANGE_FOR_UPDATES: long
- MIN_TIME_BW_UPDATES: long
locationManager: LocationManager
+ GPSTracker(context: Context)
+ getLocation(): Location
+ stopUsingGPS(): void
+ getLatitude(): double
+ getLongitude(): double
+ canGetLocation(): boolean
+ showSettingsAlert(): void
+ onLocationChanged(location: Location): void
+ onProviderDisabled(provider: String): void
+ onProviderEnabled(provider: String): void
+ onStatusChanged(provider: String, status: int, extras: Bundle): void
+ onBind(arg0: Intent): IBinder

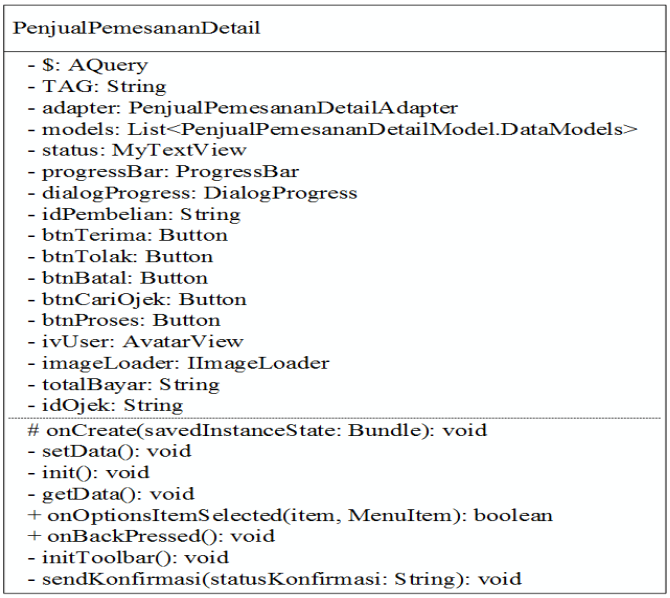

PembeliCariTokoDetail
- db: SQLiteHandler
- data: HashMap<String, String>
- idSembako: ArrayList<String>
- hargaSembako: ArrayList<String>
- jumlahSembako: ArrayList<String>
- user: HashMap<String, String>
- dialogProgress: DialogProgress
- jarakVal: String
onCreate(savedInstanceState: Bundle): void
+ onCreateOptionsMenu(menu: Menu): boolean
+ onOptionsItemSelected(item: MenuItem): boolean
+ onBackPressed(): void
- initToolbar(): void
- sendData(): void


PenjualPemesananCariOjek
- session: SessionManager
- user: HashMap<String, String>
- TAG: String
- swipeRefreshLayout: SwipeRefreshLayout
- models: List<PenjualPemesananCariOjekModel.DataModel>
- adapter: PenjualPemesananCariOjekAdapter
- llStatus: LinearLayout
- tvRadius: MyTextView
- idPembelian: String
- idOjek: String
- dialogProgress: DialogProgress
- totalBayar: String
onCreate(savedInstanceState: Bundle): void
- getData(): void
- sendTambahOrderOjek(): void
- initToolbar(): void
onResume(): void
+ onBackPressed: void


OjekPemesananDetail
- $: AQuery
- TAG: String
- models: OjekPemesananDetailModel.DataModel
- dialogProgress: DialogProgress
- idOrderOjek: String
- btnTerima: Button
- btnTolak: Button
- btnBatal: Button
- btnProses: Button
- ivUser: AvatarView
- imageLoader: IImageLoader
- idOjek: String
onCreate(savedInstanceState: Bundle): void
- setData(): void
- init(): void
- getData(): void
+ onOptionsItemSelected(item: MenuItem): boolean
+ onBackPressed(): void
- initToolbar(): void
- showDialogKonfirmasi(idStatus: String, message: String): void
- sendKonfirmasi(statusKonfirmasi: String): void

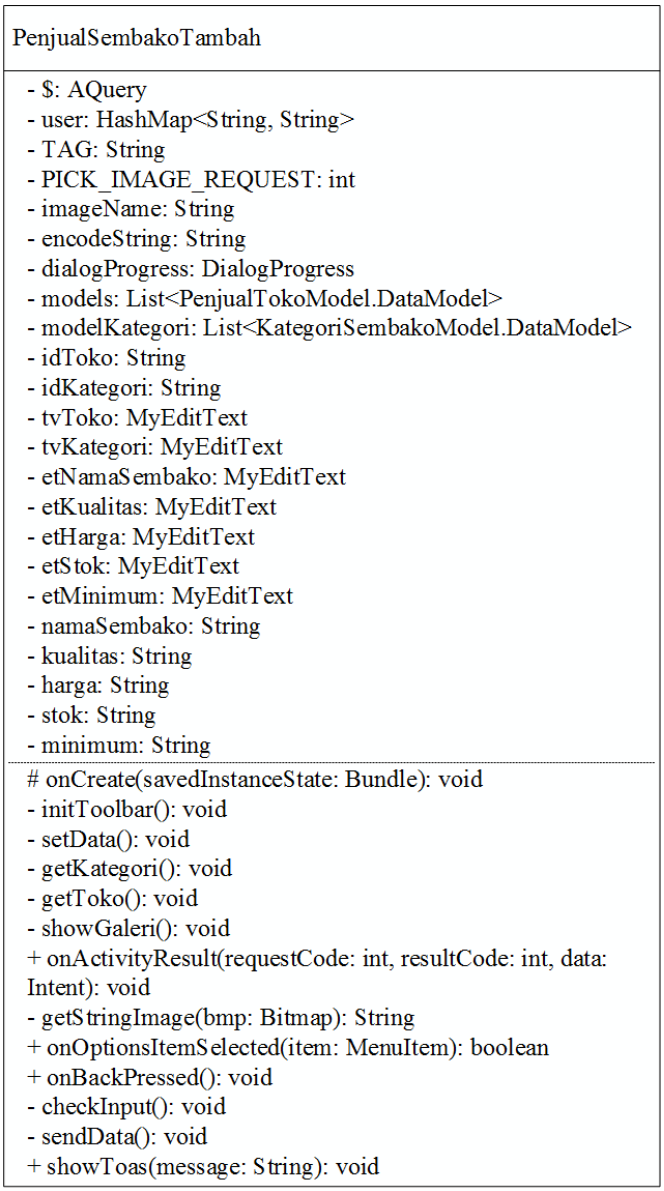

OjekPemesananDetailModel
- status: String
- dataModel: DataModel
+ getStatus(): String
+ setStatus(status: String): void
+ getDataModel(): DataModel
+ setDataModel(dataModel: DataModel): void

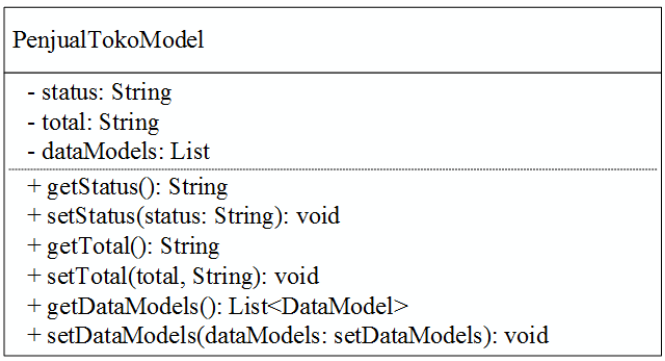

PenjualOjekDetail
- $: AQuery
- TAG: String
- models: PenjualOjekDetailModel.DataModel
- dialogProgress: DialogProgress
- idOrderOjek: String
- btnBatal: Button
- ivUser: AvatarView
- imageLoader: IImageLoader
- idOjek: String
onCreate(savedInstanceState: Bundle): void
- setData(): void
- getData(): void
+ onOptionsItemSelected(item, MenuItem): boolean
+ onBackPressed(): void
- initToolbar(): void
- sendKonfirmasi(statusKonfirmasi: String): void

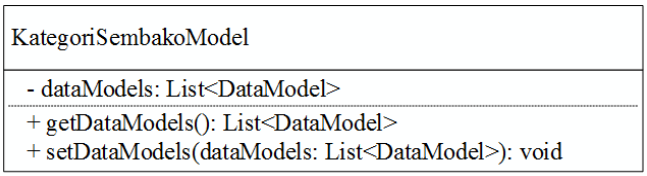

PenjualPemesananDetail
- $: AQuery
- TAG: String
- adapter: PenjualPemesananDetailAdapter
- models: List<PenjualPemesananDetailModel.DataModels>
- status: MyTextView
- progressBar: ProgressBar
- dialogProgress: DialogProgress
- idPembelian: String
- btnTerima: Button
- btnTolak: Button
- btnBatal: Button
- btnCariOjek: Button
- btnProses: Button
- ivUser: AvatarView
- imageLoader: IImageLoader
- totalBayar: String
- idOjek: String
onCreate(savedInstanceState: Bundle): void
- setData(): void
- init(): void
- getData(): void
+ onOptionsItemSelected(item, MenuItem): boolean
+ onBackPressed(): void
- initToolbar(): void
- sendKonfirmasi(statusKonfirmasi: String): void

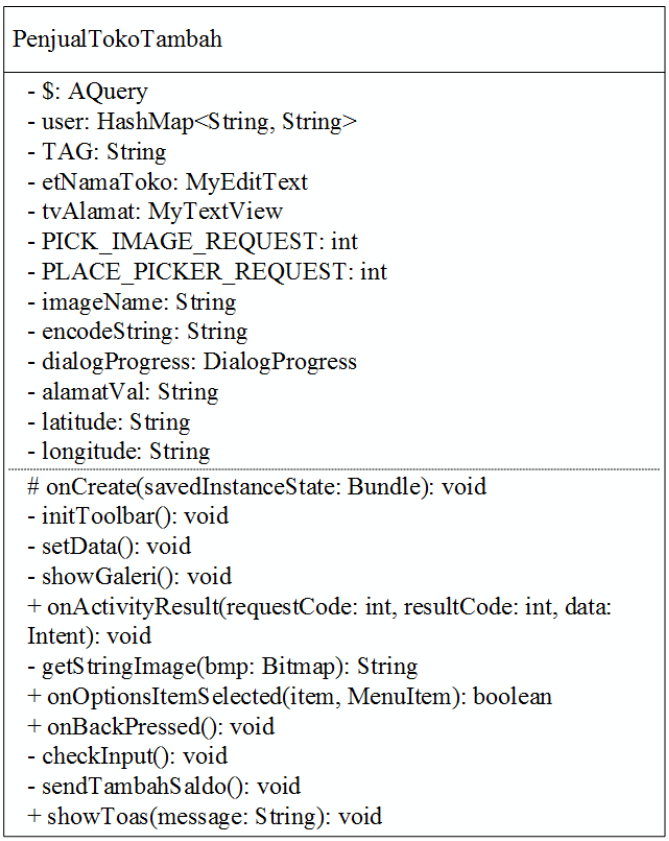
PenjualPemesananDetailAdapter
- $: AQuery - models: List<PenjualPemesananDetailModel.DataModels> - context: Context - listener: OnRecycleClick
+ PenjualPemesananDetailAdapter(context: Context, models List<PenjualPemesananDetailModel.DataModels>, listener:OnRecycleClick) + onCreateViewHolder(): ViewHolder + setFilter(model: List<PenjualPemesananDetailModel.DataModels>): void + onBindViewHolder(holder: ViewHolder, position: int): void + getItemCount(): int + OnRecycleClick(): interface

PenjualPemesananDetailModel
- jarak: String - idPembeli: String - namaPembeli: String - alamatPembeli: String - gambarPembeli: String - ongkos: String - totalPembelian: String - statusPembelian: String - statusPembayaran: String - idOjek: String - namaOjek: String - statusOrderOjek: String - status: String - total: String - dataModels: List<PenjualPemesananDetailModel.DataModels>
+ getJarak(): String + setJarak(jarak: String): void + getIdPembeli(): String + setIdPembeli(idPembeli: String): void + getNamaPembeli(): String + setNamaPembeli(namaPembeli, String): void + getAlamatPembeli(): String + setAlamatPembeli(alamatPembeli: String): void + getGambarPembeli(): String + setGambarPembeli(gambarPembeli: String): void + getOngkos(): String + setOngkos(ongkos: String): void + getTotalPembelian(): String + setTotalPembelian(totalPembelian: String): void + getStatusPembelian(): String + setStatusPembelian(statusPembelian: String): void + getStatusPembayaran(): String + setStatusPembayaran(statusPembayaran: String): void + getIdOjek(): String + setIdOjek(idOjek: String): void + getNamaOjek(): String + setNamaOjek(namaOjek: String): void + getStatusOrderOjek(): String + setStatusOrderOjek(statusOrderOjek: String): void + getStatus(): String + setStatus(status: String): void + getTotal(): String + setTotal(total: String): void + getDataModels(): List<DataModels> + setDataModels(dataModels: List<DataModels>): void


PenjualSembakoTambah
- $: AQuery
- user: HashMap<String, String>
- TAG: String
- PICK_IMAGE_REQUEST: int
- imageName: String
- encodeString: String
- dialogProgress: DialogProgress
- models: List<PenjualTokoModel.DataModel>
- modelKategori: List<KategoriSembakoModel.DataModel>
- idToko: String
- idKategori: String
- tvToko: MyEditText
- tvKategori: MyEditText
- etNamaSembako: MyEditText
- etKualitas: MyEditText
- etHarga: MyEditText
- etStok: MyEditText
- etMinimum: MyEditText
- namaSembako: String
- kualitas: String
- harga: String
- stok: String
- minimum: String
onCreate(savedInstanceState: Bundle): void
- initToolbar(): void
- setData(): void
- getKategori(): void
- getToko(): void
- showGaleri(): void
+ onActivityResult(requestCode: int, resultCode: int, data: Intent): void
- getStringImage(bmp: Bitmap): String
+ onOptionsItemSelected(item: MenuItem): boolean
+ onBackPressed(): void
- checkInput(): void
- sendData(): void
+ showToas(message: String): void


PenjualTokoModel
- status: String
- total: String
- dataModels: List
+ getStatus(): String
+ setStatus(status: String): void
+ getTotal(): String
+ setTotal(total, String): void
+ getDataModels(): List<DataModel>
+ setDataModels(dataModels: setDataModels): void



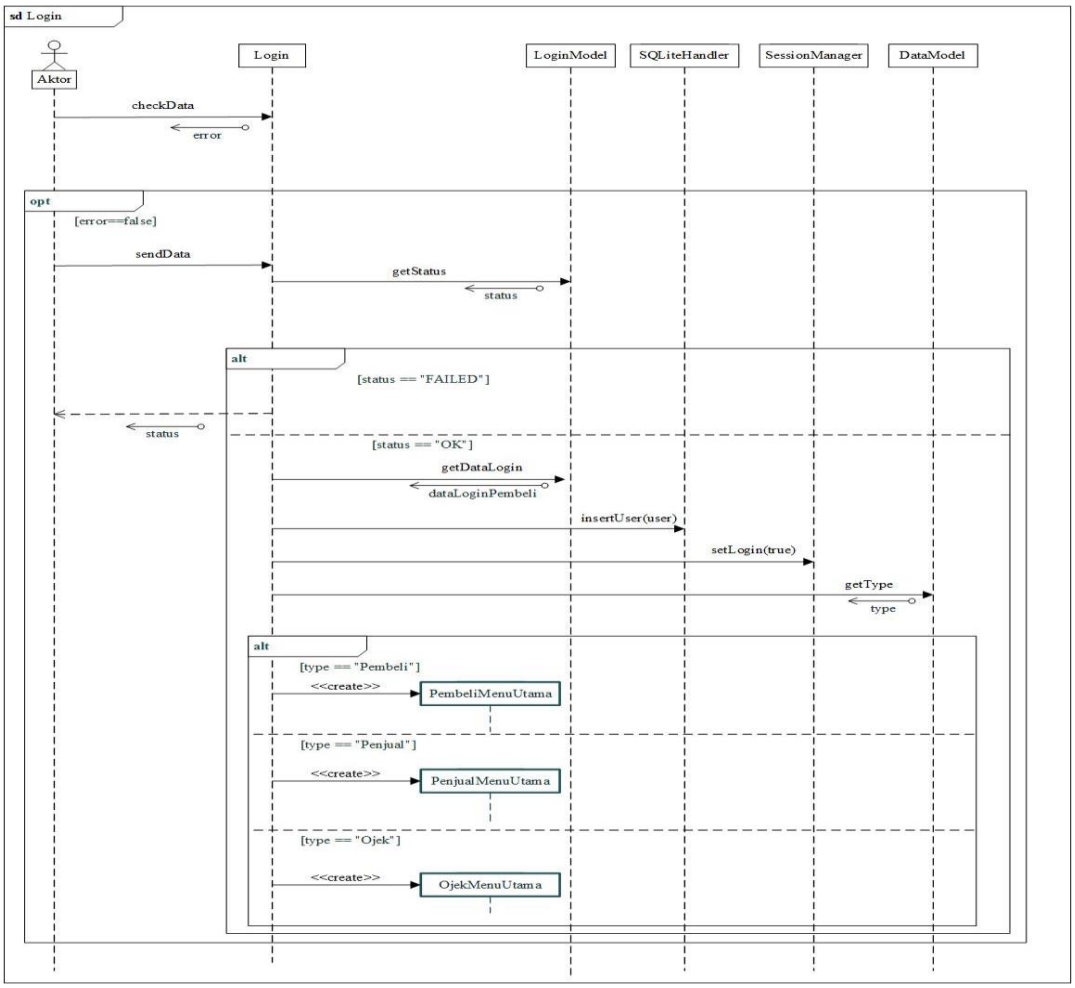


3.1.6.1.5 *Sequence* Diagram

Sequence Diagram merupakan gambaran interaksi masing-masing objek pada setiap *use case* dalam urutan waktu. Interaksi ini berupa pengiriman serangakain data antar objek-objek yang saling berinteraksi. Berikut ini adalah sequence diagram yang dapat digambarkan, diantaranya adalah:

