BAB III

ANALISIS DAN PERANCANGAN SISTEM

III. 1. Analisis Sistem

Secara global analisis sistem adalah mengidentifikasi apa saja yang diperlukan oleh sistem yang akan dibangun dan memahami dengan jelas proses yang akan berjalan pada sistem. Analisis sistem dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh kedalam bagian–bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi segala permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan – kebutuhan lain yang diharapkan, sehingga dapat dilakukan perbaikan pada tahap analisis sebuah sistem dan dapat mencapai tujuan dari pembangunan sistem. Tahap ini juga akan menentukan informasi yang diperlukan oleh para pengguna dari sistem yang baru, disamping juga persyaratan teknis dari sistem itu sendiri, sehingga tahap analisis sistem merupakan tahap yang kritis dan penting karena kesalahan ditahap ini akan menyebabkan kesalahan lain di tahap selanjutnya.

III. 2. Analisis Masalah

Berdasarkan hasil pengamatan pada Toko Arsefa Collection, ada beberapa permasalahan yang dihadapi didalam sistem yang sedang berjalan saat ini. Permasalahan – permasalahan tersebut diantaranya:

a) Proses penjualan yang berjalan masih dilakukan dengan cara konvensional seperti pembeli masih harus datang langsung ke toko jika ingin melakukan proses pembelian dan membayar secara tunai.
b) Belum adanya media promosi di Arsefa Collection sehingga info tentang produk dari Arsefa Collection menjadi sangat kurang.
c) Pengolahan data laporan penjualan masih manual dengan merekapitulasi nota transaksi penjualan setiap harinya dan mencatat laporan penjualan di buku.

III. 3. Analisis Prosedur yang sedang Berjalan

Prosedur adalah kumpulan dari proses dalam suatu sistem yang saling terkait antara satu dengan lainnya untuk mencapai tujuan yang telah diterapkan. Sistem yang saat ini sedang berjalan di Arsefa Collection memiliki dua prosedur, yaitu prosedur penjualan produk dan prosedur pembuatan laporan penjualan.

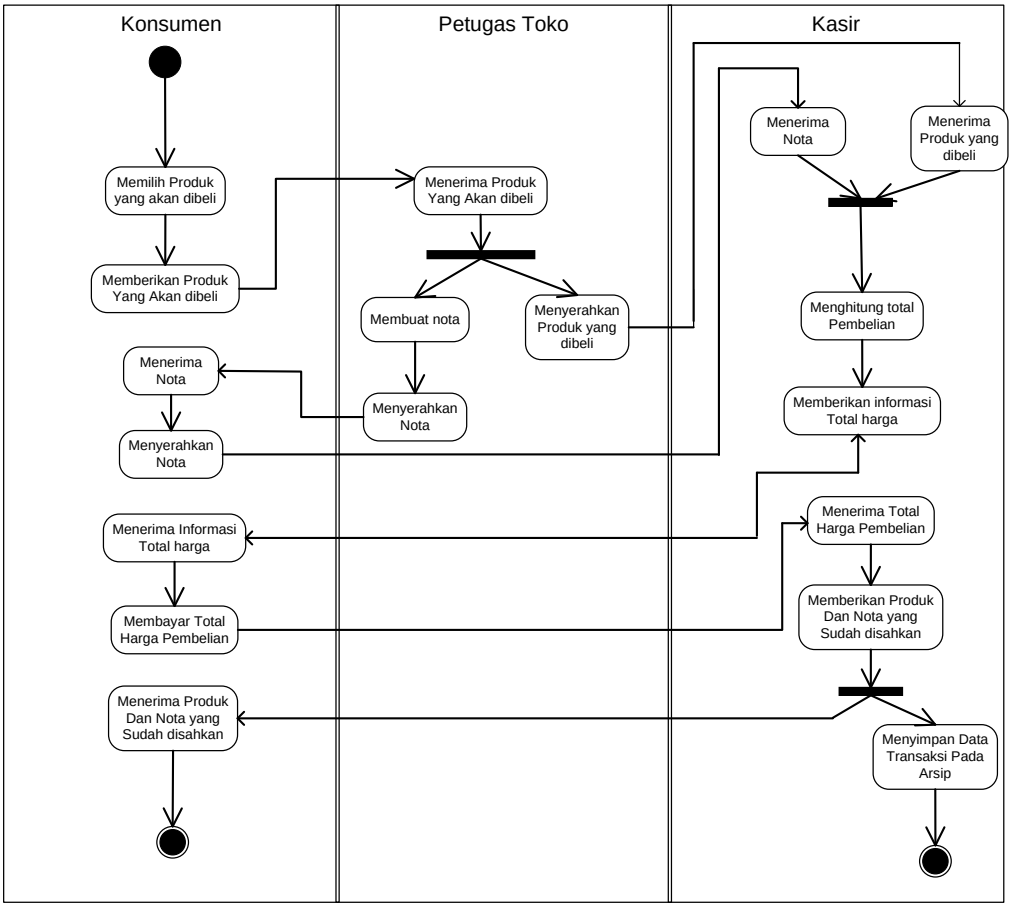
Berikut penjelasan dari prosedur - prosedur yang terlibat dalam sistem transaksi penjualan di Arsefa Collection.

III. 3. 1. Prosedur Penjualan Produk

Prosedur Penjualan adalah tahapan yang harus dilalui oleh konsumen dalam membeli produk dari mulai melihat produk, memilih produk, membayar sampai proses penerimaan produk oleh konsumen. Tahapan dari prosedur penjualan produk ini sebagai berikut:

1. Konsumen datang ke Arsefa Collection dan memilih produk yang akan dibeli yang kemudian menyerahkannya ke petugas toko.

2. Pegawai toko akan mencatat label produk ke nota transaksi, lalu menyerahkan nota transaksi ke konsumen dan menyerahkan produk yang dipesan ke bagian kasir.
3. Konsumen menyerahkan nota transaksi berikut sejumlah uang sesuai total bayar dari produk yang dibeli kepada kasir.
4. Kasir menerima nota transaksi dan uang pembayaran dari konsumen.
5. Kasir memberikan produk yang telah disiapkan kepada konsumen beserta nota transaksi yang sudah disahkan dan produk yang dibeli.
6. Konsumen menerima nota transaksi yang sudah disahkan dan produk yang dibeli.



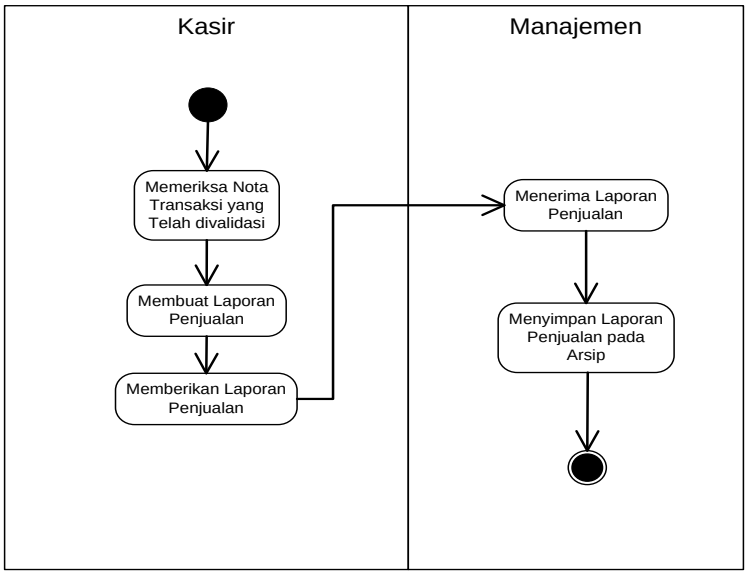
Gambar 3.1 *Activity Diagram* Penjualan Produk

III. 3. 2. Prosedur Laporan Penjualan

Prosedur laporan penjualan adalah proses pembuatan laporan yang didalamnya berisi nota transaksi yang akan ditandatangani oleh pimpinan. Adapun prosedur laporan penjualan ini meliputi produk apa saja yang telah dibeli, sehingga dari data pembelian produk dapat dijadikan evaluasi untuk mengetahui jenis produk apa yang paling disenangi oleh konsumen. Adapun tahap prosedur laporan sebagai berikut :

Prosedur laporan penjualan adalah proses pembuatan laporan oleh bagian kasir dari hasil rekapitulasi transaksi yang terjadi setiap harinya.

a) Kasir mengumpulkan nota transaksi dari hasil penjualan setiap harinya.
b) Kasir kemudian membuat laporan penjualan dari hasil rekapitulasi nota transaksi.
c) Kasir menyerahkan laporan penjualan ke pihak manajemen Arsefa Collection.



Gambar 3.2 *Activity Diagram* Laporan Penjualan

III. 3. 3. Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan tahapan untuk menganalisa kebutuhan-kebutuhan yang diperlukan dalam proses pengembangan sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Tahapan ini juga menentukan kebutuhan *input* atau masukan yang diperlukan, proses yang diperlukan dalam mengolah data masukan dan menentukan *output* atau keluaran sehinggga menghasilkan keluaran yang diharapkan. Analisis kebutuhan non fungsional ini meliputi analisis perangkat keras, analisis perangkat lunak dan analisis pengguna.