1. *Sequence* Diagram Login

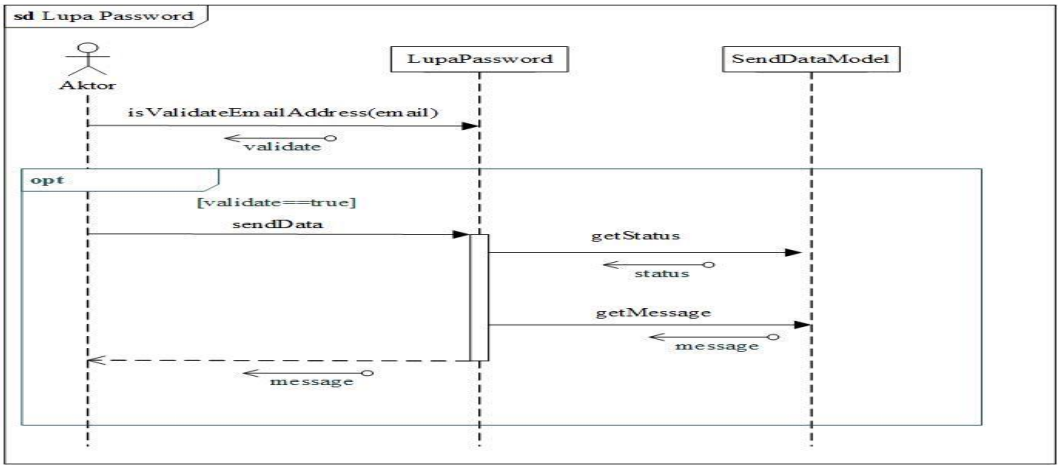
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses login. *Sequence* Diagram login dapat dilihat pada Gambar 3.32 *Sequence* Diagram Login.



Gambar 3.32 *Sequence* Diagram Login

2. *Sequence* Diagram Lupa Password

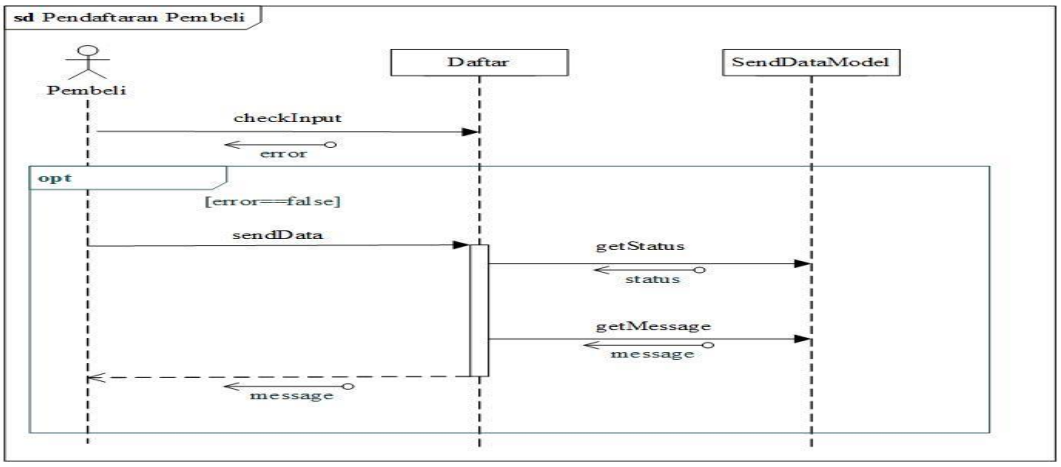
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses lupa password. *Sequence* Diagram lupa password dapat dilihat pada Gambar 3.33 *Sequence* Diagram Lupa Password.



Gambar 3.33 *Sequence* Diagram Lupa Password

3. *Sequence* Diagram Pendaftaran Pembeli

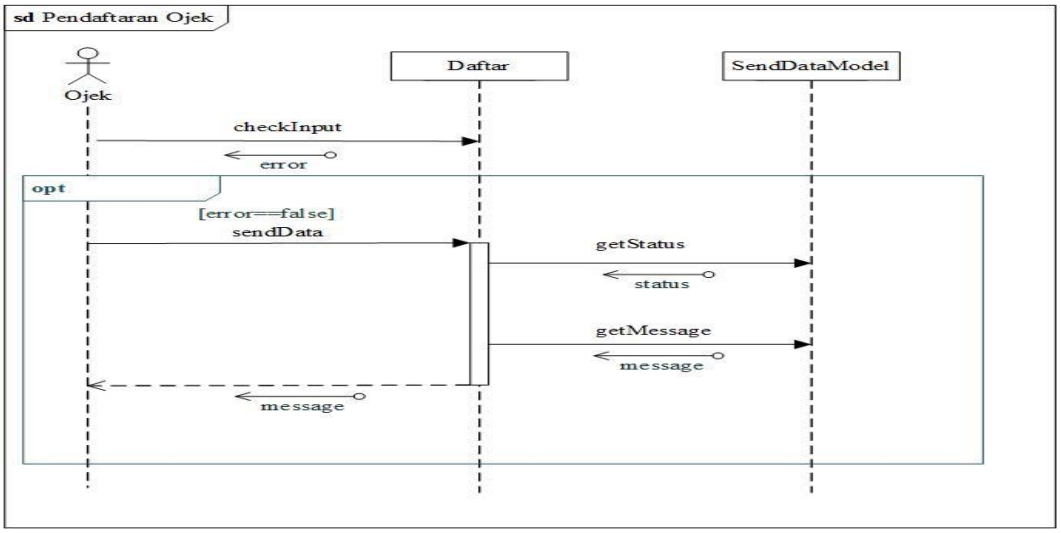
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses pendaftaran pembeli. *Sequence* Diagram pendaftaran pembeli dapat dilihat pada Gambar 3.34 *Sequence* Diagram Pendaftaran Pembeli.



Gambar 3.34 *Sequence* Diagram Pendaftaran Pembeli

4. *Sequence* Diagram Pendaftaran Ojek

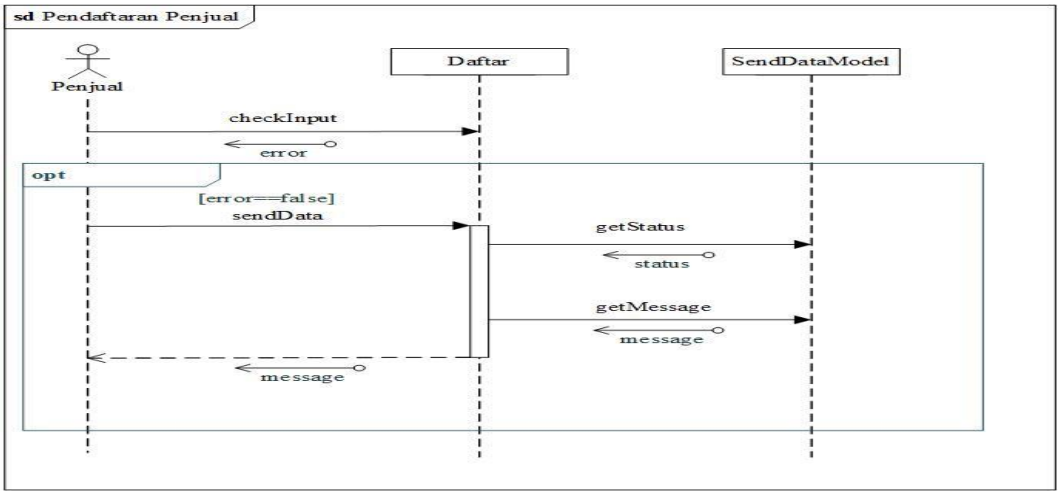
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses pendaftaran ojek. *Sequence* Diagram pendaftaran ojek dapat dilihat pada Gambar 3.35 *Sequence* Diagram Pendaftaran Ojek.



Gambar 3.35 *Sequence* Diagram Pendaftaran Ojek

5. *Sequence* Diagram Pendaftaran Penjual

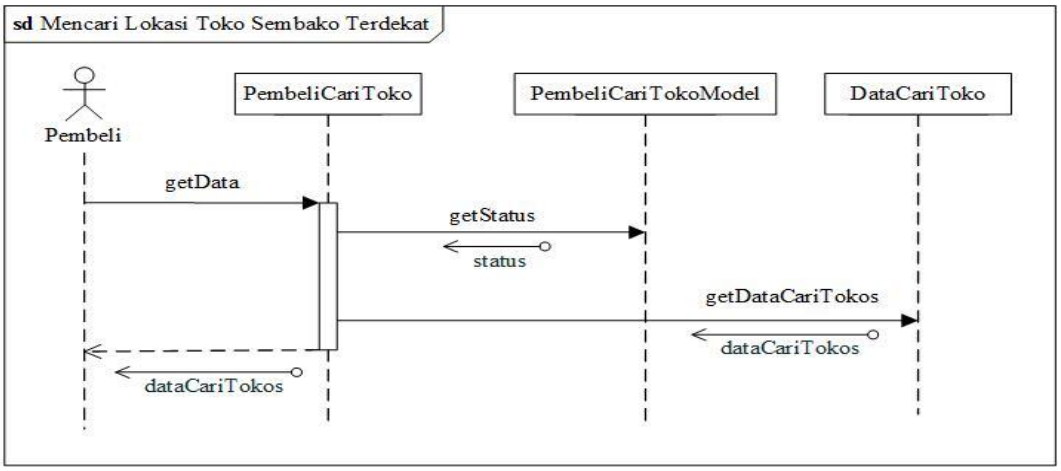
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses pendaftaran penjual. *Sequence* Diagram pendaftaran penjual dapat dilihat pada Gambar 3.36 *Sequence* Diagram Pendaftaran Penjual.



Gambar 3.36 *Sequence* Diagram Pendaftaran Penjual

6. *Sequence* Diagram Mencari Lokasi Toko Sembako Terdekat

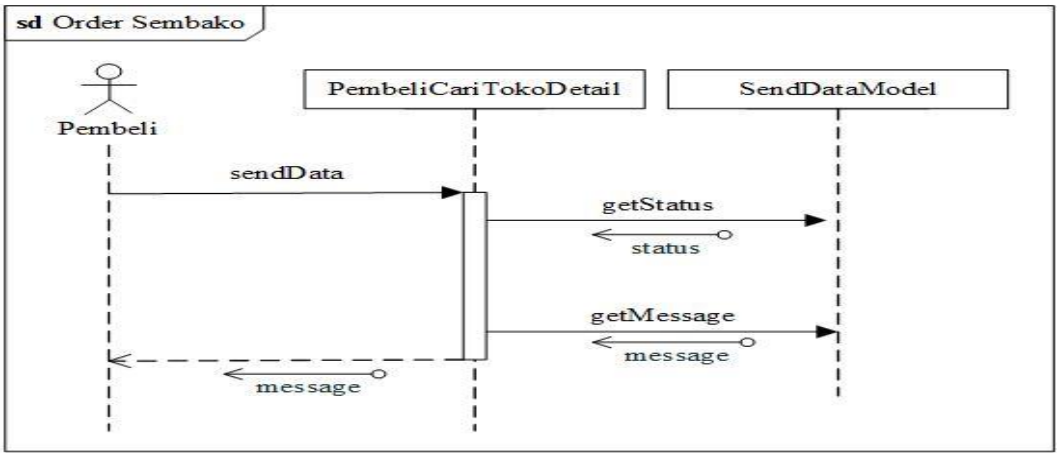
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses mencari lokasi toko sembako terdekat. *Sequence* Diagram Mencari Lokasi Toko Sembako Terdekat dapat dilihat pada Gambar 3.37 *Sequence* Diagram Mencari Lokasi Toko Sembako Terdekat.



Gambar 3.37 *Sequence* Diagram Mencari Lokasi Toko Sembako Terdekat

7. *Sequence* Diagram Order Sembako

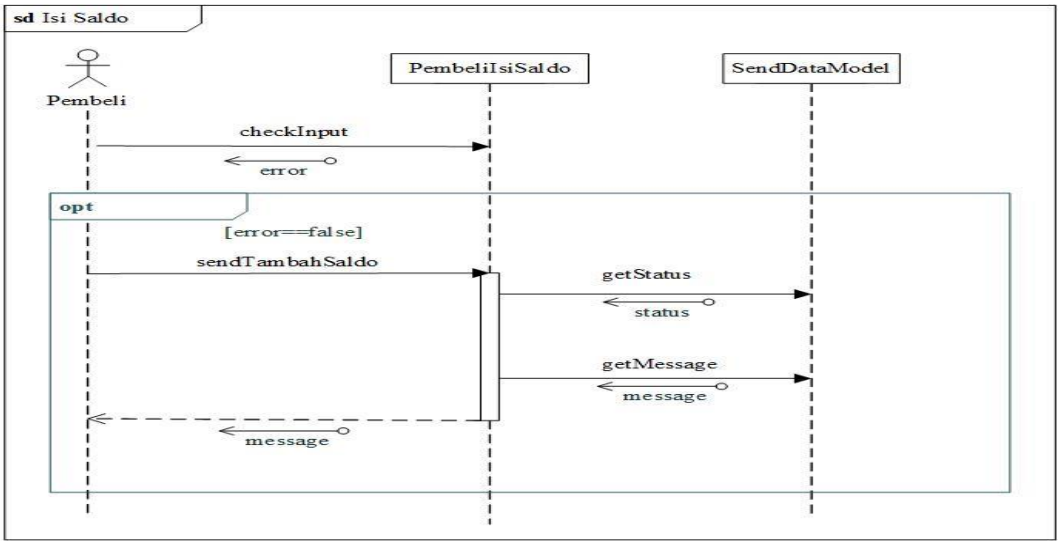
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses order sembako. *Sequence* Diagram Order Sembako dapat dilihat pada Gambar 3.38 *Sequence* Diagram Order Sembako.



Gambar 3.38 *Sequence* Diagram Order Sembako

8. *Sequence* Diagram Isi Saldo

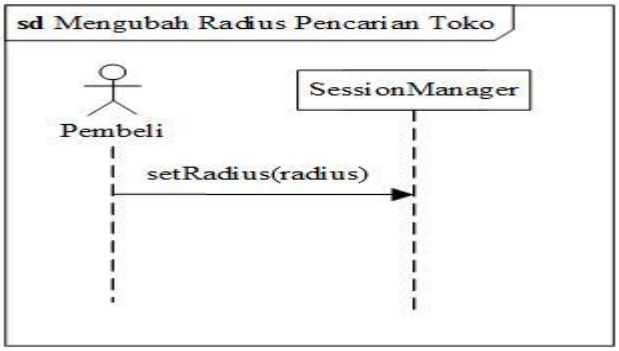
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses isi saldo. *Sequence* Diagram Isi Saldo dapat dilihat pada Gambar 3.39 *Sequence* Diagram Isi Saldo.



Gambar 3.39 *Sequence* Diagram Isi Saldo

9. *Sequence* Diagram Mengubah Radius Pencarian

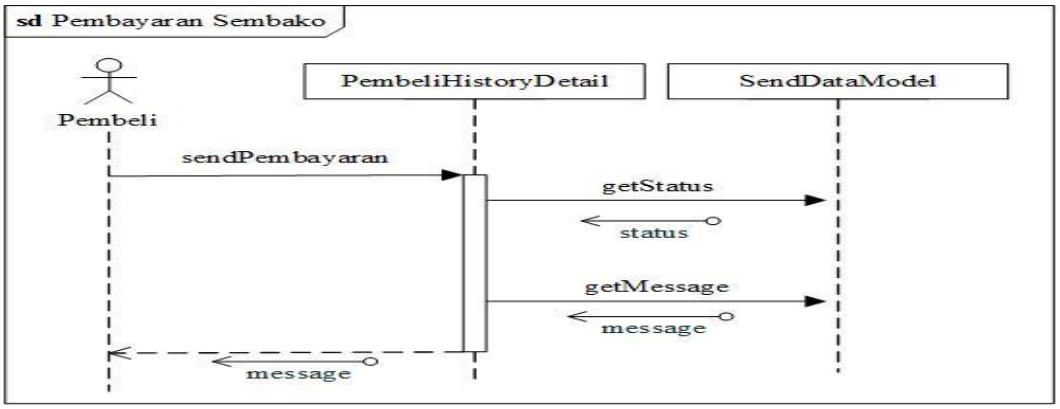
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses mengubah radius pencarian toko. *Sequence* Diagram Mengubah Radius Pencarian Toko dapat dilihat pada Gambar 3.40 *Sequence* Diagram Mengubah Radius Pencarian Toko.



Gambar 3.40 *Sequence* Diagram Mengubah Radius Pencarian Toko

10. *Sequence* Diagram Pembayaran Sembako

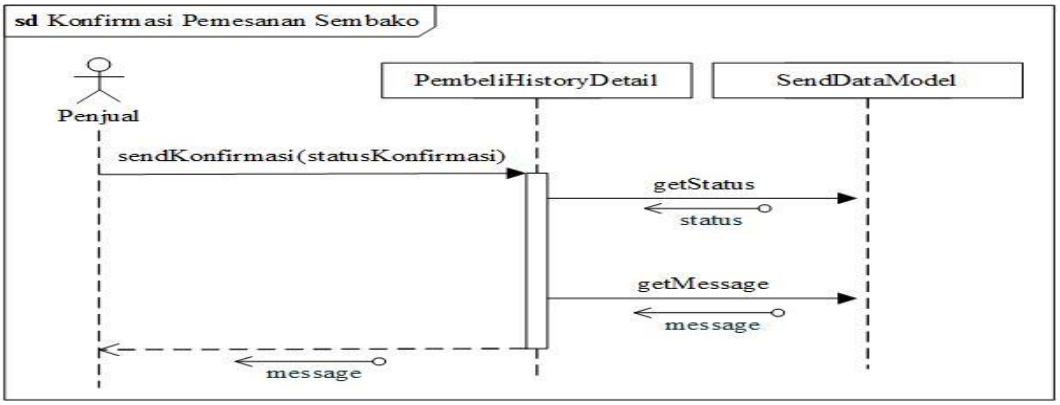
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses mengubah pembayaran sembako. *Sequence* Diagram Pembayaran Sembako dapat dilihat pada Gambar 3.41 *Sequence* Diagram Pembayaran Sembako.



Gambar 3.41 *Sequence* Diagram Pembayaran Sembako

11. *Sequence* Diagram Konfirmasi Pemesanan Sembako

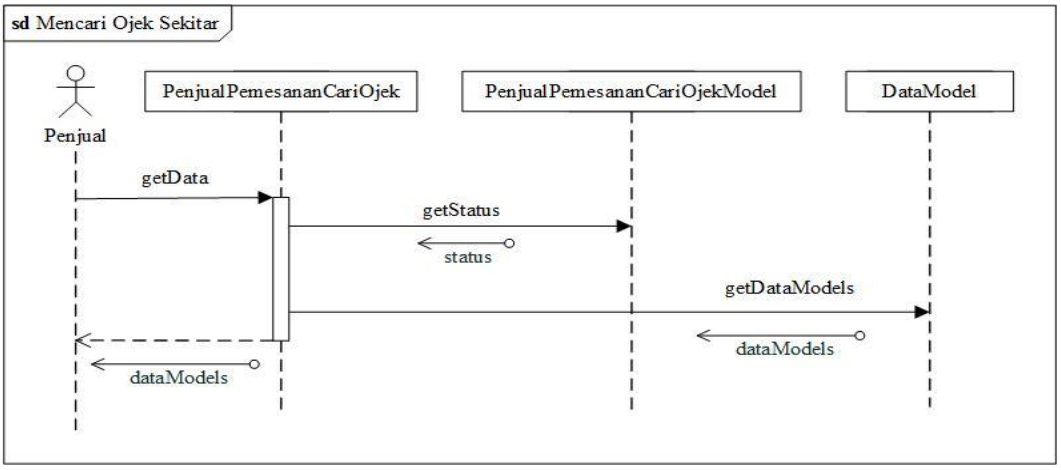
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses konfirmasi pembayaran sembako. *Sequence* Diagram Konfirmasi Pembayaran Sembako dapat dilihat pada Gambar 3.42 *Sequence* Diagram Konfirmasi Pemesanan Sembako.



Gambar 3.42 *Sequence* Diagram Konfirmasi Pemesanan Sembako

12. *Sequence* Diagram Mencari Ojek Di Sekitar

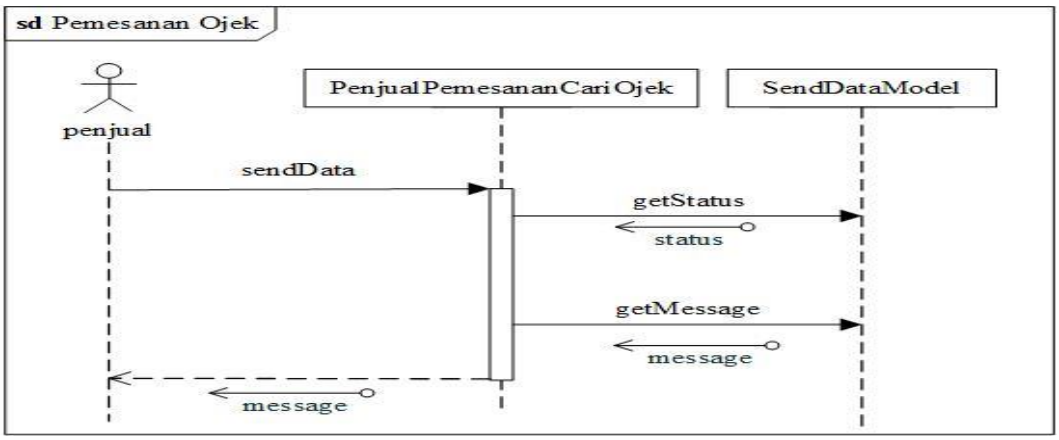
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses konfirmasi mencari ojek sekitar. *Sequence* Diagram Mencari Ojek Sekitar dapat dilihat pada Gambar 3.43 *Sequence* Diagram Mencari Ojek Sekitar.



Gambar 3.43 *Sequence* Diagram Mencari Ojek Sekitar

13. *Sequence* Diagram Pemesanan Ojek

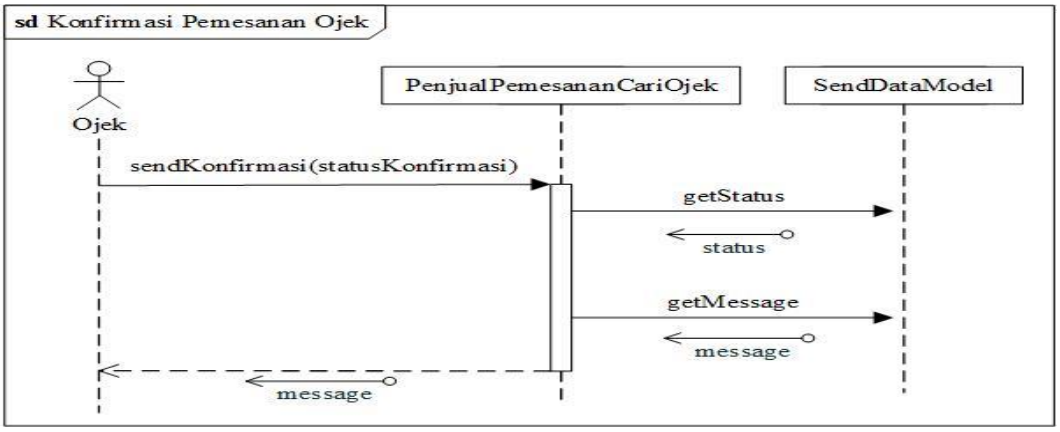
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses pemesanan ojek. *Sequence* Diagram Pemesanan Ojek dapat dilihat pada Gambar 3.44 *Sequence* Diagram Pemesanan Ojek.



Gambar 3.44 *Sequence* Diagram Pemesanan Ojek

14. *Sequence* Diagram Konfirmasi Pemesanan Ojek

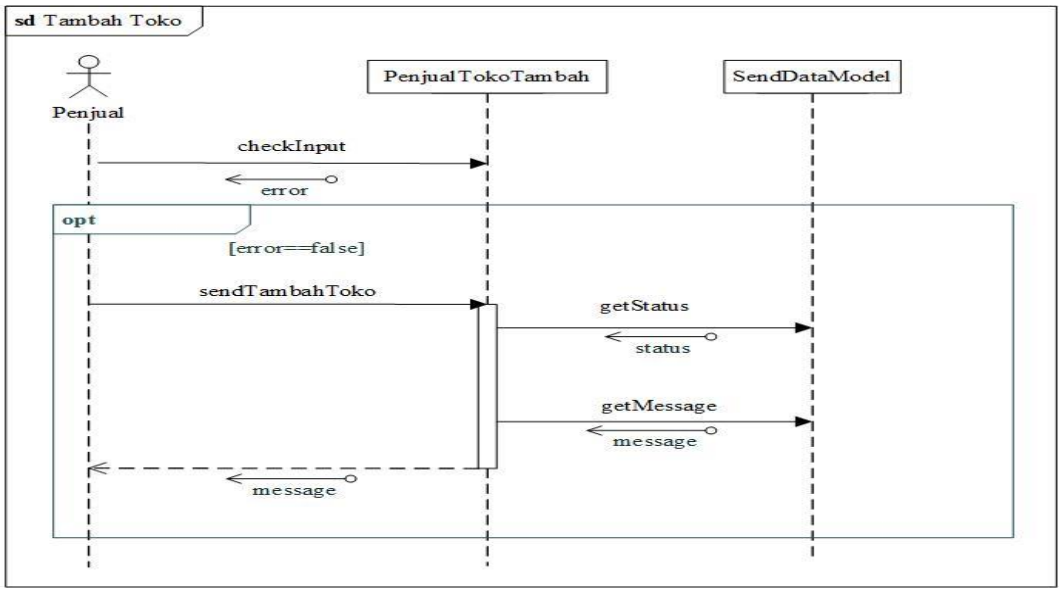
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses konfirmasi pemesanan ojek. *Sequence* Diagram Konfirmasi Pemesanan Ojek dapat dilihat pada Gambar 3.45 *Sequence* Diagram Konfirmasi Pemesanan Ojek.



Gambar 3.45 *Sequence* Diagram Konfirmasi Pemesanan Ojek

15. *Sequence* Diagram Tambah Toko

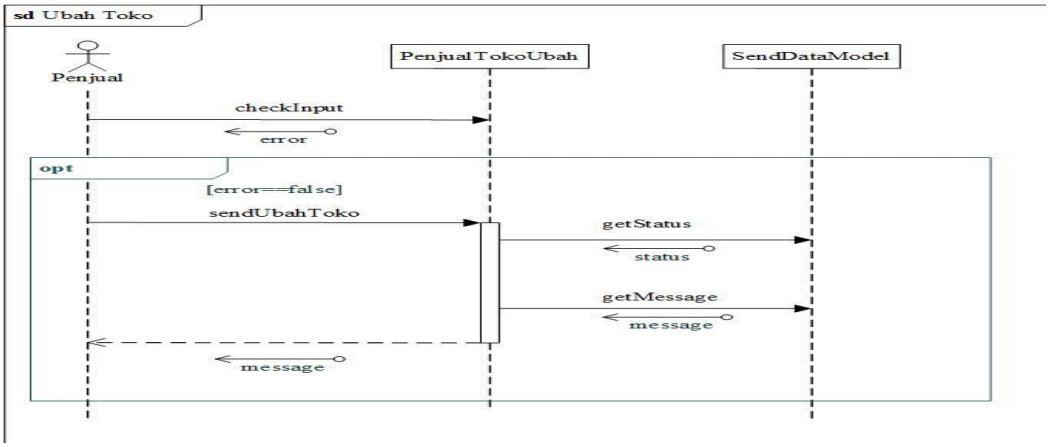
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses tambah data produk. *Sequence* Diagram Tambah Toko dapat dilihat pada Gambar 3.46 *Sequence* Diagram Tambah Toko.



Gambar 3.46 *Sequence* Diagram Tambah Toko

16. *Sequence* Diagram Ubah Toko

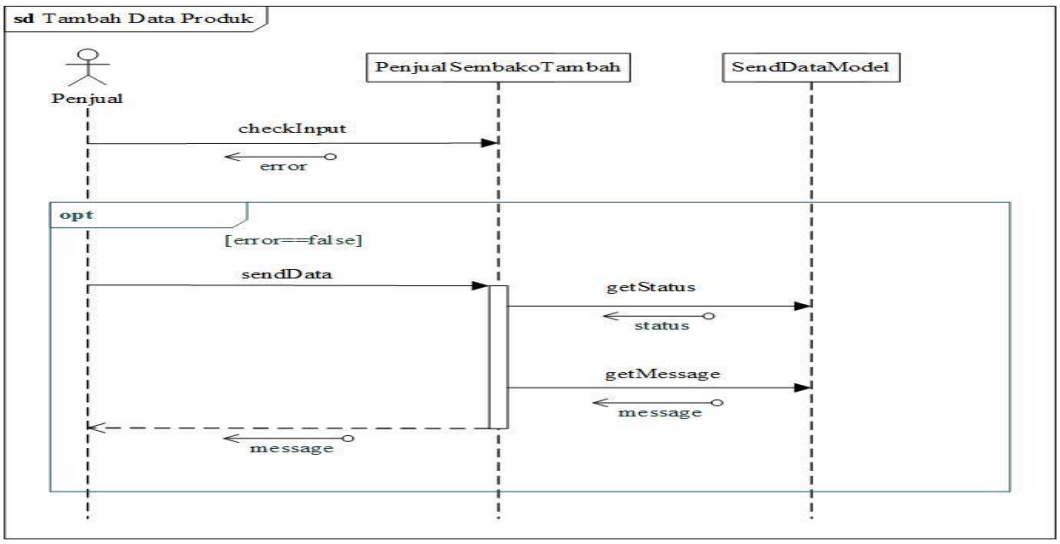
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses ubah data produk. *Sequence* Diagram Ubah Toko dapat dilihat pada Gambar 3.47 *Sequence* Diagram Ubah Toko.



Gambar 3.47 *Sequence* Diagram Ubah Toko

17. *Sequence* Diagram Tambah Data Produk

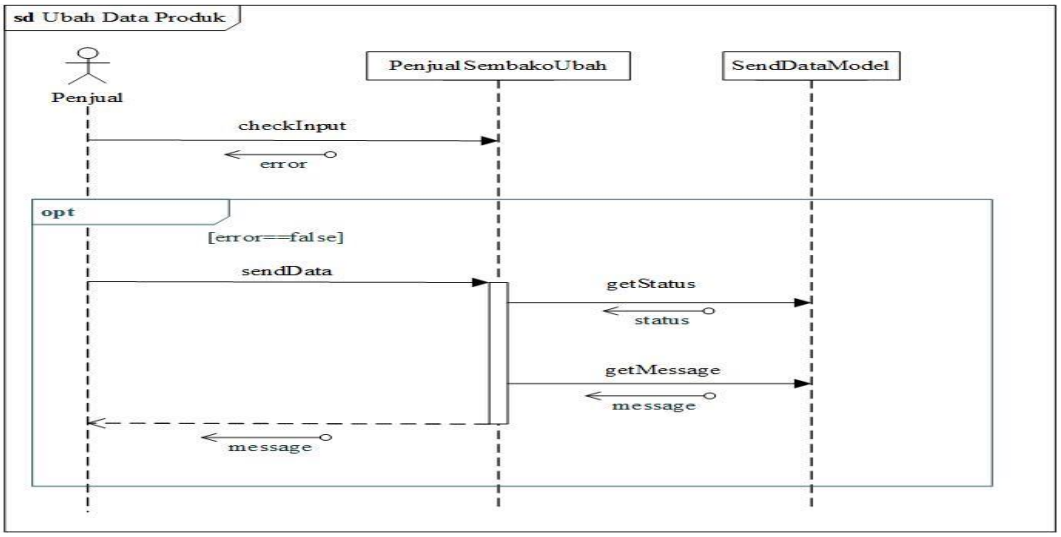
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses ubah data produk. *Sequence* Diagram Tambah Data Produk dapat dilihat pada Gambar 3.48 *Sequence* Diagram Tambah Data Produk.