III.3.3.1. Analisis Perangkat Keras (*Hardware*)

Sistem yang akan dibangun akan diimplementasikan langsung dalam jaringan *Internet*. Adapun kebutuhan perangkat keras yang digunakan untuk membangun *e-commerce* ini minimal memiliki spesifikasi sebagai berikut:

a) Prosessor Intel Pentium IV atau setara dengan kecepatan 1.4 GHz
b) RAM 512 Mb
c) *Harddisk* dengan ruang kosong 5 Gb.
d) K*eyboard* dan *Mouse.*
e) Monitor dengan resolusi 1024 x 768 pixels.
f) Terkoneksi dengan *internet.*

Sedangkan untuk pengguna, spesifikasi minimum perangkat keras yang dibutuhkan, yaitu:

a) Prosessor Intel Pentium III atau setara dengan 800 MHz.
b) Memori 512 MB.

c) *Harddisk* dengan ruang kosong 500 MB.

d) K*eyboard* dan *Mouse.*

e) Monitor dengan resolusi 1024 x 768 pixels .

f) Terkoneksi dengan *internet*.

III.3.3.2. Analisis Perangkat Lunak (*Software*)

Perangkat lunak yang dipergunakan untuk menggunakan aplikasi yang akan dibangun ini minimal membutuhkan spesifikasi sebagai berikut:

a) Sistem operasi Windows XP.

b) Macromedia Dreamweaver 8, Adobe Photoshop CS2 dan Macromedia Flash 8 sebagai *tools* pembangun aplikasi.

c) WampServer 2.0 sebagai *web server* saat pembangunan aplikasi.

d) DBMS menggunakan MySQL.

e) Bahasa pemrograman yang digunakan adalah PHP dan untuk optimalisasi tampilannya menggunakan CSS.

f) Mozila Firefox, Google Chrome, Internet explorer sebagai *browser.*

III.3.3.3. Analisis Pengguna (*User*)

Analisis pengguna dilakukan untuk mengetahui tugas dan karakteristik pengguna yang akan menggunakan sistem yang akan dibangun. Dengan mempertimbangkan tingkat pengalaman pengguna dalam menggunakan komputer, dapat dirancang sebuah sistem yang mudah dipelajari dan digunakan.

Terdapat tiga kategori pengguna dari sistem yang akan dibangun, yaitu admin, pembeli sebagai *member* dan pengunjung/*guest, adalah* sebagai berikut:

1. Admin

Admin adalah pengelola sistem yang memiliki tugas dan akses khusus kedalam sistem seluruh data, dengan karakteristik sebagai berikut:

a. Memiliki kemampuan dasar dalam menggunakan perangkat komputer.

b. Dapat menjalankan sistem operasi yang digunakan yaitu *Windows XP*.

c. Memiliki pengalaman di bidang komputer dan administrasi.

2. Member

Member adalah pengguna sistem yang telah terdaftar sebagai anggota dari sistem yang dibangun sehingga dapat melakukan transaksi pembelian dan melihat semua product yang dijual, dengan karakteristik sebagai berikut.

a. Memiliki kemampuan dasar dibidang komputer.

b. Pernah membuka halaman website.

c. Memahami prosedur pendaftaran dalam situs - situs transaksi *on-line*

d. Dapat mengoperasikan sistem operasi yang digunakan yaitu *Windows* XP.

3. Pengunjung (*guest*)

Pengunjung adalah pengguna sistem yang tidak terdaftar dan hanya diberi hak untuk melihat informasi product yang ada dan tidak dapat melakukan pembelian produk lewat sistem yang ada. Karakteristik pengunjung adalah sebagai berikut:

a. Memiliki kemampuan dasar di bidang komputer.

b. Pernah membuka halaman website.

c. Dapat mengoperasikan sistem operasi yang digunakan yaitu *Windows* XP.

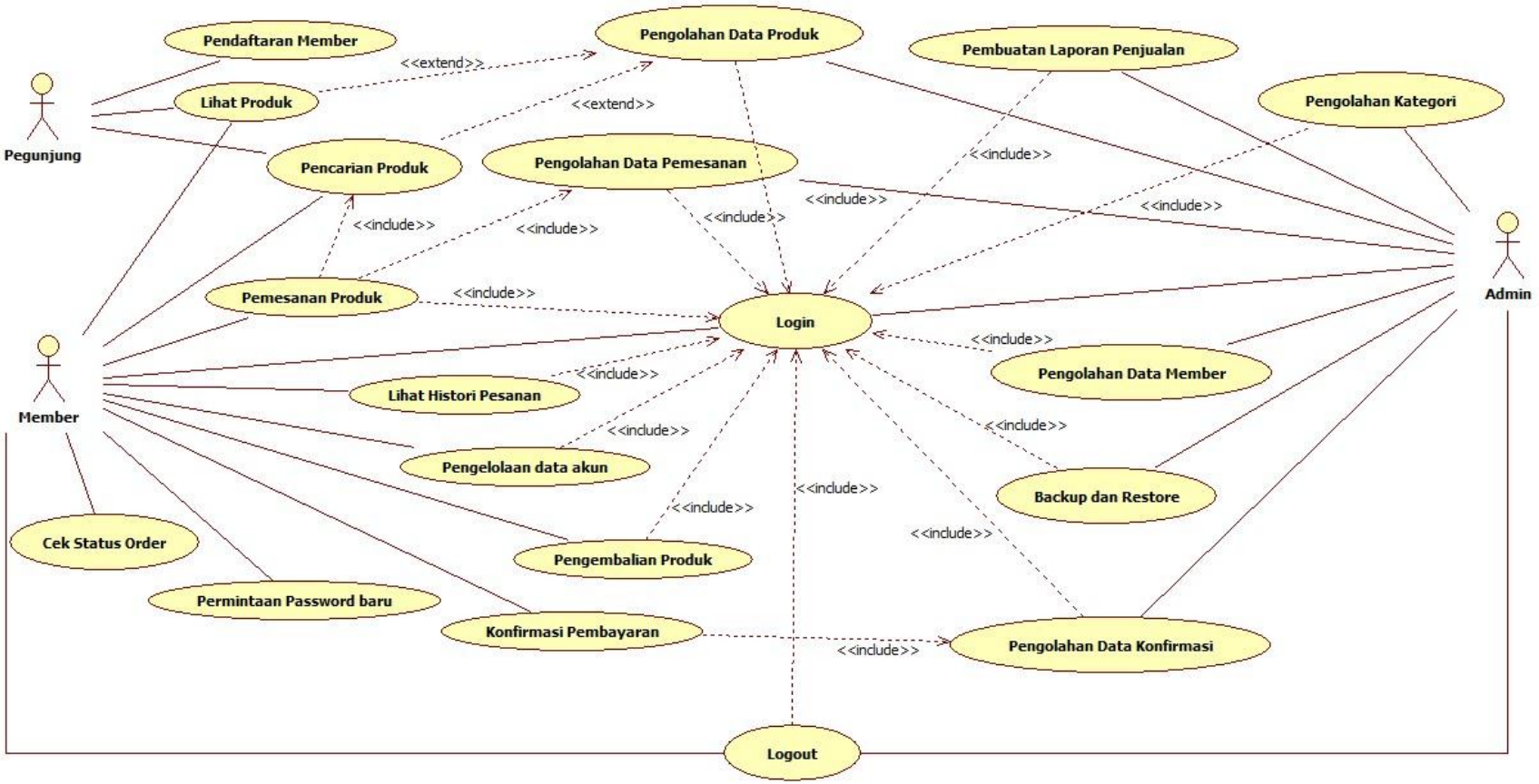
III.3.3.4. Identifikasi Aktor

Aplikasi *e-commerce* yang dibangun akan digunakan oleh tiga orang aktor yaitu admin, member dan pengunjung. Deskripsi dari masing-masing aktor adalah sebagai berikut :

a) Aktor pertama adalah *admin* sebagai pengelola yang memiliki semua hak akses yang ada pada sistem termasuk untuk menambah, mengubah dan menghapus *member*.
b) Aktor kedua adalah *Member* sebagai pengunjung yang memiliki hak untuk melakukan transaksi pembelian.
c) Aktor ketiga adalah Pengunjung yang memiliki hak akses hanya melihat dan melakukan pencarian produk.

III.3.3.5. *Use case Diagram*

Use case adalah interaksi atau dialog antara sistem dan aktor, termasuk pertukaran pesan dan tindakan yang dilakukan oleh sistem. *Use case* diagram menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar.



Gambar 3.3 *use case diagram* untuk *E-commerce* pada Toko Arsefa

III.3.3.6. *Use case Scenario*

Use Case Scenario menjelaskan interaksi antara aktor dan *use case* dalam melakukan sebuah proses tertentu. Berikut adalah *use case scenario* berdasarkan *use case* pada *E-commerce* Arsefa.

1. **Skenario *Use Case Login***

   Interaksi antara aktor admin dan *membe*r dengan *use case login* dapat dilihat pada tabel 3.1.