Gambar 3.48 *Sequence* Diagram Tambah Data Produk

18. *Sequence* Diagram Ubah Data Produk

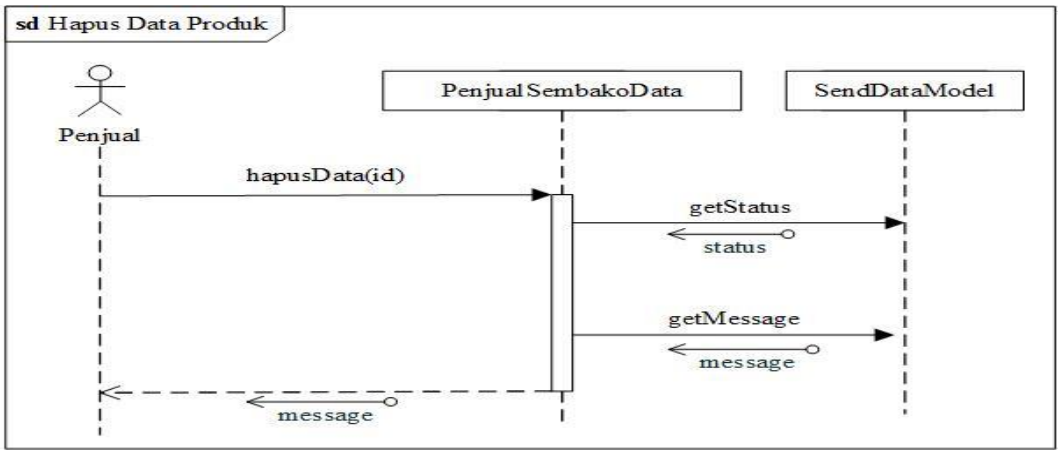
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses ubah data produk. *Sequence* Diagram Ubah Data Produk dapat dilihat pada Gambar 3.49 *Sequence* Diagram Ubah Data Produk.



Gambar 3.49 *Sequence* Diagram Ubah Data Produk

19. *Sequence* Diagram Hapus Data Produk

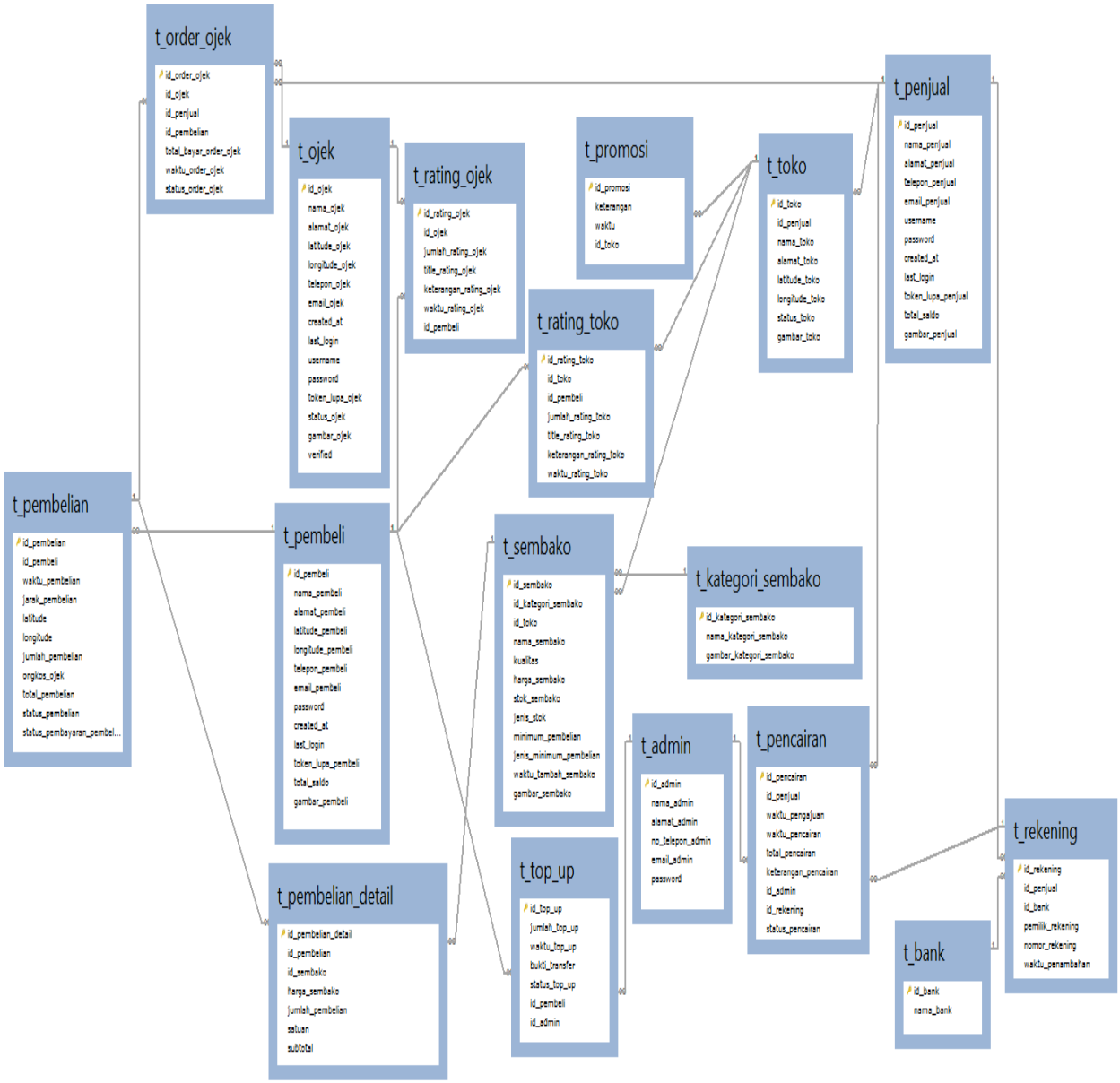
Berikut ini adalah interaksi antara aktor dengan objek-objek yang ada pada proses ubah data produk. *Sequence* Diagram Hapus Data Produk dapat dilihat pada Gambar 3.50 *Sequence* Diagram Hapus Data Produk.



Gambar 3.50 *Sequence* Diagram Hapus Data Produk

3.1.6.2 Skema Relasi

Skema Relasi merupakan rangkaian hubungan antara dua tabel atau lebih pada sistem basis data. Gambar 3.51 Skema Relasi merupakan gambaran rangkaian basis data pada pembangunan aplikasi pencarian toko sembako terdekat.



Gambar 3.51 Skema Relasi

3.1.6.2.1 Struktur Tabel

Struktur tabel menggambarkan detail yang berisi field, tipe data, pangang data, dan keterangan lainnya. Tabel yang terdapat pada basis data pembangunan sistem Pencarian Toko Sembako Terdekat ini adalah t_ojek, t_pembelian, t_top_up, t_rating_ojek, t_pembelian_detail, t_pembeli, t_bank, t_rekening, t_sembako, t_rating_toko, t_penjual, t_kategori_sembako, t_toko, t_order_ojek, t_admin, t_admin, t_pencairan.

1. Tabel t_ojek

Berikut adalah Tabel 3.28 Struktur Tabel t_ojek yang digunakan pada basis data perangkat lunak.

Tabel 3.28 Struktur Tabel t_ojek

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_ojek	Int	11	Tidak	Auto increment	Primary key
nama_ojek	varchar	100	Tidak		
alamat_ojek	varchar	100	Ya		
latitude_ojek	Double	-	Ya		
longitude_ojek	Double	-	Ya		
telepon_ojek	varchar	15	Ya		
email_ojek	varchar	100	Tidak	Unique	
created_at	datetime	-	Tidak		
last_login	datetime	-	Ya		
username	varchar	50	Tidak		
password	varchar	50	Tidak		
token_lupa_ojek	varchar	50	Ya		
status_ojek	Char	1	Ya		
gambar_ojek	varchar	30	Ya		

2. Tabel t_pembelian

Berikut adalah Tabel 3.29 Struktur Tabel t_pembelian yang digunakan pada basis data perangkat lunak.

Tabel 3.29 Struktur Tabel t_pembelian

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_pembelian	Int	11	Tidak	Auto increment	Primary key
id_pembeli	Int	11	Tidak		Foreign key references t_pembeli (id_pembeli)
waktu_pembelian	datetime	-	Tidak		
Jarak_pembelian	double	-	Ya		
lattitude	double	-	Ya		
longitude	double	-	Ya		
jumlah_pembelian	int	11	Ya		
total_pembelian	int	11	Ya		
status_pembelian	char	1	Ya		
Status_pembayaran_pembelian	char	1	Ya		

3. Tabel t_rating_ojek

Berikut adalah Tabel 3.30 Struktur Tabel t_rating_ojek yang digunakan pada basis data perangkat lunak.

Tabel 3.30 Struktur Tabel t_rating_ojek

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_rating_ojek	Int	11	Tidak	Auto increment	Primary key
id_ojek	Int	11	Tidak		Foreign key references t_ojek(id_ojek)
jumlah_rating_ojek	Double	-	Ya		
title_rating_ojek	varchar	20	Ya		
keterangan_rating_ojek	varchar	100	Ya		
waktu_rating_ojek	datetime	-	Ya		
id_pembeli	Int	11	Tidak		Foreign key references t_pembeli(id_pembeli)

4. Tabel t_pembelian_detail

Berikut adalah Tabel 3.31 Struktur Tabel t_pembelian_detail yang digunakan pada basis data perangkat lunak.

Tabel 3.31 Struktur Tabel t_pembelian_detail

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_pembelian_detail	Int	11	Tidak	Auto increment	Primary key
id_pembelian	Int	11	Tidak		Foreign key references t_pembelian(id_pembelian)
id_sembako	Int	11	Tidak		Foreign key references t_sembako(id_sembako)
jumlah_pembelian	Int	11	Tidak		
subtotal	Int	11	Tidak		

5. Tabel t_pembeli

Berikut adalah Tabel 3.32 Struktur Tabel t_pembeli yang digunakan pada basis data perangkat lunak.

Tabel 3.32 Struktur Tabel t_pembeli

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_pembeli	Int	11	Tidak	Auto increment	Primary key
nama_pembeli	Varchar	100	Tidak		
alamat_pembeli	Varchar	100	Tidak		
latitude_pembeli	double	-	Ya		
longitude_pembeli	double	-	Ya		
telepon_pembeli	varchar	15	Ya		
email_pembeli	varchar	100	Tidak	Unique	
password	varchar	50	Tidak		
created_at	datetime	-	Tidak		
last_login	datetime	-	Tidak		
token_lupa_pembeli	varchar	50	Ya		
total_saldo	int	11	Ya		
gambar_pembeli	varchar	30	Ya		

6. Tabel t_sembako

Berikut adalah Tabel 3.33 Struktur Tabel t_sembako yang digunakan pada basis data perangkat lunak.

Tabel 3.33 Struktur Tabel t_sembako

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_sembako	int	11	Tidak	Auto increment	Primary key
id_kategori_sembako	int	11	Tidak		Foreign key references t_kategori_sembako(id_kategori_sembako)
id_toko	int	11	Tidak		Foreign key references t_toko(id_toko)
nama_sembako	varchar	100	Tidak		
Kualitas	varchar	50	Ya		
harga_sembako	int	11	Tidak		
stok_sembako	int	11	Ya		
jenis_stok	varchar	10	Ya		
minimum_pembelian	int	11	Ya		
jenis_minimum_pembelian	varchar	10	Ya		
waktu_tambah_sembako	Datetime	-	Tidak		
gambar_sembako	varchar	30	Ya		

7. Tabel t_rating_toko

Berikut adalah Tabel 3.34 Struktur Tabel t_rating_toko yang digunakan pada basis data perangkat lunak.

Tabel 3.34 Struktur Tabel t_rating_toko

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_rating_toko	int	11	Tidak	Auto increment	Primary key
id_toko	int	11	Tidak		Foreign key references t_toko(id_toko)
id_pembeli	int	11	Tidak		Foreign key references t_pembeli(id_pembeli)
jumlah_rating_toko	double	-	Ya		
title_rating_toko	varchar	20	Ya		
keterangan_rating_toko	varchar	20	Ya		
waktu_rating_toko	datetime	-	Tidak		

8. Tabel t_top_up

Berikut adalah Tabel 3.35 Struktur Tabel t_top_up yang digunakan pada basis data perangkat lunak.

Tabel 3.35 Struktur Tabel t_top_up

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_top_up	int	11	Tidak	Auto increment	Primary key
jumlah_top_up	int	11	Tidak		
waktu_top_up	datetime	-	Tidak		
bukti_transfer	varchar	20	Tidak		
status_top_up	char	1	Ya		
id_pembeli	int	11	Tidak		Foreign key references t_pembeli(id_pembeli)
id_admin	int	11	Tidak		Foreign key references t_admin(id_admin)

9. Tabel t_kategori_sembako

Berikut adalah Tabel 3.36 Struktur Tabel t_kategori_sembako yang digunakan pada basis data perangkat lunak.

Tabel 3.36 Struktur Tabel t_kategori_sembako

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_kategori_sembako	int	11	Tidak	Auto increment	Primary key
nama_kategori_sembako	varchar	100	Tidak		
gambar_kategori_sembako	varchar	30	Ya		

10. Tabel t_toko

Berikut adalah Tabel 3.37 Struktur Tabel t_toko yang digunakan pada basis data perangkat lunak.

Tabel 3.37 Struktur Tabel t_toko

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_toko	int	11	Tidak	Auto increment	Primary key
id_penjual	int	11	Tidak		Foreign key references t_penjual(id_penjual)
nama_toko	varchar	50	Tidak		
alamat_toko	varchar	100	Tidak		
latitude_toko	double	-	Tidak		
langitude_toko	double	-	Tidak		
status_toko	char	1	Ya		
gambar_toko	varchar	30	Ya		

11. Tabel t_pencairan

Berikut adalah Tabel 3.38 Struktur Tabel t_pencairan yang digunakan pada basis data perangkat lunak.

Tabel 3.38 Struktur Tabel t_pencairan

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_riwayat_pencairan	int	11	Tidak	Auto increment	Primary key
id_penjual	int	11	Tidak		Foreign key references t_penjual(id_penjual)
waktu_pengajuan	datetime	-	Tidak		
waktu_pencairan	datetime	-	Ya		
total_pencairan	int	11	Tidak		
keterangan_pencairan	varchar	100	Ya		
id_admin	int	11	Tidak		Foreign key references t_admin (id_admin)
id_rekening	int	11	Ya		Foreign key references t_rekening (id_rekening)
status_pencairan	char	1	Tidak		

12. Tabel t_admin

Berikut adalah Tabel 3.39 Struktur Tabel t_admin yang digunakan pada basis data perangkat lunak.

Tabel 3.39 Struktur Tabel t_admin

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_admin	int	11	Tidak	Auto increment	Primary key
nama_admin	varchar	50	Tidak		
alamat_admin	varchar	100	Tidak		
no_telepon_admin	varchar	15	Ya		
email_admin	varchar	100	Tidak	Unique	
Password	varchar	100	Tidak		

13. Tabel t_penjual

Berikut adalah Tabel 3.40 Struktur Tabel t_penjual yang digunakan pada basis data perangkat lunak.

Tabel 3.40 Struktur Tabel t_penjual

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_penjual	int	11	Tidak	Auto increment	Primary key
nama_penjual	varchar	100	Tidak		
alamat_penjual	varchar	15	Tidak		
telepon_penjual	varchar	100	Ya		
email_penjual	varchar	50	Tidak	Unique	
username	varchar	50	Tidak		
password	varchar	50	Tidak		
created_at	datetime	-	Tidak		
last_login	datetime	-	Ya		
token_lupa_penjual	varchar	50	Ya		
total_saldo	int	11	Ya		
gambar_penjual	varchar	30	Ya		

14. Tabel t_bank

Berikut adalah Tabel 3.41 Struktur Tabel t_bank yang digunakan pada basis data perangkat lunak.

Tabel 3.41 Struktur Tabel t_bank

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_bank	Int	11	Tidak	Auto increment	Primary key
nama_bank	Varchar	50	Tidak		

15. Tabel t_rekening

Berikut adalah Tabel 3.42 Struktur Tabel t_rekening yang digunakan pada basis data perangkat lunak.

Tabel 3.42 Struktur Tabel t_rekening

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_rekening	Int	11	Tidak	Auto increment	Primary key
id_penjual	Int	11	Tidak		Foreign key references t_penjual(id_penjual)
id_bank	Int	11	Tidak		Foreign key references t_bank(id_bank)
pemilik_rekening	varchar	100	Tidak		
nomor_rekening	varchar	20	Tidak		
waktu_penambahan	datetime	-	Tidak		

16. Tabel t_order_ojek

Berikut adalah Tabel 3.43 Struktur Tabel t_order_ojek yang digunakan pada basis data perangkat lunak.

Tabel 3.43 Struktur Tabel t_order_ojek

Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_order_ojek	Int	11	Tidak	Auto increment	Primary key
id_ojek	Int	11	Tidak		Foreign key references t_ojekl(id_ojek)
id_penjual	Int	11	Tidak		Foreign key references t_penjual(id_penjual)
id_pembelian	Int	11	Tidak		Foreign key references t_pembelian(id_pembelian)
total_bayar_order_ojek	Int	11	Tidak		
waktu_order_ojek	datetime	-	Tidak		
status_order_ojek	char	1	Tidak		

17. Tabel t_promosi

Berikut adalah Tabel 3.44 Struktur Tabel t_promosi yang digunakan pada basis data perangkat lunak.

Tabel 3.44 Struktur Tabel t_promosi

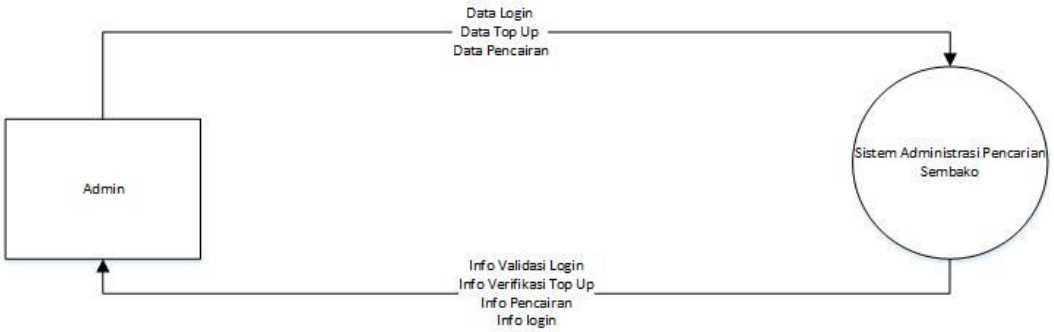
Nama Field	Tipe Data	Panjang	Null	Ekstra	Kunci
id_promosi	int	11	Tidak	Auto increment	Primary key
keterangan	varchar	50	Tidak		
waktu	datetime	-	Tidak		
id_toko	int	11	Tidak		Foreign key references t_toko(id_toko)

3.1.6.3 Analisis Kebutuhan Fungsional di *Platform Web*

Analisis kebutuhan fungsional di platform web menggunakan analisis terstruktur yang meliputi Diagram Konteks, Data Flow Diagram, Spesifikasi Proses dan Kamus data.

3.1.6.3.1 Diagram Konteks

Diagram konteks merupakan salah satu alat bantu dalam melakukan analisis terstruktur. Diagram konteks ini menggambarkan suatu sistem secara garis besarnya dan keseluruhannya saja. Dalam diagram konteks juga digambarkan entitas eksternal yang merupakan brainware yang menghasilkan data yangakan diolah sistem maupun tujuan dari informasi yang dihasilkan oleh sistem. Adapun diagram konteks yang diusulkan ditunjukan oleh Gambar 3.52 Diagram Konteks.



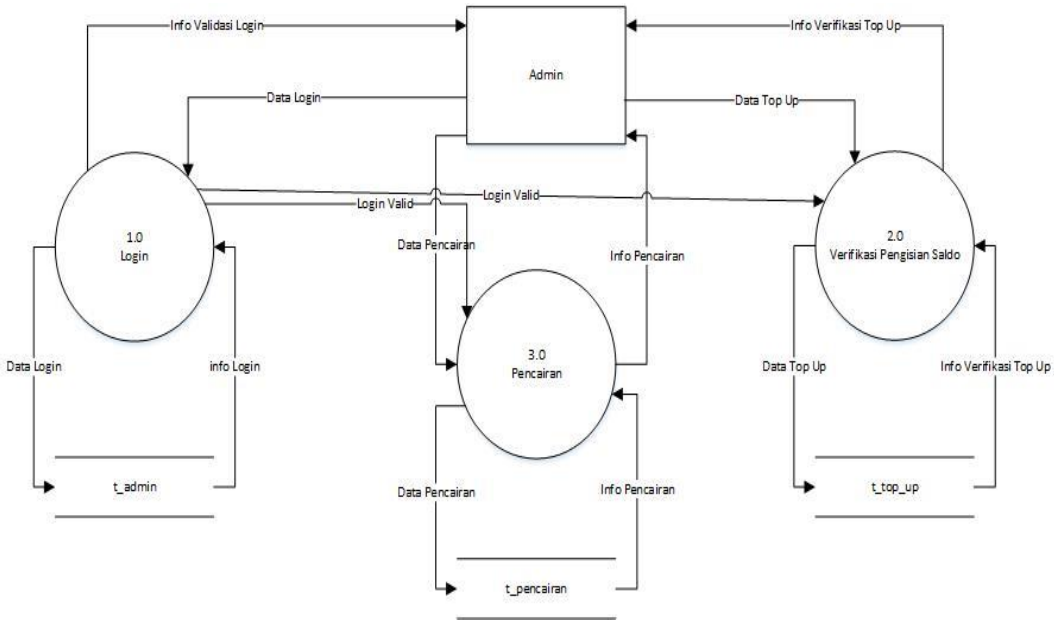
Gambar 3.52 Diagram Konteks

3.1.6.3.2 *Data Flow Diagram* (DFD)

Data Flow Diagram merupakan sebuah representasi dari suatu sistem yang menggambarkan bagian-bagian dari sistem yang akan dibangun beserta keterkaitan antara bagian-bagian yang ada. Dari *Data Flow Diagram* ini seseorang bisa mengetahui sumber dari informasi di dalam sistem maupun tujuan dari masukan yang berasal dari entitas eksternal. Adapun *Data Flow Diagram* pada pembangunan sistem di *platform website* adalah sebagai berikut.

1. DFD Level 1

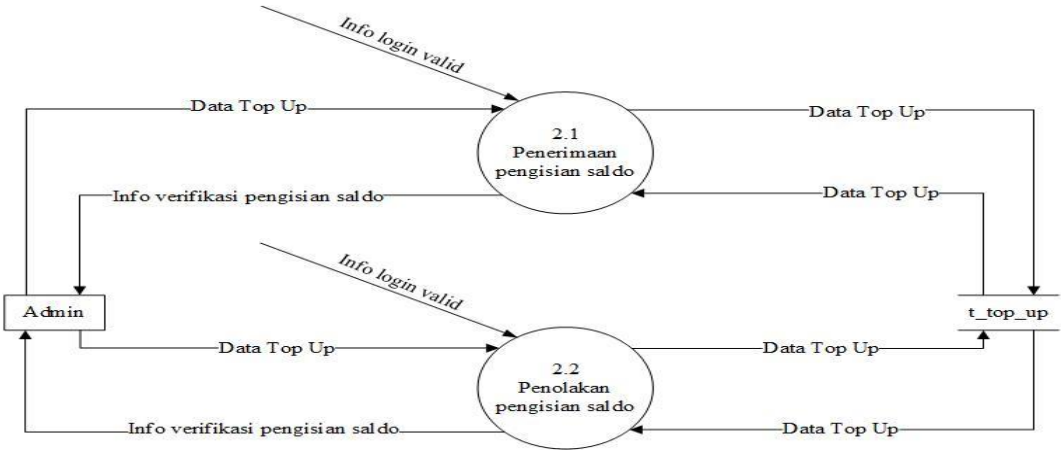
Berikut ini adalah *data flow diagram* level 1 ditunjukan pada Gambar 3.53 DFD Level 1 dibawah ini.



Gambar 3.53 DFD Level 1

2. DFD Level 2 Verifikasi Pengisian Saldo

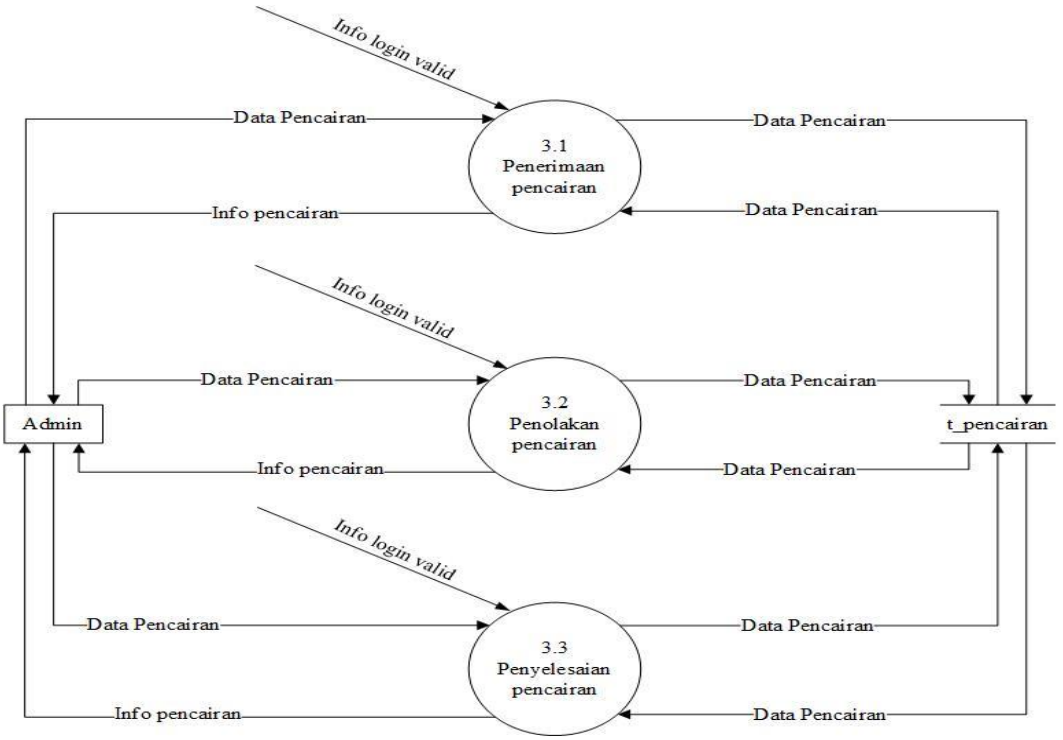
Berikut ini adalah *data flow diagram* level 2 proses 2.0 Verifikasi Pengisian Saldo ditunjukan pada Gambar 3.54 DFD Level 2 proses 2.0 Verifikasi Pengisian Saldo dibawah ini.



Gambar 3.54 DFD Level 2 Verifikasi Pengisian Saldo

3. DFD Level 2 Pencairan

Berikut ini adalah *data flow diagram* level 2 proses 3.0 Pencairan ditunjukan pada Gambar 3.49 DFD Level 2 Pencairan dibawah ini.



Gambar 3.55 DFD Level 2 Pencairan

3.1.6.3.3 Spesifikasi Proses

Spesifikasi proses digunakan untuk menggambarkan proses model aliran yang terdapat pada *Data Flow Diagram* (DFD). Berikut ini spesifikasi proses pada pembangunan sistem di *platform web*.

1. Spesifikasi Proses DFD Level 1

Spesifikasi proses DFD level 1 digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD level 1.

Tabel 3.45 Spesifikasi Proses Level 1

No	Proses	Keterangan
1	No. Proses	1.0
	Nama Proses	Login
	Source	Admin
	Input	Username dan password
	Output	Info Login
	Destination	Admin
	Logika Proses	1. Admin memasukandata login pada halaman login. 2. Sistem melalukan validasi login
2	No. Proses	2.0
	Nama Proses	Verifikasi Pengisian Saldo
	Source	Admin
	Input	Data Top Up
	Output	Info Verifikasi Top Up
	Destination	Admin
	Logika Proses	1. Admin verifikasi top up 2. Sistem mengubah status top up
3	No. Proses	3.0
	Nama Proses	Pencairan
	Source	Admin
	Input	Data Pencairan
	Output	Info Pencairan
	Destination	Admin

	Logika Proses	1. Admin mengolah data pencairan 2. Sistem mengubah status pencairan

2. Spesifikasi Proses DFD Level 2 Login

Spesifikasi proses DFD level 2 Login digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD level 2 Login.

Tabel 3.46 Spesifikasi Proses Level 2 Login

No	Proses	Keterangan
1	No. Proses	1.1
	Nama Proses	Validasi Login Gagal
	Source	Admin
	Input	Username dan password
	Output	Infovalidasi login “gagal”
	Destination	Admin
	Logika Proses	1. Admin memasukan data login 2. Sistem melalukan validasi datalogin
2	No. Proses	1.2
	Nama Proses	Validasi Login Berhasil
	Source	Admin
	Input	Username dan password
	Output	Info validasi login “berhasil”
	Destination	Admin
	Logika Proses	1. Admin memasukan data login 2. Sistem melakukan validasi data login

3. Spesifikasi Proses DFD Level 2 Verifikasi Pengisian Saldo

Spesifikasi proses DFD level 2 Verifikasi Pengisian Saldo digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD level 2 Verifikasi Login.

Tabel 3.47 Spesifikasi Proses Level 2 Verifikasi Pengisian Saldo

No	Proses	Keterangan
1	No. Proses	2.1
	Nama Proses	View data nominal pengisian
	Source	Admin
	Input	Data top up
	Output	Info top up
	Destination	Admin
	Logika Proses	1. Admin memilih menu top up 2. Sistem menampilkan data top up
2	No. Proses	2.2
	Nama Proses	Verifikasi pengisian saldo ditolak
	Source	Admin
	Input	Data top up
	Output	Info verifikasi top up “ditolak”
	Destination	Admin
	Logika Proses	1. Admin memilih data top up 2. Sistem menampilkan detail data top up 3. Admin menolak top up 4. Sistem mengubah status top up
3	No. Proses	2.3
	Nama Proses	Verifikasi pengisian saldo diterima
	Source	Admin
	Input	Data top up
	Output	Info verifikasi top up “diterima”
	Destination	Admin
	Logika Proses	1. Admin memilih data top up 2. Sistem menampilkan detail data top up 3. Admin menerima top up 4. Sistem mengubah status top up

4. Spesifikasi proses DFD level 2 Pencairan

Spesifikasi Proses DFD Level 2 Pencairan digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD level 2 Pencairan.

Tabel 3.48 Spesifikasi Proses Level 2 Pencairan

No	Proses	Keterangan
1	No. Proses	3.1
	Nama Proses	View Data Top Up
	Source	Admin
	Input	Data pencairan
	Output	Info pencairan
	Destination	Admin
	Logika Proses	1. Admin memilih menu pencairan 2. Sistem menampilkan data pencairan
2	No. Proses	3.2
	Nama Proses	Pencairan ditolak
	Source	Admin
	Input	Data pencairan
	Output	Info pencairan “ditolak”
	Destination	Admin
	Logika Proses	1. Admin memilih data pencairan 2. Sistem menampilkan data detail pencairan 3. Admin menolak pencairan 4. Sistem mengubah status pencairan
3	No. Proses	3.3
	Nama Proses	Pencairan diterima
	Source	Admin
	Input	Data pencairan
	Output	Info pencairan “diterima”
	Destination	Admin
	Logika Proses	1. Admin memilih data pencairan 2. Sistem menampilkan data detail pencairan 3. Admin menerima pencairan

		4. Sistem mengubah status pencairan

3.1.6.3.4 Kamus Data

Kamus data merupakan deskripsi formal untuk menjelaskan semua data yang tercangkup dalam DFD, maka digunakan kamus data. Kamus data yang terdapat pada sistem yang dibuat dapat dilihat pada Table 3.48 Kamus Data.