Tabel 3.1 Skenario *Use Case Login*

Identifikasi			
Nama	*Login*		
Tujuan	Masuk ke dalam sistem		
Deskripsi	Proses *Login* merupakan proses autentikasi untuk menggunakan hak akses sebagai *administrator* atau *member* menggunakan sistem.		
Aktor	*Admin, Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Sistem menampilkan form login		
Aksi		**Reaksi Sistem**	
1	Mengisi form login	2	Mengecek data pada tabel di *database.*
		3	Bila data ada dan cocok maka akan masuk ke dalam sistem dan menampilkan menu utama sesuai hak akses.
Skenario Alternatif (Autentifikasi Gagal)			
Aksi		**Reaksi Sistem**	
		4	Menampilkan pesan data login salah.
5	Mengisi kembali form login	6	Mengecek data *Member* di *database*.
		7	Bila akun ada dan cocok maka akan masuk ke dalam sistem dan menampilkan menu utama sesuai hak akses.
Kondisi Akhir	Berhasil masuk ke dalam sistem dan menampilkan menu utama		

2. Skenario *Use Case* Permintaan Password Baru

Interaksi antara aktor admin, dan *member* dengan *use case* Permintaan Password Baru dapat dilihat pada tabel 3.2.

Tabel 3.2 Skenario *Use Case* Permintaan Password Baru

Identifikasi			
Nama	Permintaan Password Baru		
Tujuan	Mereset password aktor		
Deskripsi	Proses untuk mereset password, memberitahu admin dan memberikan password baru melalui email yang didaftarkan.		
Aktor	*Admin, Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Memilih menu lupa password		
Aksi		**Reaksi Sistem**	
1	Mengisi form username atau email yang sebelumnya didaftarkan	2	Mencari data berdasarkan username atau email yang diisikan, kemudian mereset password dan mengirimkannya ke email yang ada pada *database*.
Kondisi Akhir	Password telah direset dan dikirm ke email.		

3. Skenario *Use Case Logout*

Interaksi antara aktor admin, dan member dengan *use case logout* dapat dilihat pada tabel 3.3.

Tabel 3.3 Skenario *Use Case Logout*

Identifikasi			
Nama	Logout		
Tujuan	Keluar dari sistem		
Deskripsi	Merupakan proses untuk berhenti menggunakan dan berhenti dari sesi *login*		
Aktor	*Admin, Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu utama sesuai hak akses		
Aksi		**Reaksi Sistem**	
1	Memilih menu keluar	2	Menghentikan akses dan mengarahkan pada form *login*
Kondisi Akhir	berhasil *logout* dari sistem		

4. Skenario *Use Case* Pendaftaran Member

Interaksi antara aktor pengunjung dengan *use case* pendaftaran member dapat dilihat pada tabel 3.4.

Tabel 3.4 Skenario *Use Case* Pendaftaran Member

Identifikasi			
Nama	Pendaftaran *Member*		
Tujuan	Melakukan pendaftaran *member*		
Deskripsi	Proses untuk mendaftar/registrasi menjadi *member*		
Aktor	Pengunjung		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan halaman awal (*home*)		
Aksi		**Reaksi Sistem**	
1	Memilih menu daftar/akun	2	Menampilkan halaman registrasi dan form pendaftaran *member*
3	Mengisi form pendaftaran member	4	Memvalidasi data masukan, jika data tidak valid maka tampil pesan kesalahan dan menampilkan kembali form pendaftaran, jika data valid maka data akan disimpan dan mengirimkan e-mail konfirmasi registrasi berhasil.
Kondisi Akhir	Pengunjung berhasil mendaftar sebagai *member*		

5. Skenario *Use Case* Lihat Produk

Interaksi antara aktor pengunjung dan *member* dengan *use case* lihat produk dapat dilihat pada tabel 3.5.

Tabel 3.5 Skenario *Use Case* Lihat Produk

Identifikasi			
Nama	Lihat Produk		
Tujuan	Melihat info produk		
Deskripsi	Merupakan proses untuk melihat list dan info produk yang tersedia		
Aktor	Pengunjung, *Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan halaman awal (*home*)		
Aksi		**Reaksi Sistem**	
1	Memilih tampil produk	2	Mengambil data produk dari *database* menampilkannya pada halaman produk
Kondisi Akhir	Halaman Produk berhasil ditampilkan		

6. Skenario *Use Case* Pencarian Produk

Interaksi antara aktor pengunjung dan member dengan *use case* pencarian produk dapat dilihat pada tabel 3.6.

Tabel 3.6 Skenario *Use Case* Pencarian Produk

Identifikasi			
Nama	Pencarian Produk		
Tujuan	Mencari dan Melihat info produk		
Deskripsi	Proses untuk mencari data produk dan menampilkannya		
Aktor	Pengunjung, *Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan halaman awal (*home*)		
Aksi Aktor		**Reaksi Sistem**	
1	Mengisi data cari produk Berdasarkan kode/nama/kategori produk.	2	Mencari data produk pada *database*, jika data ditemukan maka akan menampilkan list data produk, jika data tidak ditemukan maka akan menampilkan pesan data tidak ditemukan
Kondisi Akhir	Menampilkan hasil pencarian produk		

7. Skenario *Use Case* Pemesanan Produk

Interaksi antara aktor member dengan *use case* Pemesanan produk dapat dilihat pada tabel 3.7.

Tabel 3.7 Skenario *Use Case* Pemesanan Produk

Identifikasi			
Nama	Pemesanan produk		
Tujuan	Melakukan pemesanan produk		
Deskripsi	Proses untuk memilih produk yang akan dipesan		
Aktor	Member		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan halaman awal (beranda)		
Aksi Aktor		**Reaksi**	
1	Memilih kategori dan produk yang akan dipesan	2	Mengecek apakah member sudah login, jika sudah login maka akan menampilkan form order, jika belum login maka akan menampilkan pesan harus login dan menampilkan form login
3	Menekan tombol tambah ke keranjang	4	Menyimpan data *order* pada keranjang
5	Menekan tombol *checkout*	6	Mengambil alamat dari data member dan menampilkan sebagai alamat pengiriman
7	Memilih tipe jasa pengiriman	8	Menyimpan data ongkos kirim sesuai tipe jasa pengiriman yang dipilih
9	Menekan tombol ‘Lanjutkan’	10	Menampilkan alamat dari member sebagai alamat pengiriman dan menampilkan pilihan cara pembayaran.
11	Menekan tombol ‘Lanjutkan’	12	Menampilkan rangkuman *order* yang telah dipesan, ongkos kirim sesuai alamat pengiriman, total yang harus dibayar dan alamat pengiriman.
13	Menekan tombol ‘Konfirmasi Pemesanan`	14	Menyimpan data order pada *database*
15		16	Mengirimkan e-mail konfirmasi *order* kepada *member*
Kondisi Akhir	Pemesan berhasil melakukan *order* produk		

8. Skenario *Use Case* Lihat *Histori* Pemesanan

Interaksi antara aktor *member* dengan *use case* lihat *history* pemesanan dapat dilihat pada tabel 3.8

Tabel 3.8 Skenario *Use Case* Lihat *History* Pemesanan

Identifikasi			
Nama	Lihat *History* Pemesanan		
Tujuan	Melihat pemesanan yang pernah dilakukan member		
Deskripsi	Proses untuk melihat daftar pemesanan yang sudah dilakukan oleh member		
Aktor	*Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu utama *member*		
Aksi		**Reaksi Sistem**	
1	Memilih menu history pemesanan	2	Mencari data pesanan pada *database* berdasarkan id member. Jika data ditemukan maka data akan ditampilkan, jika tidak akan tampil pesan data *history* kosong
Kondisi Akhir	Member dapat melihat *history* pemesanan		

9. Skenario *Use Case* Pengelolaan Data Akun

Interaksi antara aktor member dengan *use case* pengolahan akun dapat dilihat pada tabel 3.9.

Tabel 3.9 Skenario *Use Case* Pengelolaan Data Akun

Identifikasi			
Nama	Pengolahan Akun		
Tujuan	Mengolah data akun *member*		
Deskripsi	Proses untuk mengolah data akun *member*		
Aktor	*Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu utama member		
Aksi Aktor		**Reaksi Sistem**	
1	Memilih menu data akun	2	Mencari data akun berdasarkan id member dan menampilkannya.
3	Memilih menu “edit”	4	Menampilkan form ubah data akun
5	Mengisi data akun yang baru	6	Menyimpan data masukan ke *database*
Kondisi Akhir	Member berhasil mengolah data akun		

10. Skenario *Use Case* Konfirmasi pembayaran

Interaksi antara aktor member dengan *use case* konfirmasi pembayaran dapat dilihat pada tabel 3.10.