Tabel 3.49 Kamus Data

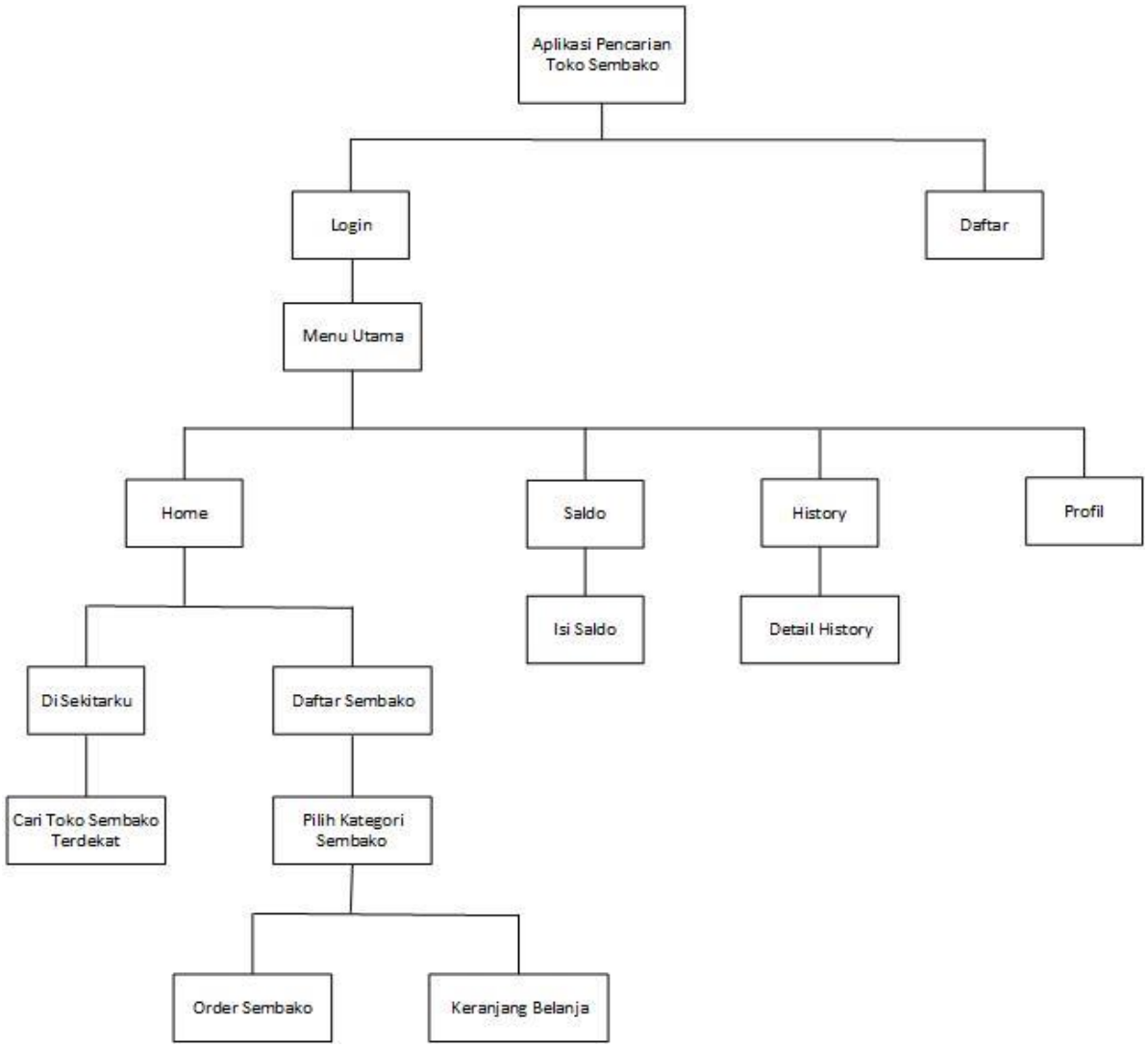
Nama Aliran Data	Data Login
Where used/how used	Proses 1.0
Deskripsi	Berisi data login
Struktur data	email_admin + password
email_admin	[A-Z|a-z|0-100]
Password	[[A-Z|a-z|0-50]
Nama Aliran Data	Data Top Up
Where used/how used	Proses 2.0, proses 2.1, proses 2.2
Deskripsi	Berisi data top up
Struktur data	id_top_up + status_top_up
id_top_up	[0-11]
status_top_up	[0-1]
Nama Aliran Data	Data Pencairan
Where used/how used	Proses 3.0, proses 3.1, proses 3.2, proses 3.3
Deskripsi	Berisi data pencairan
Struktur data	id_pencairan + status_pencairan
id_pencairan	[0-11]
status_pencairan	[0-1]

3.1.6.4 Struktur Menu

Perancangan struktur menu merupakan gambaran jalur pemakaian aplikasi sehingga aplikasi yang mudah dipahami dan mudah digunakan. Perancangan struktur menu dari pembangunan aplikasi pencarian toko sembako dapat dilihat sebagai berikut :

1. Struktur Menu *Platform Mobile* Pembeli

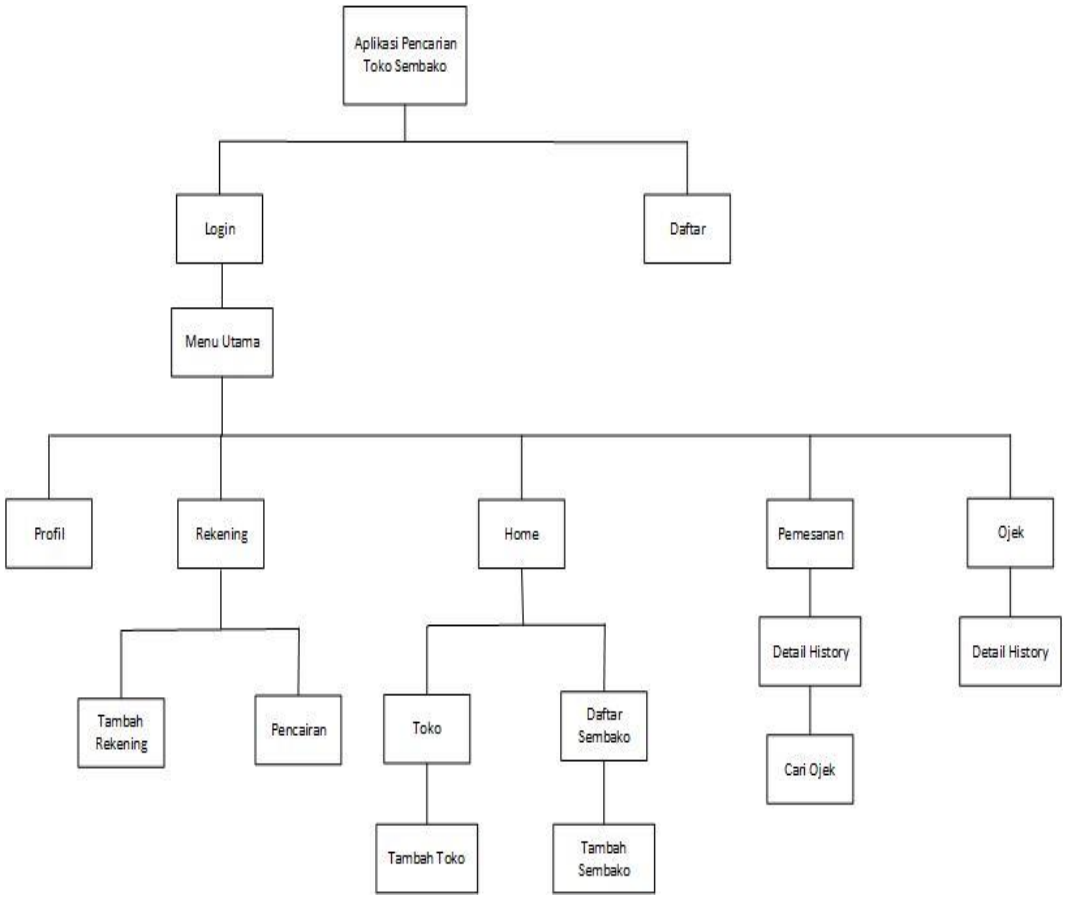
   Berikut ini adalah Gambar 3.56 Struktur Menu *Platform Mobile* Pembeli :



Gambar 3.56 Struktur Menu *Platform Mobile* Pembeli

2. Struktur Menu Platform Mobile Penjual

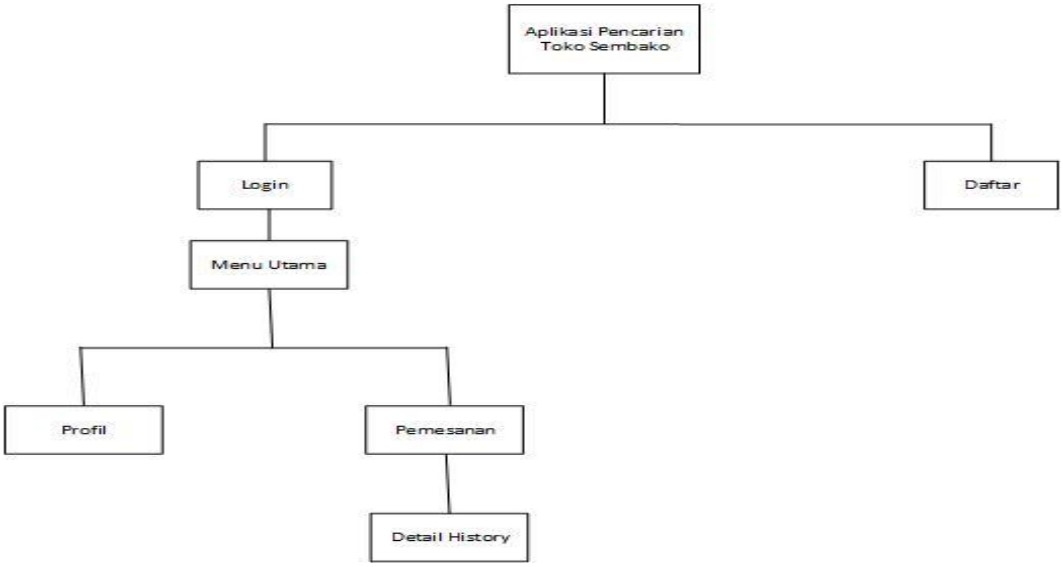
   Berikut ini adalah Gambar 3.57 Struktur Menu *Platform Mobile* Penjual :



Gambar 3.57 Struktur Menu *Platform Mobile* Penjual

3. Struktur Menu Platform Mobile Ojek

   Berikut ini adalah Gambar 3.58 Struktur Menu *Platform Mobile* Ojek :



Gambar 3.58 Struktur Menu *Platform Mobile* Ojek

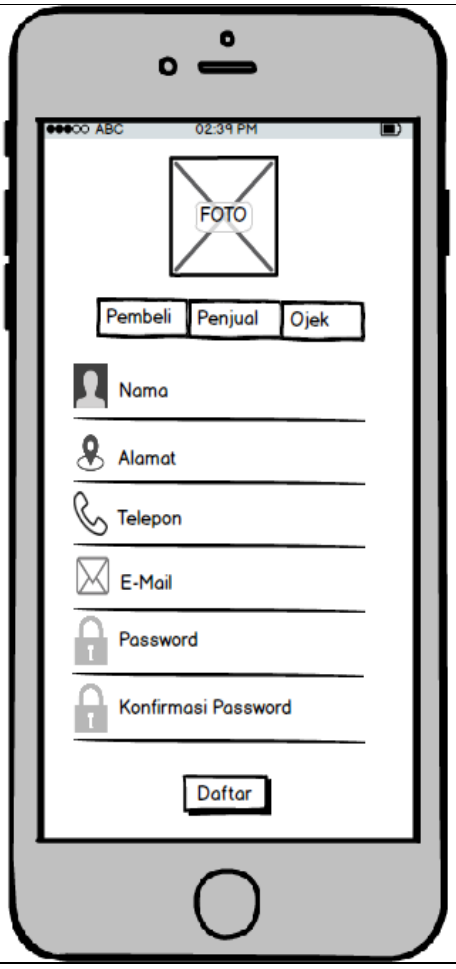
3.1.6.5 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun. Perancangan ini diimplementasikan menjadi sebuah program yang utuh dan dapat digunakan oleh pengguna sistem. Rancangan antarmuka pembangunanaplikasipencarian toko sembako terdekat adalah sebagai berikut :

1. Perancangan Antarmuka Mobile Pembeli

a. Perancangan Antarmuka Daftar

Berikut adalah Gambar 3.59 Antarmuka M01 Daftar yang menggambarkan rancangan antarmuka Daftar.

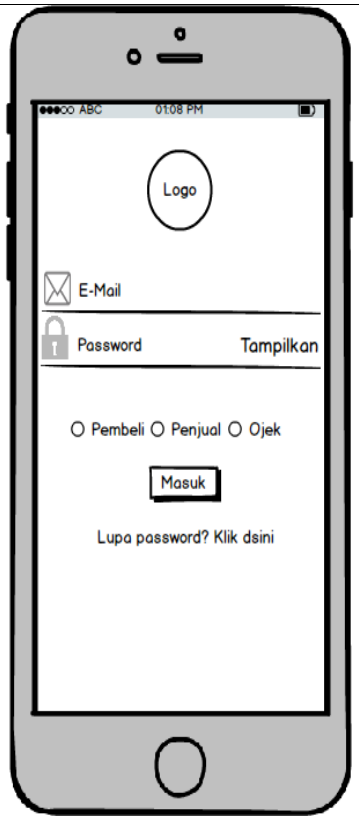
M01 Daftar	
	1. Data diri diisi terlebih dahulu 2. Tekan Daftarbila telah selesai mengisi data 3. bila data diisi dengan benar akan masuk ke M02 4. bila salah mengisi data akan menampilkan pesan PM01 dan kembali ke M01

Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.59 Antarmuka M01 Daftar

b. Perancangan Antarmuka Login

Berikut adalah Gambar 3.60 Antarmuka M.02 Login yang menggambarkan rancangan antarmuka Login.

M02 Login	
	1. Email dan Password di isi 2. Pilih salah satu radio button misalnya Anda login sebagai pembeli, Anda harus mengklik radio buttan pembeli 3. Setelah selesai lalu tekan masukkemudian akan menampilkan M03 4. bila salah mengisi data akan menampilkan pesan PM02 dan kembali ke M02
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.60 Antarmuka M.02 Login

c. Perancangan Antarmuka Menu Utama Pembeli

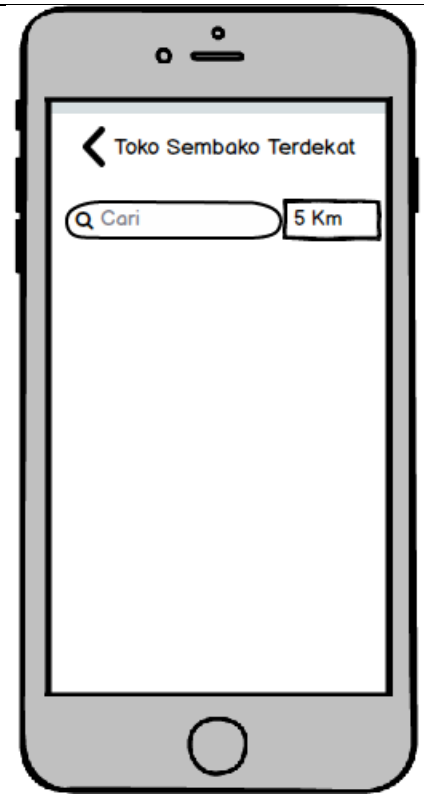
Berikut adalah Gambar 3.61 Antarmuka M03 Menu Utama Pembeli yang menggambarkan rancangan antarmuka Menu Utama Pembeli.

M03 Menu Utama Pembeli	
	1. M03 menampilkan Menu Utama Pembeli 2. Tekan disekitarku bila ingin menentukan radius pencarian 3. Kemudian akan masuk ke M04
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.61 Antarmuka M03 Menu Utama Pembeli

d. Perancangan Antarmuka Cari Toko

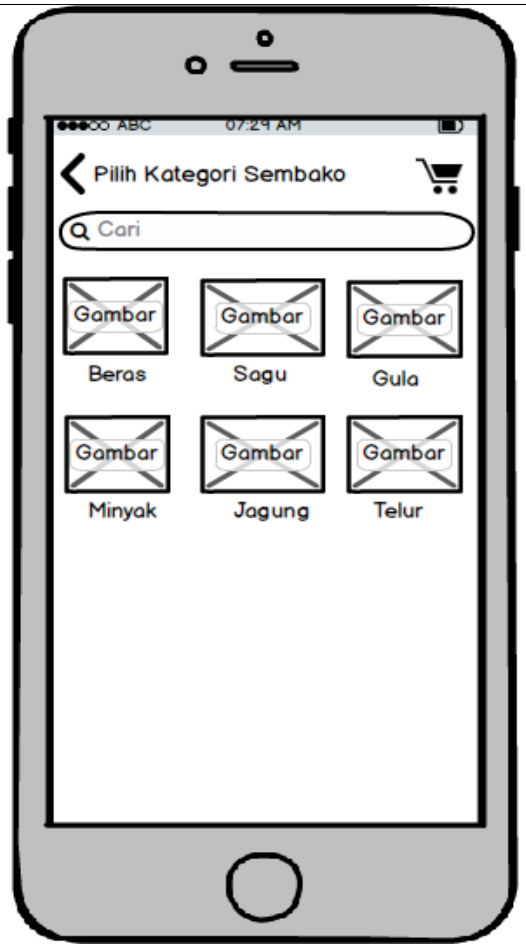
Berikut adalah Gambar 3.62 Antarmuka M04 Cari Toko yang menggambarkan rancangan antarmuka Cari Toko.

<table>
<tr><td>M04 Cari Toko</td><td></td></tr>
<tr><td>
</td><td>1. Tekan radius untuk menentukan radius yang dinginkan
2. Tekan cari untuk mecari daerah toko sembako yang di inginkan
3. Bila tidak ditemukan toko sembako maka akan menampilkan pesan kesalahan PM03
4. Tekan gambar panah ke samping untuk kembali ke M03</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.62 Antarmuka M04 Cari Toko

e. Perancangan Antarmuka Menu Daftar Sembako

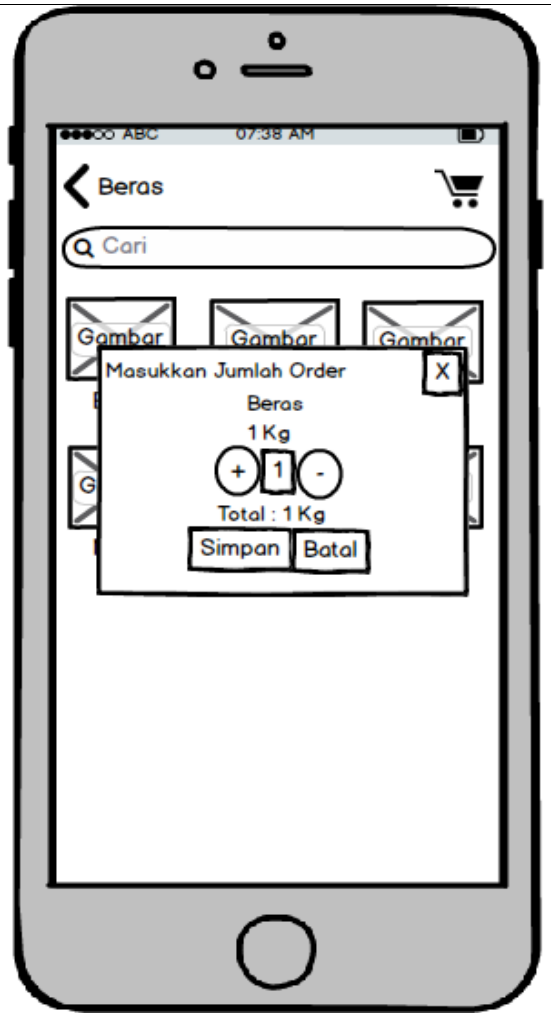
Berikut adalah Gambar 3.63 Antarmuka M05 Menu Daftar Sembako yang menggambarkan rancangan antarmuka Menu Daftar Sembako.