Tabel 3.10 Skenario *Use Case* Konfirmasi pembayaran

Identifikasi			
Nama	Konfirmasi pembayaran		
Tujuan	Melakukan konfirmasi pembayaran untuk sebuah *order*		
Deskripsi	Proses konfirmasi pembayaran *order* untuk pembayaran yang dilakukan melalui transfer bank		
Aktor	*Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu awal (beranda)		
Aksi		**Reaksi Sistem**	
1	Mengisi form konfirmasi pembayaran meliputi order ID, jumlah bayar, bank tujuan, bank asal dan nama pembayar.	2	Menyimpan data masukan ke dalam *database.*
		3	Mengirim e-mail konfirmasi pembayaran berhasil
Kondisi Akhir	Member berhasil melakukan konfirmasi pembayaran		

11. Skenario *Use Case* Cek Status *Order*

Interaksi antara aktor member dengan *use case* Cek Status *Order* dapat dilihat pada tabel 3.11

Tabel 3.11 Skenario *Use Case* Cek Status *Order*

Identifikasi			
Nama	Cek Status *Order*		
Tujuan	Melihat status order tertentu		
Deskripsi	Proses mengecek data order pada *database* dan menampilkannya kepada *member*		
Aktor	*Member*		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu awal (beranda)		
Aksi		**Reaksi Sistem**	
1	Mengisi form cek status order dengan order ID	2	Mencari data order pada *database* berdasarkan order ID, Jika data ditemukan maka akan ditampilkan.
Skenario Alternatif (Data order tidak ditemukan)			
		3	Menampilkan pesan data order tidak ditemukan
4	Mengisi kembali form cek status order	5	Mencari data order pada *database* berdasarkan order ID, Jika data ditemukan maka akan ditampilkan.
Kondisi Akhir	Menampilkan status data *order*		

12. Skenario *Use Case* Pengolahan Data *Member*

Interaksi antara aktor admin dengan *use case* pengolahan data *member* dapat dilihat pada tabel 3.12

Tabel 3.12 Skenario *Use Case* Pengolahan Data *Member*

Identifikasi			
Nama	Pengolahan Data *Member*		
Tujuan	Mengelola Data *Member* pada sistem		
Deskripsi	Proses pengolahan data member merupakan proses untuk mengatur dan mengolah data *member* pada sistem yang mencakup tampil data, tambah data, ubah data, hapus data dan cari data *member*.		
Aktor	Admin		
***Use Case* yang Berkaitan**	-		
Skenario Utama			
Kondisi Awal	Menampilkan data user dan menu pengolahan data user		
Aksi Aktor		**Reaksi Sistem**	
1	Memilih menu Member	2	Menampilkan Halaman *member*
3	Memilih menu masukan	4	Menampilkan form tambah member
5	Mengisi form tambah data user dan menekan tombol simpan	6	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman member, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form tambah data user
Skenario Alternatif (Ubah Data user)			
7	Memilih menu edit data pada data user yang diinginkan	8	Menampilkan form ubah data yang berisi data dari *database* yang akan diubah.
9	Mengubah data yang ada pada form ubah data dan menekan tombol simpan	10	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman member, jika tidak valid maka akan Menampilkan pesan kesalahan dan menampilkan kembali form ubah data user
Skenario Alternatif (Hapus Data user)			
11	Memilih data user yang akan dihapus dan menekan tombol Hapus	12	Menampilkan pesan konfirmasi apakah data akan dihapus
13	Menekan tombol ya pada pesan konfirmasi	14	Menghapus data user pada *database*

Tabel 3.12 Skenario *Use Case* Pengolahan Data *Member (lanjutan)*			
15	Menekan tombol tidak pada pesan konfirmasi	16	Membatalkan proses penghapusan dan mengarahkan kembali ke halaman pelanggan
Kondisi Akhir	Data member berhasil diolah		

13. Skenario *Use Case* Pengolahan Kategori

Interaksi antara aktor admin dan operator dengan *use case* pengolahan data kategori dapat dilihat pada tabel 3.13

Tabel 3.13 Skenario *Use Case* Pengolahan Data Kategori

Identifikasi			
Nama	Pengolahan Data kategori		
Tujuan	Mengelola data kategori pada sistem		
Deskripsi	Proses pengolahan data kategori merupakan proses untuk mengatur dan mengolah data kategori pada sistem yang mencakup tampil data, tambah data, ubah data, hapus data		
Aktor	Admin		
***Use Case* yang Berkaitan**	-		
Skenario			
Kondisi Awal	Menampilkan data kategori dan menu pengolahan kategori		
Aksi		**Reaksi Sistem**	
1	Memilih menu kategori	2	Menampilkan Halaman kategori
3	Mengisi form tambah data kategori dan menekan tombol simpan	4	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman kategori, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form tambah data kategori
Skenario Alternatif (Ubah Data kategory)			
5	Memilih menu ubah data pada data	6	Menampilkan form ubah data yang berisi data dari *database* yang akan diubah.
7	Mengubah data yang ada pada form ubah data dan menekan tombol simpan	8	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman kategori, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form ubah data kategori
Skenario Alternatif (Hapus Data kategory)			

Tabel 3.13 Skenario *Use Case* Pengolahan Data Kategori(lanjutan)			
9	Memilih data kategori yang akan dihapus	10	Menampilkan pesan konfirmasi apakah data akan dihapus
11	Menekan tombol ya pada pesan konfirmasi	12	Menghapus data kategori pada *database*
13	Menekan tombol tidak pada pesan konfirmasi	14	Membatalkan proses penghapusan dan mengarahkan kembali ke halaman kategori
Skenario Alternatif (Cari Data kategori)			
15	Memasukan data cari dan memilih tombol cari	16	Mencari data pada *database*, jika data Ditemukan maka data akan ditampilkan, jika tidak ditemukan maka akan menampilkan pesan data tidak ditemukan.
Kondisi Akhir	Data kategori telah berhasil diolah		

14. *Use case* Pembuatan Laporan Penjualan

Interaksi antara aktor admin dan operator dengan *use case* pembuatan laporan penjualan dapat dilihat pada tabel 3.14

Tabel 3.14 Skenario *Use Case* Pembuatan Laporan Penjualan

Identifikasi			
Nama	Pembuatan laporan		
Tujuan	Membuat laporan dari data yang telah diolah		
Deskripsi	Merupakan proses pembuatan laporan dari data yang telah diolah sebelumnya mencakup produk, order, dan pembayaran.		
Aktor	Admin		
Use Case yang Berkaitan	-		
Skenario			
Kondisi Awal	Menampilkan menu pilihan laporan		
Aksi Aktor		**Reaksi**	
1	Memilih data yang akan dijadikan laporan	2	Mengecek data pada *database*, jika data ditemukan maka akan menampilkan data, jika tidak ditemukan maka akan menampilkan pesan data tidak ditemukan
3	Memilih jangka waktu yang akan dibuat laporan dan menekan tombol cetak laporan	4	Membuat laporan berdasarkan jangka waktu yang dipilih
Kondisi Akhir	laporan berhasil dibuat		

15. Skenario Use Case Backup dan Restore

Interaksi antara aktor admin dan operator dengan *use case Backup* dan *Restore Database* dapat dilihat pada tabel 3.15

Tabel 3.15 Skenario *Use Case Backup* dan *Restore Database*

Identifikasi	
Nama	Backup dan Restore *Database*
Tujuan	Melakukan proses backup dan restore pada *database*
Deskripsi	Proses mengambil semua data pada *database* (backup) dan mengembalikan data pada *database* (restore)
Aktor	Admin
Use Case yang Berkaitan	-

Skenario Utama			
Kondisi Awal	Menampilkan menu backup dan restore *database*		
Aksi		**Reaksi Sistem**	
1	Menekan tombol Backup dan Restore *database*	2	Menampilkan Halaman backup dan restore *database*
3	Memilih data data yang akan di backup dengan cara menceklist pada pilihan	4	Menampilkan data data yang telah dicek untuk di backup
5	Menekan tombol cadangkan untuk mulai membackup	6	Menyimpan file SQL ke tempat penyimpanan file atau *hard drive*
Skenario Alternatif (Restore *Database*)			
		7	Menampilkan form restore *database*
8	Memilih file berformat SQL yang berisi data dari *database*		
9	Menekan tombol kembalikan	10	Membaca data dari file SQL
		11	Menyalin data dari file SQL ke *database*
Kondisi Akhir	Data telah di backup atau di restore		

16. Skenario *Use Case* Pengolahan Data Konfirmasi

Interaksi antara aktor member dengan *use case* Konfirmasi pembayaran dapat dilihat pada tabel 3.16

Tabel 3.16 Skenario *Use Case* Pengolahan Data Konfirmasi

Identifikas			
Nama	Konfirmasi Pembayaran		
Tujuan	Melakukan konfirmasi pembayaran untuk sebuah order		
Deskripsi	Proses konfirmasi pembayaran order untuk pembayaran yang dilakukan melalui transfer bank		
Aktor	Admin		
Use Case yang Berkaitan	-		
Skenario Utama			
Kondisi Awal	Menampilkan Halaman Pengolahan Konfirmasi pembayaran		
Aksi		**Reaksi Sistem**	
1	Mengisi form konfirmasi pembayaran meliputi jumlah bayar, bank tujuan, bank asal dan nama pembayar.	2	Melakukan validasi data masukan, jika data valid maka data akan disimpan pada *database*.
		3	Mengirim e-mail konfirmasi pembayaran
Skenario Alternatif (Data order tidak ditemukan)			
		4	Menampilkan pesan data order tidak ditemukan
5	Mengisi kembali form Konfirmasi pembayaran	6	Mencari data order pada *database* berdasarkan order ID, Jika data ditemukan maka akan ditampilkan.
Kondisi Akhir	Admin berhasil melakukan pengolahan data konfirmasi pembayaran		

17. Skenario *Use Case* Pengolahan Data Produk

Interaksi antara aktor admin dan operator dengan *use case* pengolahan data produk dapat dilihat pada tabel 3.17.

Tabel 3.17 Skenario *Use Case* Pengolahan Data Produk

Identifikasi			
Nama	Pengolahan Data Produk		
Tujuan	Mengelola data produk pada sistem		
Deskripsi	Proses pengolahan data produk merupakan proses untuk mengatur dan mengolah data produk pada sistem yang mencakup tampil data, tambah data, ubah data, hapus data dan cari data.		
Aktor	Admin		
Use Case yang Berkaitan	-		
Skenario Utama			
Kondisi Awal	Menampilkan data produk dan menu pengolahan produk		
Aksi Aktor		**Reaksi Sistem**	
1	Memilih menu tambah data produk	2	Menampilkan form tambah data produk
3	Mengisi form tambah data produk dan menekan tombol simpan	4	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman produk, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form tambah data produk
Skenario Alternatif (Ubah Data Produk)			
5	Memilih menu ubah data pada data produk yang diinginkan	6	Menampilkan form ubah data yang berisi data dari *database* yang akan diubah.
7	Mengubah data yang ada pada form ubah data dan menekan tombol simpan	8	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman produk, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form ubah data produk
Skenario Alternatif (Hapus Data Produk)			
9	Memilih data produk yang akan dihapus	10	Menampilkan pesan konfirmasi apakah data akan dihapus
11	Menekan tombol ya pada pesan konfirmasi	12	Menghapus data produk pada *database*
13	Menekan tombol tidak pada pesan konfirmasi	14	Membatalkan proses penghapusan dan mengarahkan kembali ke halaman produk

Tabel 3.17 Skenario *Use Case* Pengolahan Data Produk(Lanjutan)			
15	Memasukan data cari dan memilih tombol cari	16	Mencari data pada *database*, jika data ditemukan maka data akan ditampilkan, jika tidak ditemukan maka akan menampilkan pesan data tidak ditemukan.
Kondisi Akhir	Data produk telah berhasil diolah		

18. Skenario *Use Case* Pengolahan Data Pemesanan

Interaksi antara aktor admin dengan *use case* pengolahan data *Pemesanan* dapat dilihat pada tabel 3.18

Tabel 3.18 *Use Case* Skenario Pengolahan Data Pemesanan

Identifikas			
Nama	Pengolahan Data Pemesanan		
Tujuan	Mengelola data Pemesanan pada sistem		
Deskripsi	Proses pengolahan data order merupakan proses untuk mengatur dan mengolah data order pada sistem yang mencakup tampil data, tambah data, ubah data, hapus data dan cari data order.		
Aktor	Admin		
Use Case yang Berkaitan	-		
Skenario Utama			
Kondisi Awal	Menampilkan data order dan menu pengolahan data order		
Aksi		**Reaksi Sistem**	
1	Memasukan data cari dan memilih tombol cari	2	Mencari data pada *database*, jika data ditemukan maka data akan ditampilkan, jika tidak ditemukan maka akan menampilkan pesan data tidak ditemukan.
Skenario Alternatif (Ubah Data Order)			
3	Memilih menu ubah data pada data	4	Menampilkan form ubah data yang berisi data dari *database* yang akan diubah.
5	Mengubah data yang ada pada form ubah data dan menekan tombol simpan	6	Memvalidasi data masukan, jika data valid maka data disimpan pada *database* dan mengarahkan ke halaman order, jika tidak valid maka akan menampilkan pesan kesalahan dan menampilkan kembali form ubah data order
Skenario Alternatif (Hapus Data Order)			
7	Memilih data order yang akan dihapus	8	Menampilkan pesan konfirmasi apakah data akan dihapus

Tabel 3.18 *Use Case* Skenario Pengolahan Data Pemesanan(Lanjutan)			
9	Menekan tombol ya pada pesan konfirmasi	10	Menghapus data order pada *database*
11	Menekan tombol tidak pada pesan konfirmasi	12	Membatalkan proses penghapusan dan mengarahkan kembali ke halaman order
Kondisi Akhir		Data Pemesanan telah berhasil diolah	

19. Skenario *Use Case* Pengembalian Produk

Interaksi antara aktor member dengan *use case* pengembalian produk dapat dilihat pada tabel 3.19

Tabel 3.19 Skenario *use case* Pengembalian Produk

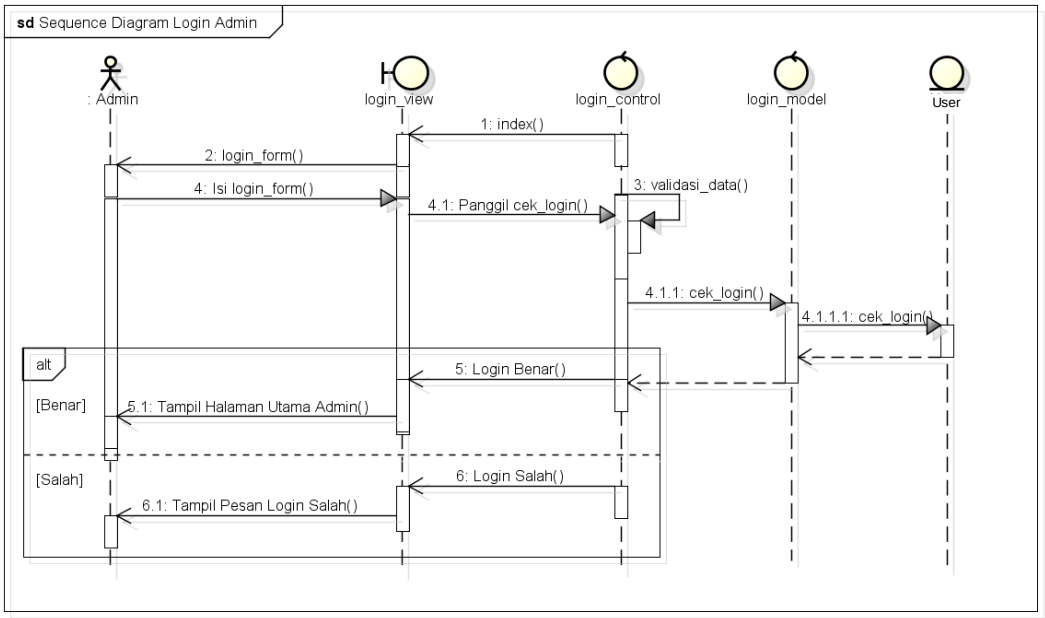
Identifikas			
Nama	Pengembalian Produk		
Tujuan	Melakukan proses pengembalian produk		
Deskripsi	Merupakan proses pengembalian produk yang tidak sesuai dengan proses pembelian		
Aktor	*Member*		
Use Case yang Berkaitan	-		
Skenario Utama			
Kondisi Awal	Menampilkan menu Pengisian data order produk		
Aksi		**Reaksi Sistem**	
1	Memasukan nomor faktur dan nomor resi pengiriman	2	Mencari data pada *database*, jika data ditemukan maka data akan ditampilkan, jika tidak ditemukan maka akan menampilkan pesan data tidak ditemukan.
Kondisi Akhir		Data pengembalian produk telah berhasil dibuat	

III.3.3.7. *Sequence* Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem berupa pesan (*message*) yang digambarkan terhadap waktu.

1. ***Sequence Diagram* Login Admin**

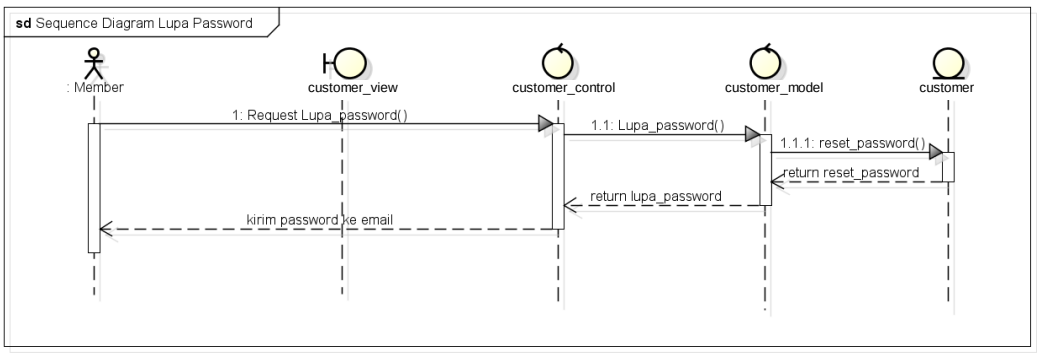
Sequence Diagram Login Admin dapat dilihat pada gambar 3.4