<table>
<tr><td>M05 Menu Daftar Sembako</td><td></td></tr>
<tr><td>
</td><td>1. Tekan cari untuk mencari kategori sembako
2. Pilih salah satu kategori untuk melakukan order sembako
3. Setelah memilih kategori sembako akan menampilkan M06</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.63 Antarmuka M05 Daftar Sembako

f. Perancangan Antarmuka Menu Order Sembako

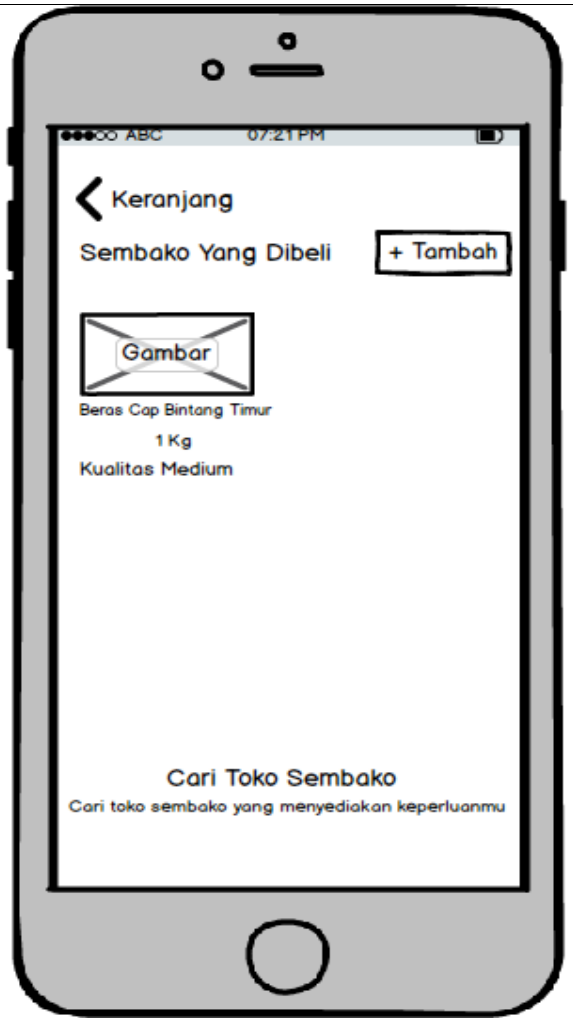
Berikut adalah Gambar 3.64 Antarmuka M06 Menu Order Sembako yang menggambarkan rancangan antarmuka Menu Order Sembako.

M06 Menu Order Sembako	
	1. Pilih salah satu kategori sembako kemudian akan menampilkan M06 2. Setelah melakukan order sembako tekan simpan maka akan masuk ke M07 Menu Keranjang
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.64 Antarmuka M06 Menu Order Sembako

g. Perancangan Antarmuka Menu Keranjang

Berikut adalah Gambar 3.65 Antarmuka M07Menu Keranjang yang menggambarkan rancangan antarmuka Menu Keranjang.

<table>
<tr><td>M07 Menu Keranjang</td><td></td></tr>
<tr><td>
</td><td>1. M07Menu Keranjang akan menampilkan sembako yang dibeli
2. Bila ingin menambahkan order sembako cukup klik Tambah di pojok kanan atas
3. Setelah selesai melakukan order sembako, maka untuk mencari toko sembako klik Cari Toko Sembako di paling bawah kemudian akan masuk ke M04
4. kemudian jika ingin melihat history order sembako klik panah yang ada di pojok kiri atas kemudian akan masuk ke menu utama M03
5. Bila ingin melihat history order sembako pilih History di Menu Utama kemudian akan masuk ke M08</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.65 Antarmuka M07 Menu Keranjang

h. Perancangan Antarmuka Menu History Belanja

Berikut adalah Gambar 3.66 Antarmuka M08 Menu History Belanja yang menggambarkan rancangan antarmuka Menu History Belanja.

M08 Menu History Belanja	
	1. M08 Menu History Belanja akan menampilkan history belanja beserta keterangannya 2. Bila ingin menambahkan saldo klik menu saldo kemudian akan masuk ke M09 Menu Saldo.
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.66 Antarmuka M08 Menu History Belanja

i. Perancangan Antarmuka Menu Saldo

Berikut adalah Gambar 3.67 Antarmuka M09 Menu Saldo yang menggambarkan rancangan antarmuka Menu Saldo.

M09 Menu Saldo	
	1. M09 Menu Saldo akan menampilkan keterangan saldo yang dimiliki 2. bila ingin menambah saldo klik Tambah di pojok kanan atas. 3. Kemudian akan masuk ke M10 Menu Isi Saldo
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.67 Antarmuka M09 Menu Saldo

j. Perancangan Antarmuka Menu Isi Saldo

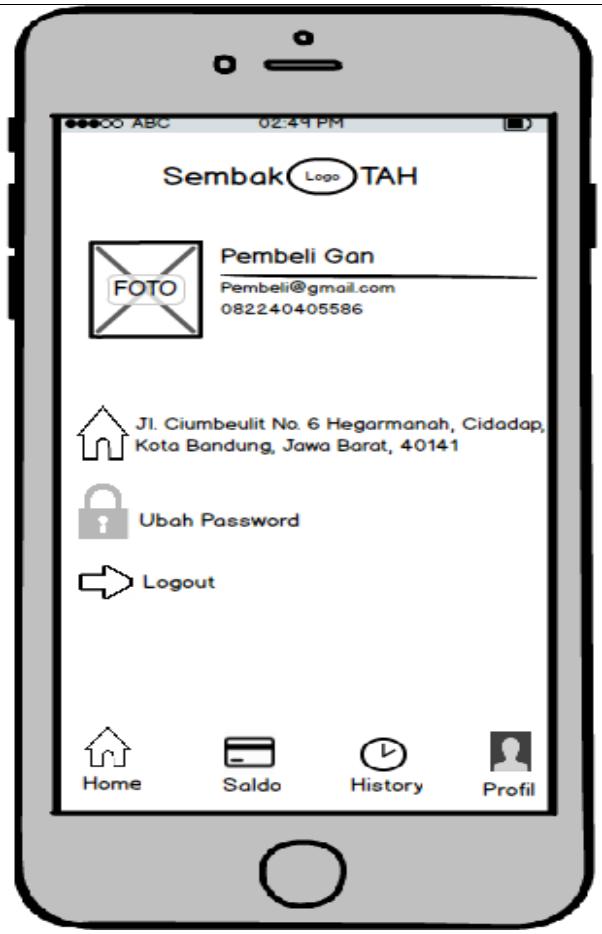
Berikut adalah Gambar 3.68 Antarmuka M1 0Menu Isi Saldo yang menggambarkan rancangan antarmuka Menu Isi Saldo.

M10 Menu Isi Saldo	
	1. M10 Menu Isi Saldo akan menampilkan data yang harus di isi bila ingin menambah saldo 2. Setelah semua data di isi klik Tambah kemudian akan kembali ke M09 bila ingin melihat riwayat pengisian saldo
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.68 Antarmuka M10 Menu Isi Saldo

k. Perancangan Angtarmuka Menu Profil

Berikut adalah Gambar 3.69 Antarmuka M11 Menu Profil yang menggambarkan rancangan antarmuka Menu Profil.

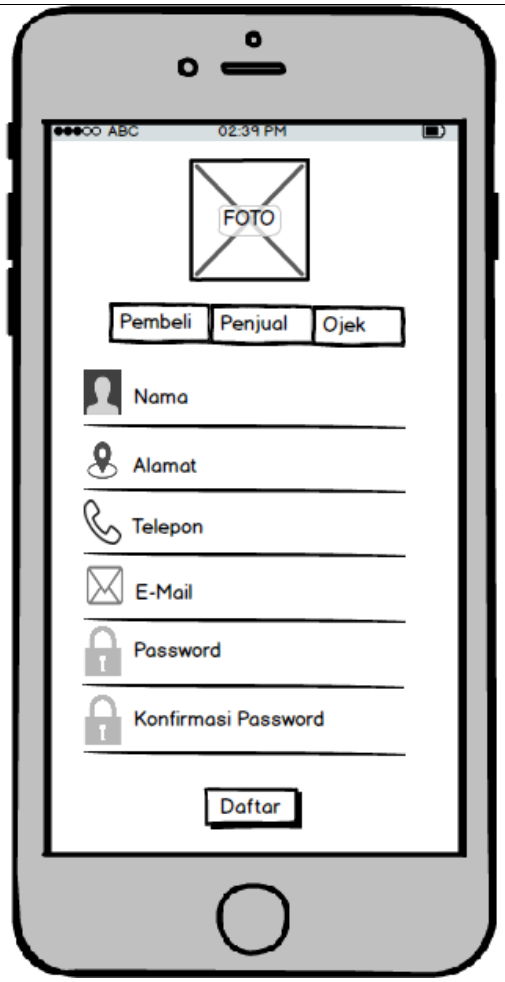
M11 Menu Profil	
	1. M11 Menu Profil akan menampilkan profil pembeli
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.69 Antarmuka M11 Menu Profil

2. Perancangan Antarmuka Mobile Penjual

a. Perancangan Antarmuka Daftar

Berikut adalah Gambar 3.70 Antarmuka M01 Daftar yang menggambarkan rancangan antarmuka Daftar.

<table>
<tr><td>M01 Daftar</td><td></td></tr>
<tr><td>
</td><td>1. Data diri diisi terlebih dahulu
2. Tekan Daftar bila telah selesai mengisi data
3. bila data diisi dengan benar akan masuk ke M02
4. bila salah mengisi data akan menampilkan pesan PM01 dan kembali ke M01</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.70 Antarmuka M01 Daftar

b. Perancangan Antarmuka Login

Berikut adalah Gambar 3.71 Antarmuka M.02 Login yang menggambarkan rancangan antarmuka Login.

M02 Login	
	1. Email dan Password di isi 2. Pilih salah satu radio button misalnya Anda login sebagai penjual, Anda harus mengklik radio button penjual 3. Setelah selesai lalu tekan masukkemudian akan menampilkan M03 4. bila salah mengisi data akan menampilkan pesan PM02 dan kembali ke M02
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.71 Antarmuka M.02 Login

c. Perancangan Antarmuka Menu Utama Penjual

Berikut adalah Gambar 3.72 Antarmuka M03 Menu Utama Penjual yang menggambarkan rancangan antarmuka Menu Utama Penjual.

M03 Menu Utama Penjual	
	1. M03 Menu Utama akan menampilkan Menu Utama Penjual 2. Tekan Toko bila ingin melihat toko yang ada disekitar 3. Kemudian akan masuk ke M04
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.72 Antarmuka M03 Menu Utama Penjual

d. Perancangan Antarmuka Toko Yang Ada Di Sekitar

Berikut adalah Gambar 3.73 Antarmuka M04 Toko Yang Ada Di Sekitaryang menggambarkan rancangan antarmuka Toko Yang Ada Di Sekitar.

M04 Toko Yang Ada Di Sekitar	
	1. M04 menampilkan Toko Yang Ada Di Sekitar 2. Tekan Simbol + yang ada di pojok kanan atas bila ingin menambahkan toko 3. Kemudian akan masuk ke M05
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.73 Antarmuka M04 Toko Yang Ada Di Sekitar

e. Perancangan Antarmuka Tambah Toko

Berikut adalah Gambar 3.74 Antarmuka M05 Tambah Toko yang menggambarkan rancangan antarmuka Tambah Toko.

<table>
<tr><td>M05 Tambah Toko</td><td></td></tr>
<tr><td>
</td><td>1. M05 menampilkan Data bila ingin menambah Toko
2. Tekan Tambah bila sudah mengisi semua data
3. Kemudian akan masuk ke M04</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.74 Antarmuka M05 Tambah Toko

f. Perancangan Antarmuka Menu Daftar Sembako

Berikut adalah Gambar 3.75 Antarmuka M06 Menu Daftar Sembako yang menggambarkan rancangan antarmuka Menu Daftar Sembako.

M06 Menu Daftar Sembako	
	1. M06 Menu Daftar Sembako akan menampilkan data kategori sembako. 2. Tekan simbol “+” untuk menambah data sembako 3. Kemudian akan masuk ke M07
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.75 Antarmuka M06 Menu Daftar Sembako

g. Perancangan Antarmuka Tambah Sembako

Berikut adalah Gambar 3.76 Antarmuka M07 Tambah Sembako yang menggambarkan rancangan antarmuka Tambah Sembako.

<table>
<tr><td>M07 Tambah Sembako</td><td></td></tr>
<tr><td>
</td><td>1. isi data untuk menambah sembako
2. setelah selesai mengisi semua data tekan tambah untuk menambahkan data sembako</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.76 Antarmuka M07 Tambah Sembako

h. Perancangan Antarmuka Menu Rekening

Berikut adalah Gambar 3.77 Antarmuka M08 Menu Rekening yang menggambarkan rancangan antarmuka Menu Rekening.

M08 Menu Rekening	
	1. M08 Menu Rekening akan menampilkan data rekening yang dimiliki 2. Tekan Tambah bila ingin menambah rekening 3. Kemudian akan masuk ke M09
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.77 Antarmuka M08 Menu Rekening

i. Perancangan Antarmuka Tambah Rekening

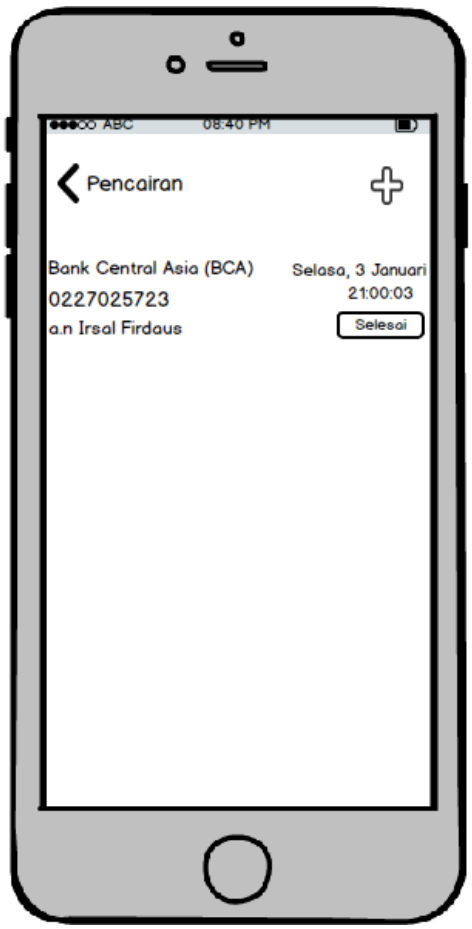
Berikut adalah Gambar 3.78 Antarmuka M09 Tambah Rekening yang menggambarkan rancangan antarmuka Tambah Rekening.

<table>
<tr><td>M09 Tambah Rekening</td><td></td></tr>
<tr><td>
</td><td>1. M09 menampilkan data tambah rekening
2. Isi data terlenih dahulu
3. bila semua data telah di isi semuatekan Tambah untuk menambah rekening</td></tr>
<tr><td>Ukuran : Disesuaikan
Warna font : Disesuaikan
Font Style : Disesuaikan
Font Size : Disesuaikan</td><td></td></tr>
</table>

Gambar 3.78 Antarmuka M09 Tambah Rekening

j. Perancangan Antarmuka Pencairan

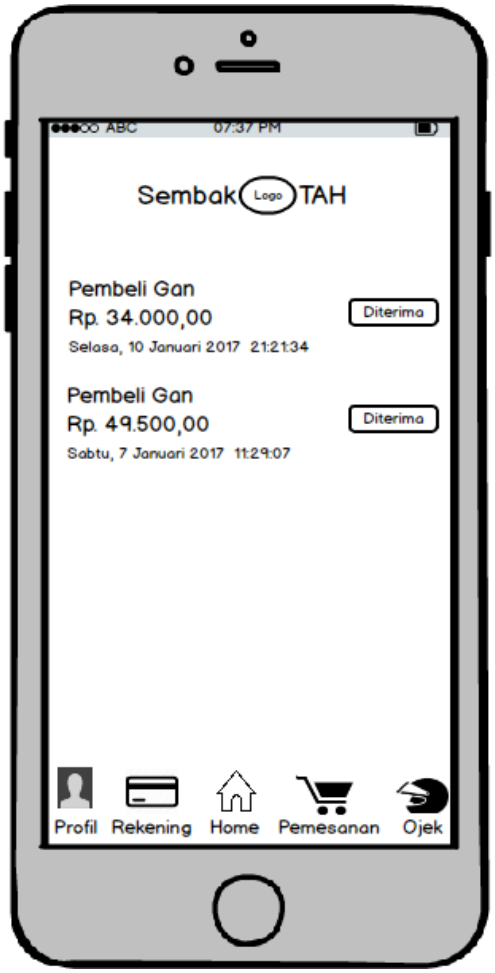
Berikut adalah Gambar 3.79 Antarmuka M10 Pencairan yang menggambarkan rancangan antarmuka Pencairan.