Gambar 3.4 *Sequence Diagram* Login Admin

2. ***Sequence Diagram* Permintaan Password Baru**

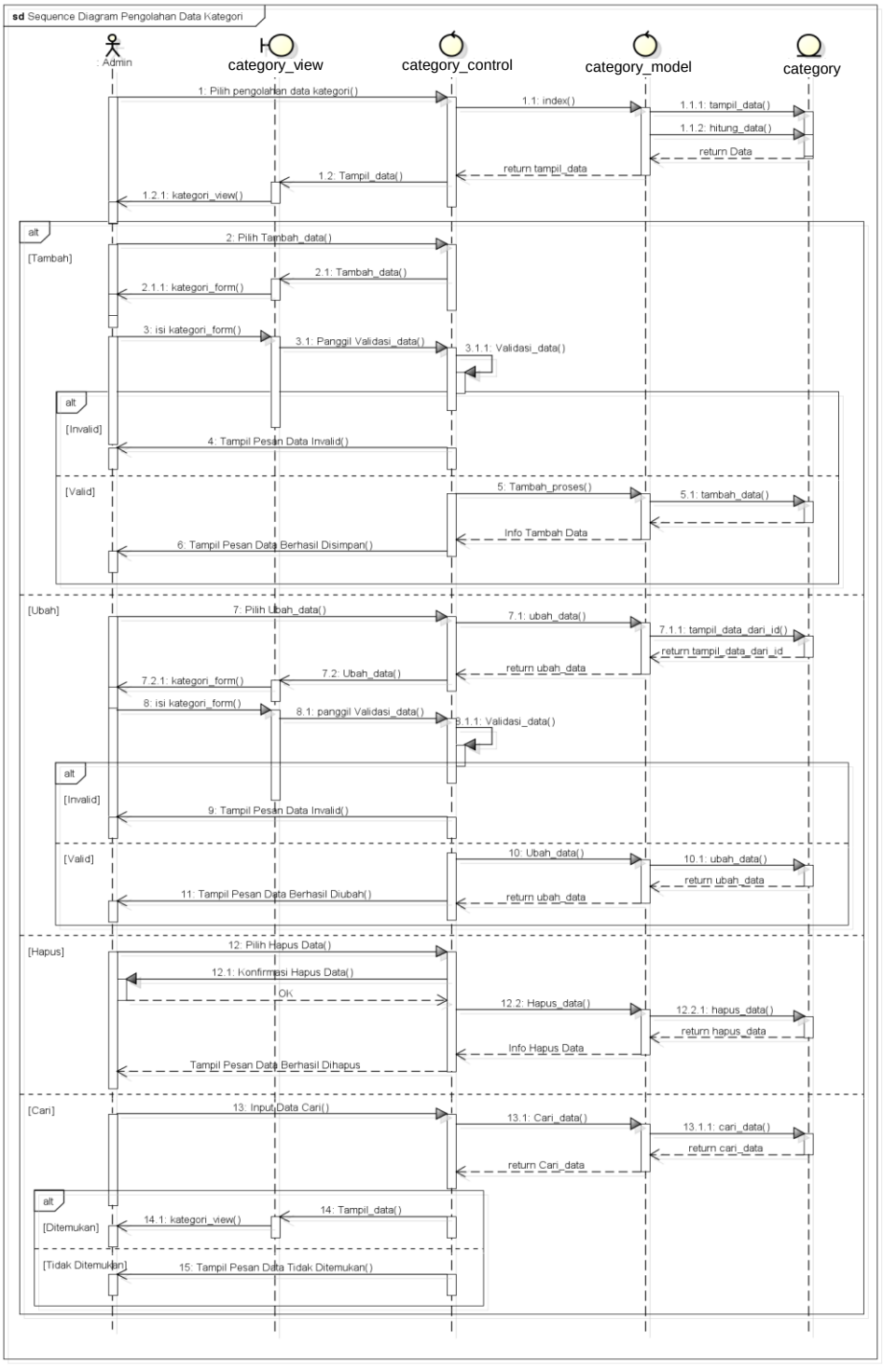
Sequence diagram Permintaan Password Baru dapat dilihat pada gambar 3.5



Gambar 3.5 *Sequence Diagram* Permintaan Password Baru

3. *Sequence Diagram* Pengolahan Data Kategori

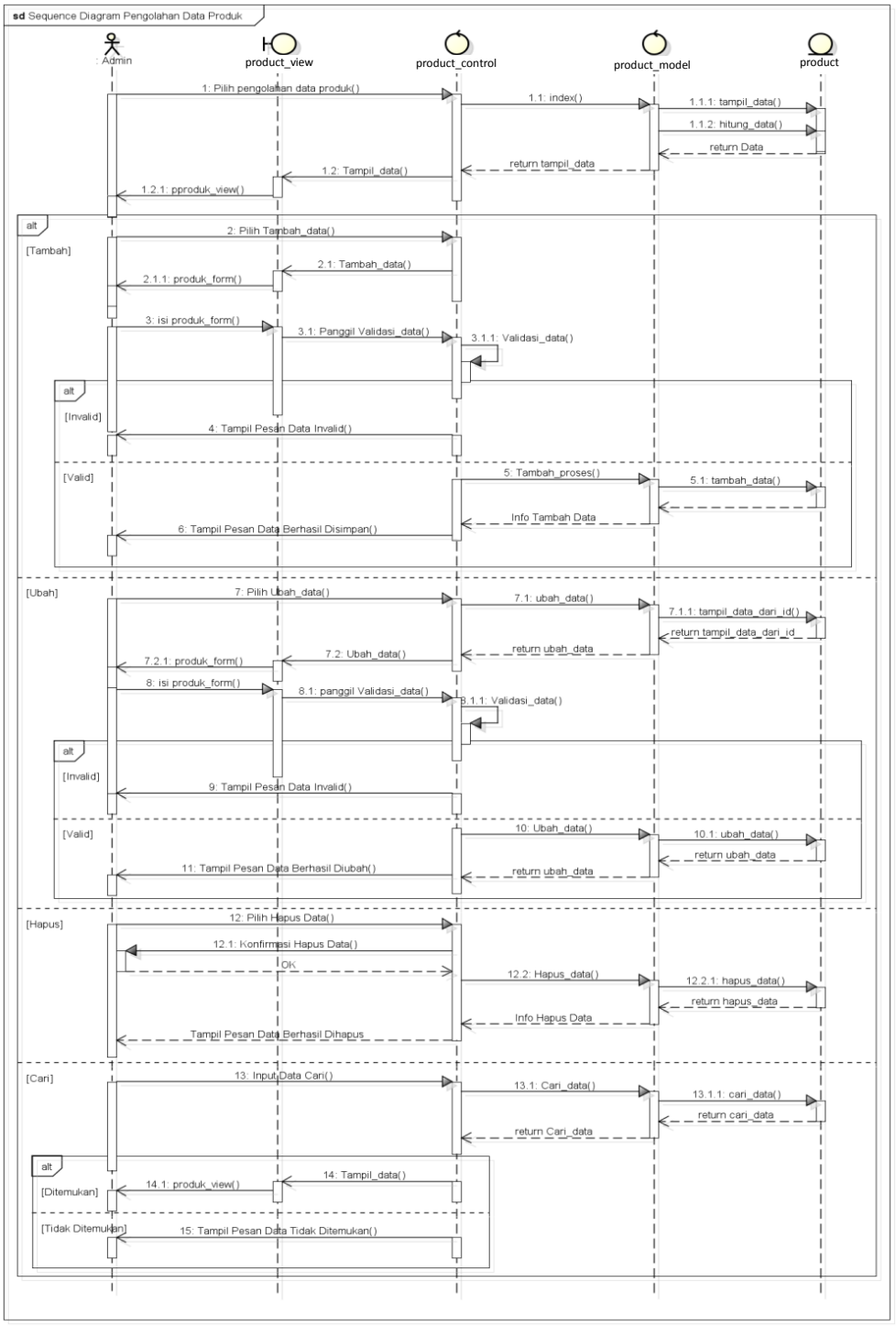
Sequence diagram pengolahan data kategori dapat dilihat pada gambar 3.6



Gambar 3.6 *Sequence diagram* pengolahan data kategori

4. Sequence Diagram Pengolahan Data Produk

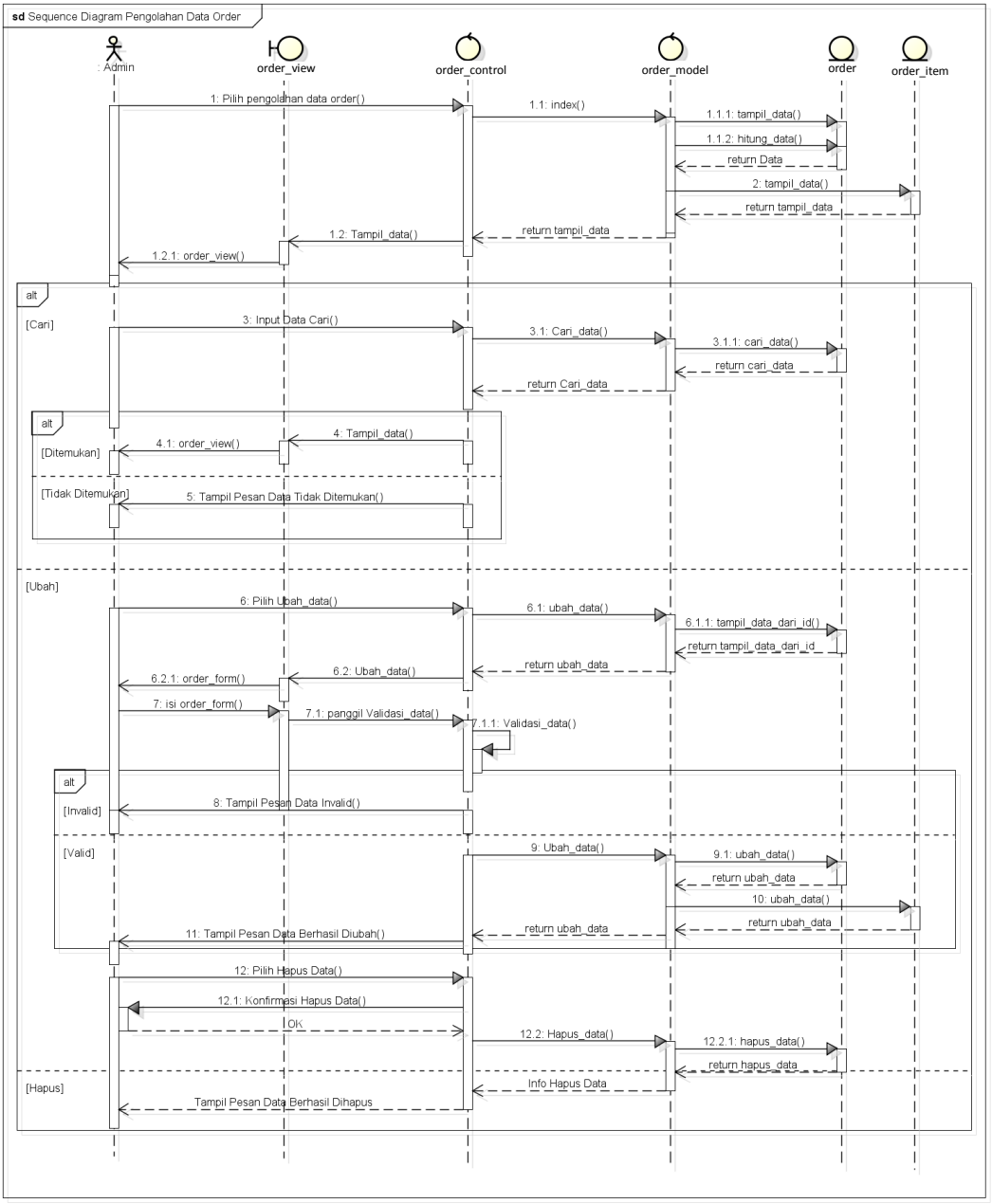
Sequence diagram pengolahan data produk dapat dilihat pada gambar 3.7



Gambar 3.7 *Sequence diagram* pengolahan data produk

5. *Sequence Diagram* Pengolahan Data Pemesanan

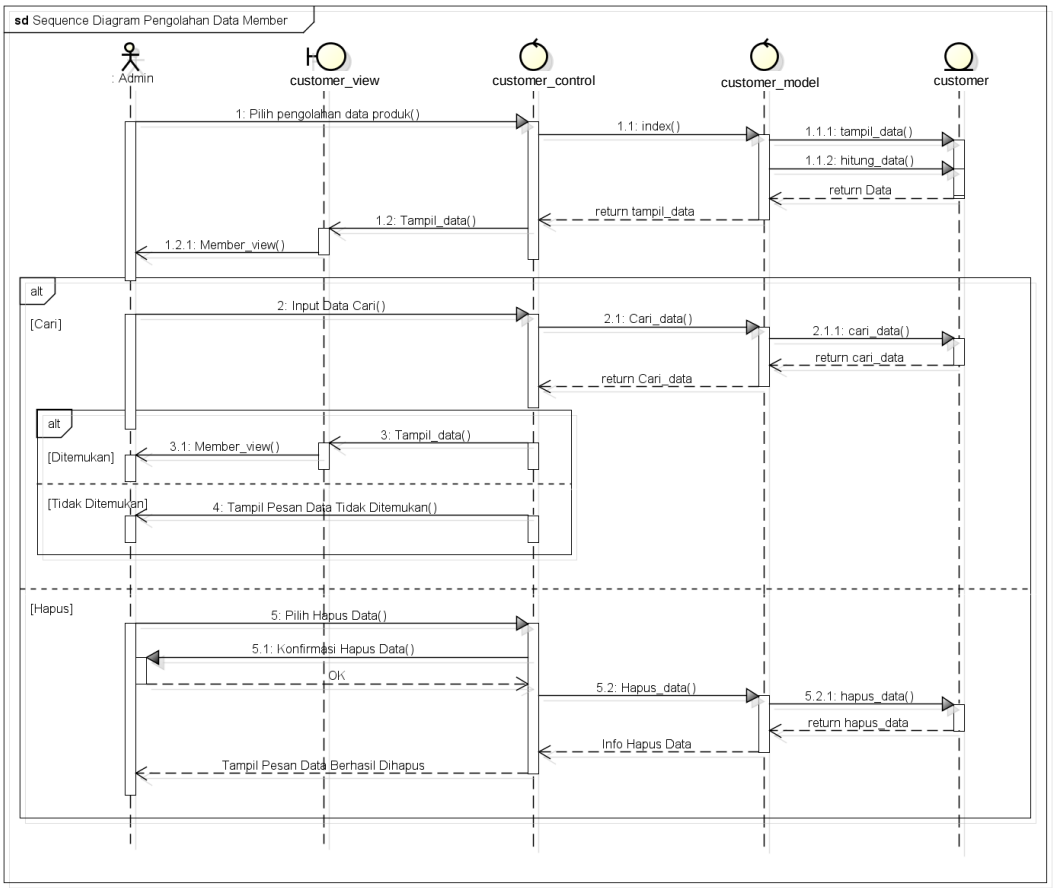
Sequence diagram pengolahan data pemesanan dapat dilihat pada gambar 3.8



Gambar 3.8 *Sequence Diagram* Pengolahan Data Pemesanan

6. *Sequence Diagram* Pengolahan Data Member

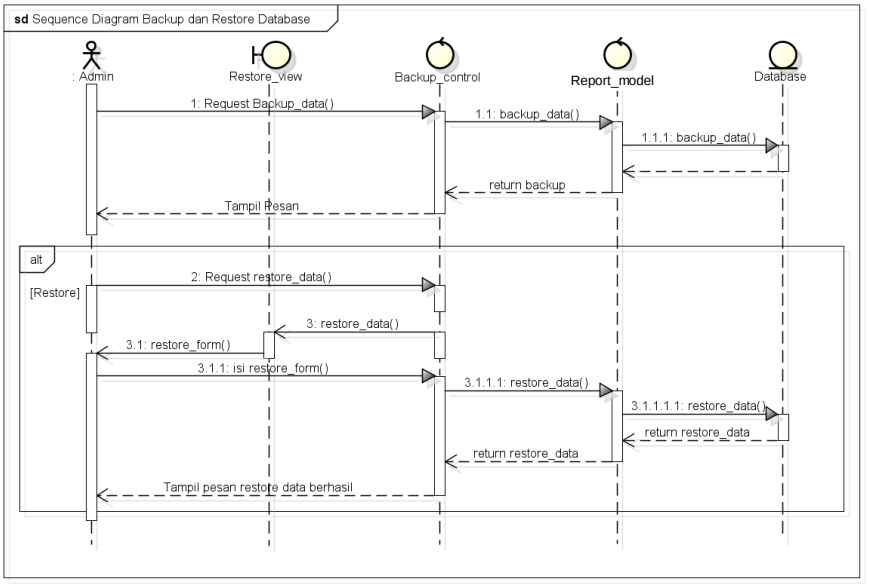
Sequence diagram pengolahan data member dapat dilihat pada gambar 3.9



Gambar 3.9 *Sequence Diagram* Pengolahan Data member

7. ***Sequence Diagram Backup* dan *Restore Database***

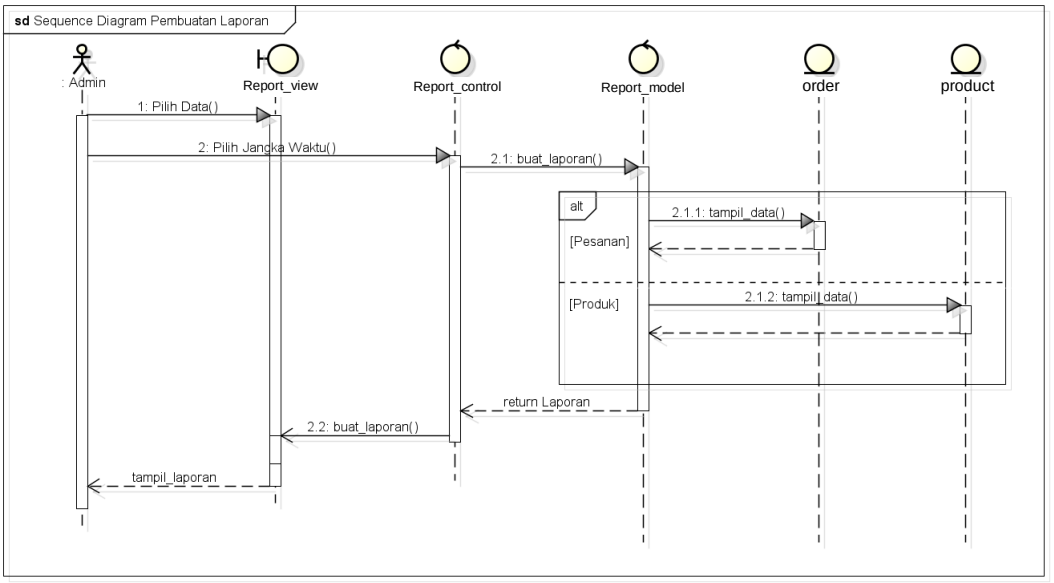
Sequence diagram backup dan restore database dapat dilihat pada gambar 3.1.