M10 Pencairan	
	1. M10 menampilkan data Pencairan 2. Klik simbol "+" untuk melakukan pencairan
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.79 Antarmuka M10 Pencairan

k. Perancangan Antarmuka Pemesanan

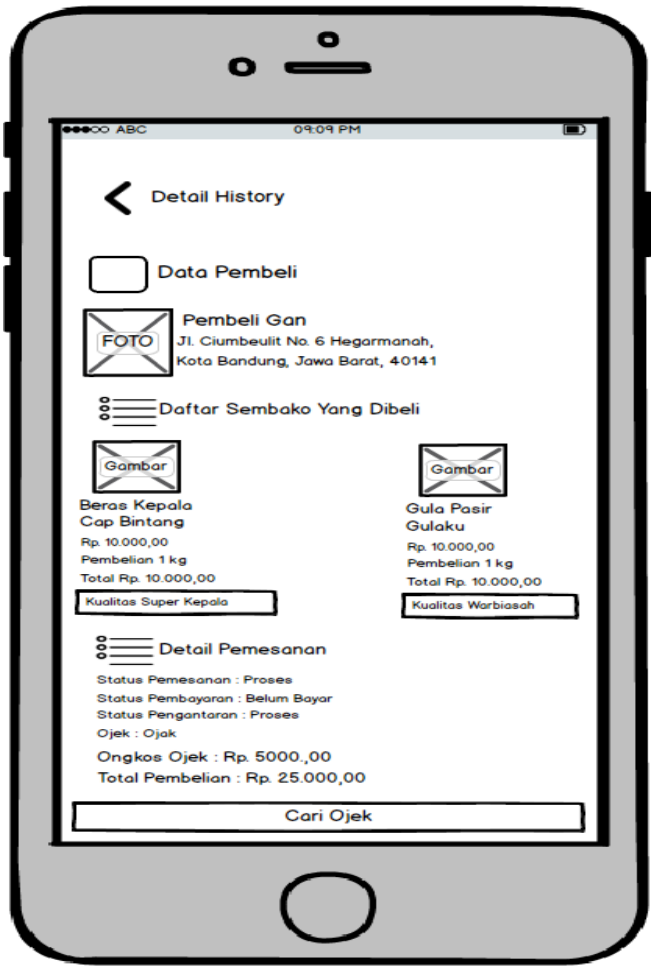
Berikut adalah Gambar 3.80 Antarmuka M11 Pemesanan yang menggambarkan rancangan antarmuka Pemesanan.

M11 Pemesanan	
	1. M11 menampilkan data Pemesanan 2. Klik teks box yang ada di sebelah kanan untuk masuk ke M12
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.80 Antarmuka M11 Pemesanan

l. Perancangan Antarmuka Detail History

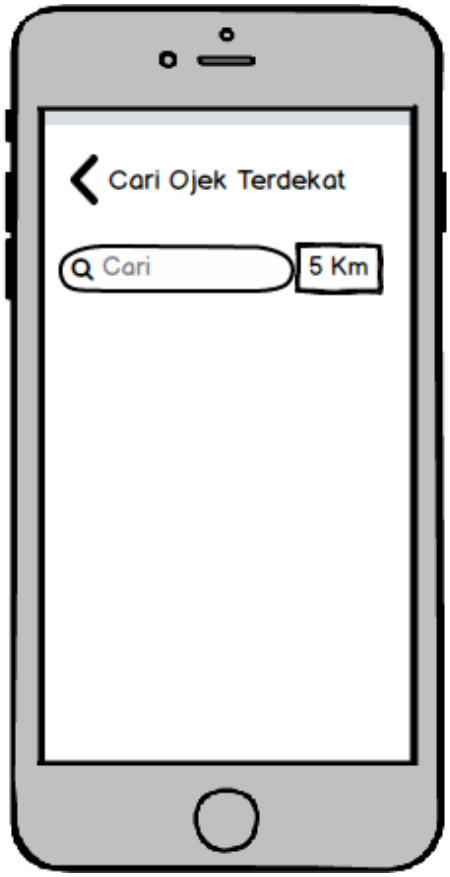
Berikut adalah Gambar 3.81 Antarmuka M12 Detail History yang menggambarkan rancangan antarmuka Detail History.

M12 Detail Hsitory	
	1. M12 menampilkan Detail History Pemesanan 2. Klik Cari Ojek untuk mencari ojek. 3. Kemudian akan masuk ke M13
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.81 Antarmuka M12 Detail History

m. Perancangan Antarmuka Cari Ojek

Berikut adalah Gambar 3.82 Antarmuka M13 Cari Ojek yang menggambarkan rancangan antarmuka Cari Ojek.

M13 Cari Ojek	
	1. M13 menampilkan menu cari ojek 2. Tentukan Radius terlebih dahulu 3. Kemudian Klik Cari untuk mencari ojek. 4. Bila tidak ditemukan ojek maka akan menampilkan pesan kesalahan PM04
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.82 Antarmuka M13 Cari Ojek

n. Perancangan Antarmuka Menu Ojek

Berikut adalah Gambar 3.83 Antarmuka M14 Menu Ojek yang menggambarkan rancangan antarmuka Menu Ojek.

M14 Menu Ojek	
	1. M14 menampilkan menu ojek yang telah di pesan
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.83 Antarmuka M14 Menu Ojek

o. Perancangan Antarmuka Detail History Ojek

Berikut adalah Gambar 3.84 Antarmuka M15 Detail History Ojek yang menggambarkan rancangan antarmuka Detail History Ojek.

M15 Detail History Ojek	
	1. M15 menampilkan menu ojek yang dipesan
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.84 Antarmuka M15 Detail History Ojek

p. Perancangan Antarmuka Menu Profil

Berikut adalah Gambar 3.85 Antarmuka M16 Menu Profil yang menggambarkan rancangan antarmuka Menu Profil.

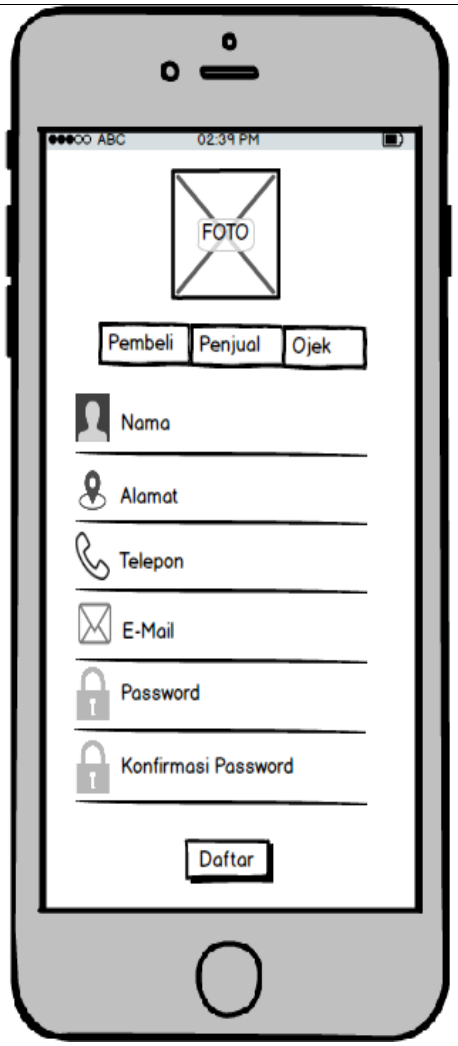
M16 Menu Profil	
	1. M16 akan menampilkan menu profil penjual
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.85 Antarmuka M16 Menu Profil

3. Perancangan Antarmuka Mobile Ojek

a. Perancangan Antar Muka Daftar

Berikut adalah Gambar 3.86 Antarmuka M01 Daftar yang menggambarkan rancangan antarmuka Daftar.

M01 Daftar	
	1. Data diri diisi terlebih dahulu 2. Tekan Daftar bila telah selesai mengisi data 3. bila data diisi dengan benar akan masuk ke M02 4. bila salah mengisi data akan menampilkan pesan PM01 dan kembali ke M01
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.86 Antarmuka M01 Daftar

b. Perancangan Antarmuka Login

Berikut adalah Gambar 3.87 Antarmuka M02 Login yang menggambarkan rancangan antarmuka Login.

M02 Login	
	1. Email dan Password di isi 2. Pilih salah satu radio button misalnya Anda login sebagai Ojek, Anda harus mengklik radio button ojek 3. Setelah selesai lalu tekan masukkemudian akan menampilkan M03 4. bila salah mengisi data akan menampilkan pesan PM02 dan kembali ke M02
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.87 Antarmuka M02 Login

c. Perancnagan Antarmuka Menu Profil

Berikut adalah Gambar 3.88 Antarmuka M03 Menu Profil yang menggambarkan rancangan antarmuka Menu Profil.

M03 Menu Profil	
	1. M03 menampilkan Menu Profil Ojek
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.88 Antarmuka M03 Menu Profil

d. Perancangan Antarmuka Menu Pemesanan

Berikut adalah Gambar 3.89 Antarmuka M04 Menu Pemesananyang menggambarkan rancangan antarmuka Menu Pemesanan.

M04 Menu Pemesanan	
	1. M03 menampilkan Menu Pemesanan.
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

Gambar 3.89 Antarmuka M04 Menu Pemesanan

e. Perancangan Antarmuka Detail History Pemesanan

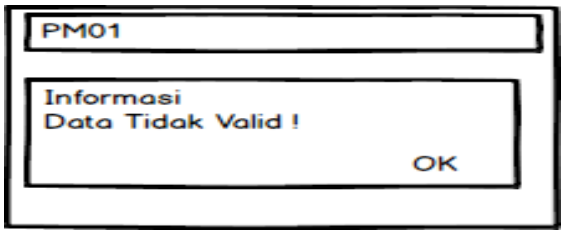
Berikut adalah Gambar 3.90 Antarmuka M05 Detail History Pemesanan yang menggambarkan rancangan antarmuka Detail History Pemesanan.

M05 Detail History Pemesanan	
	1. M03 menampilkan Detail History Pemesanan.
Ukuran : Disesuaikan Warna font : Disesuaikan Font Style : Disesuaikan Font Size : Disesuaikan	

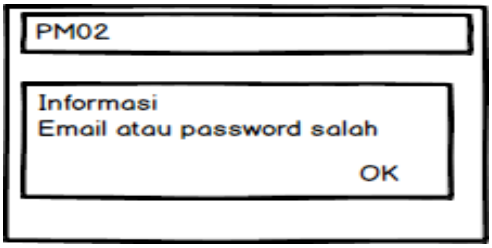
Gambar 3.90 Antarmuka M05 Detail History Pemesanan

3.1.6.5.1 Perancangan Pesan

Perancangan pesan pada pembangunan aplikasi pencarian toko sembako terdekat adalah sebagai berikut :



Gambar 3.91 Perancangan Pesan PM01



Gambar 3.92 Perancangan Pesan PM02



Gambar 3.93 Perancangan Pesan PM03



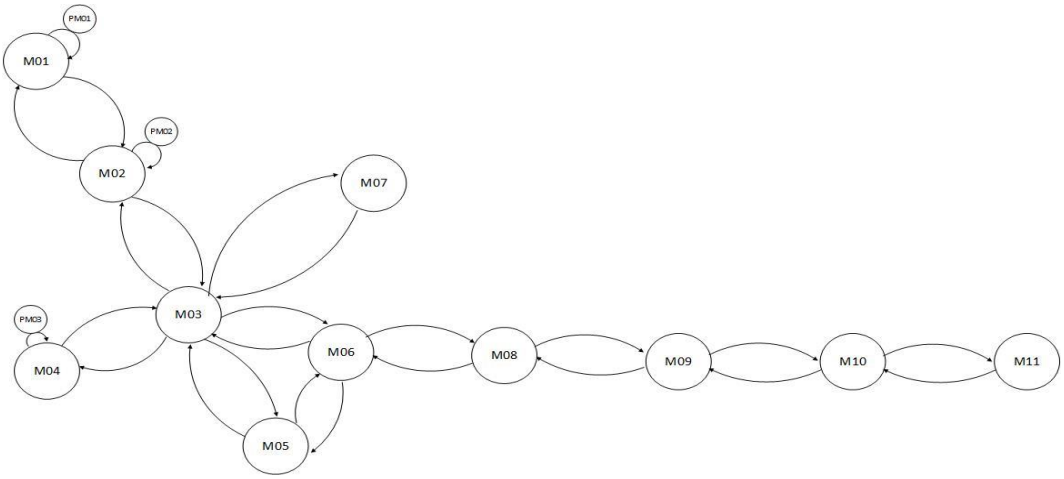
Gambar 3.94 Perancangan Pesan PM04

3.1.6.6 Jaringan Semantik

Jaringan Semantik pada pembangunan aplikasi pencarian toko sembako terdekat adalah sebagai berikut :

1. Jaringan Semantik *Mobile* Pembeli

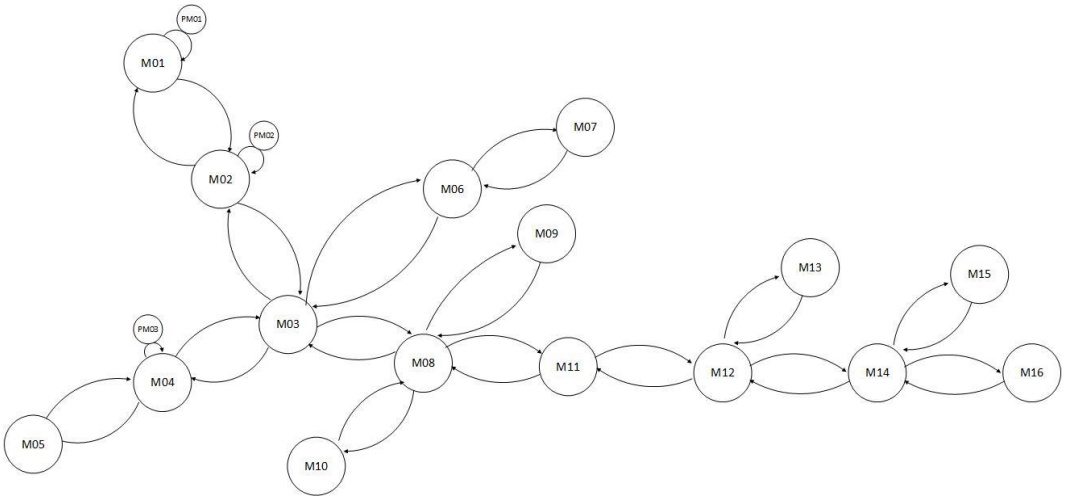
Berikut adalah jaringan semantik pada pembangunan perangkat lunak pada *platform mobile* pembeli.



Gambar 3.95 Jaringan Semantik *Mobile* Pembeli

2. Jaringan Semantik *Mobile* Penjual

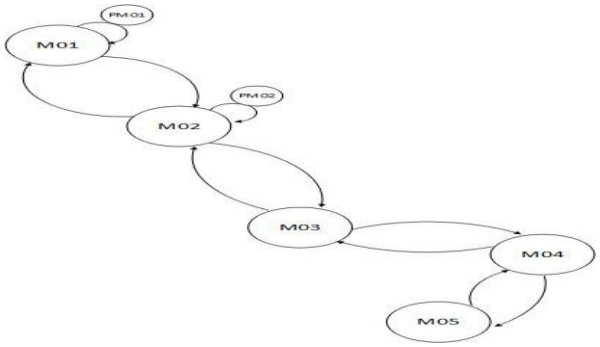
Berikut adalah jaringan semantik pada pembangunan perangkat lunak pada *platform mobile* penjual.



Gambar 3.96 Jaringan Semantik *Mobile* Penjual

3. Jaringan Semantik *Mobile* Ojek

Berikut adalah jaringan semantik pada pembangunan perangkat lunak pada *platform mobile* ojek.



Gambar 3.97 Jaringan Semantik *Mobile* Ojek