Gambar 3.10 *Sequence Diagram* Backup dan Restore *Database*

***Sequence Diagram* Pembuatan laporan**

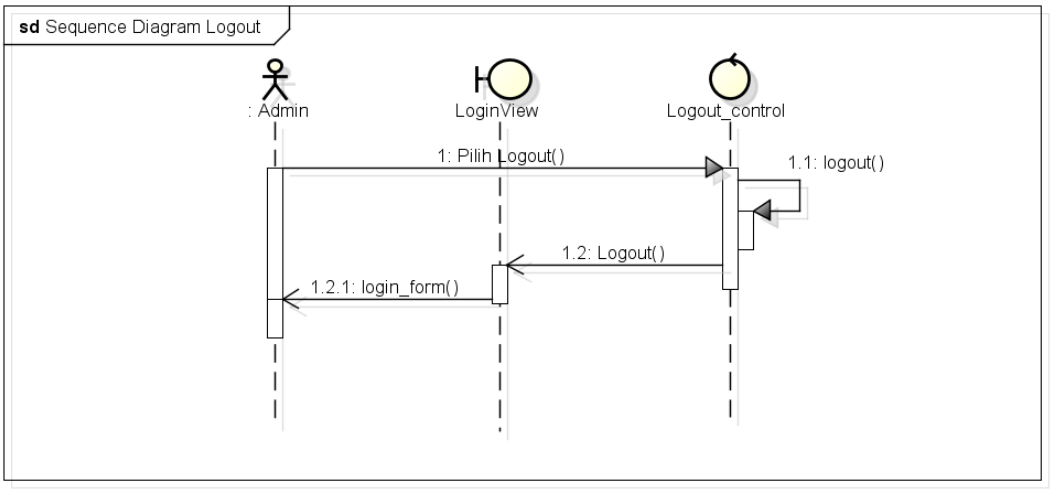
Sequence diagram pembuatan laporan dapat dilihat pada gambar 3.11



Gambar 3.11 *Sequence Diagram* Pembuatan Laporan

8. ***Sequence Diagram Logout***

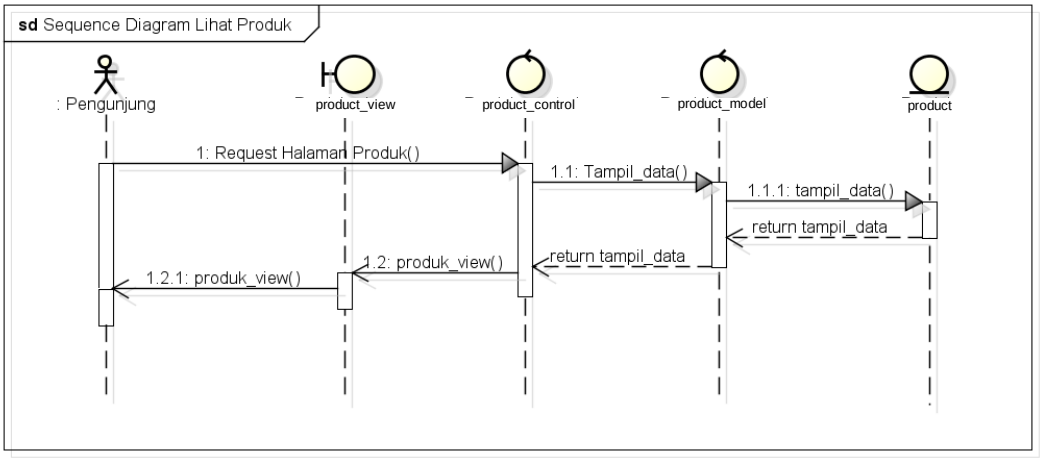
Sequence diagram Logout dapat dilihat pada gambar 3.12



Gambar 3.12 *Sequence Diagram Logout*

9. ***Sequence Diagram* Lihat Produk**

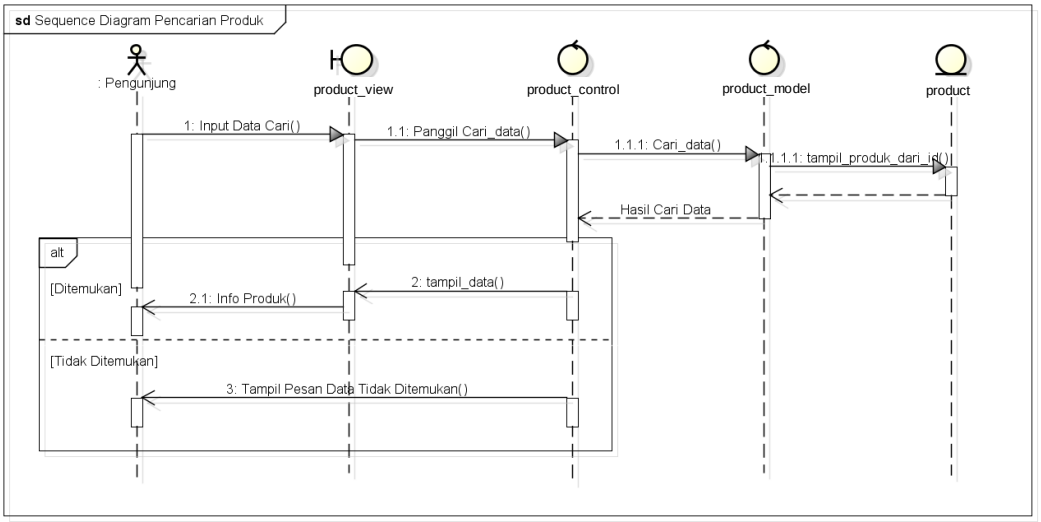
Sequence diagram lihat produk dapat dilihat pada gambar 3.13



Gambar 3.13 *Sequence Diagram* Lihat Produk

10. ***Sequence Diagram* Pencarian Produk**

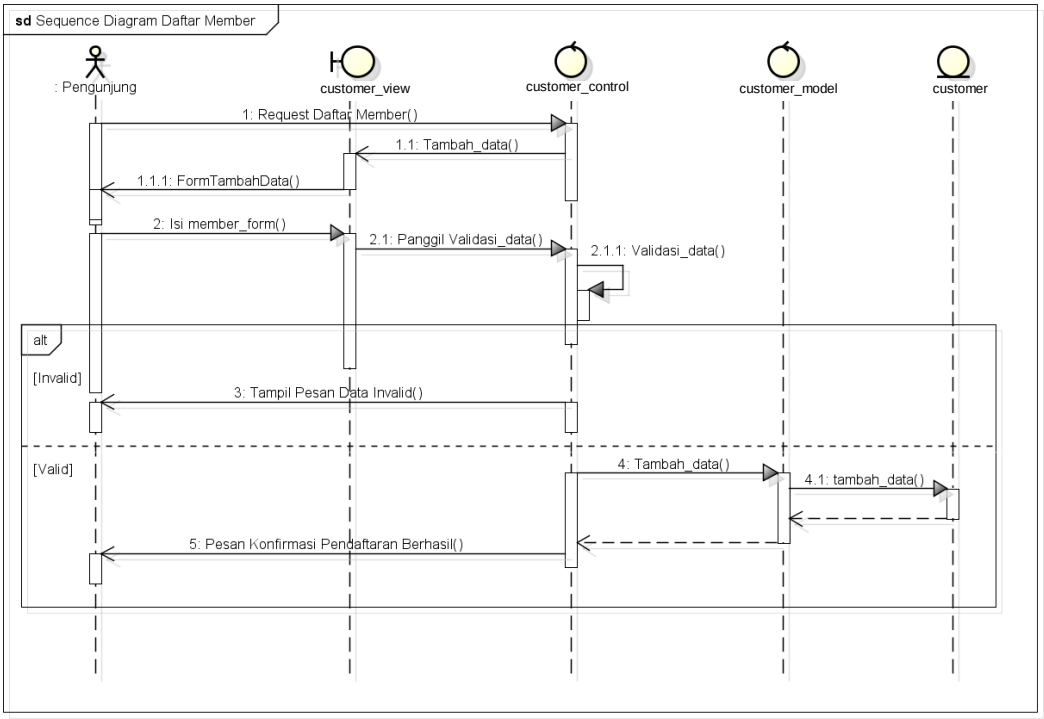
Sequence diagram pencarian produk dapat dilihat pada gambar 3.14



Gambar 3.14 *Sequence Diagram* Pencarian Produk

11. ***Sequence Diagram*** **Daftar Member**

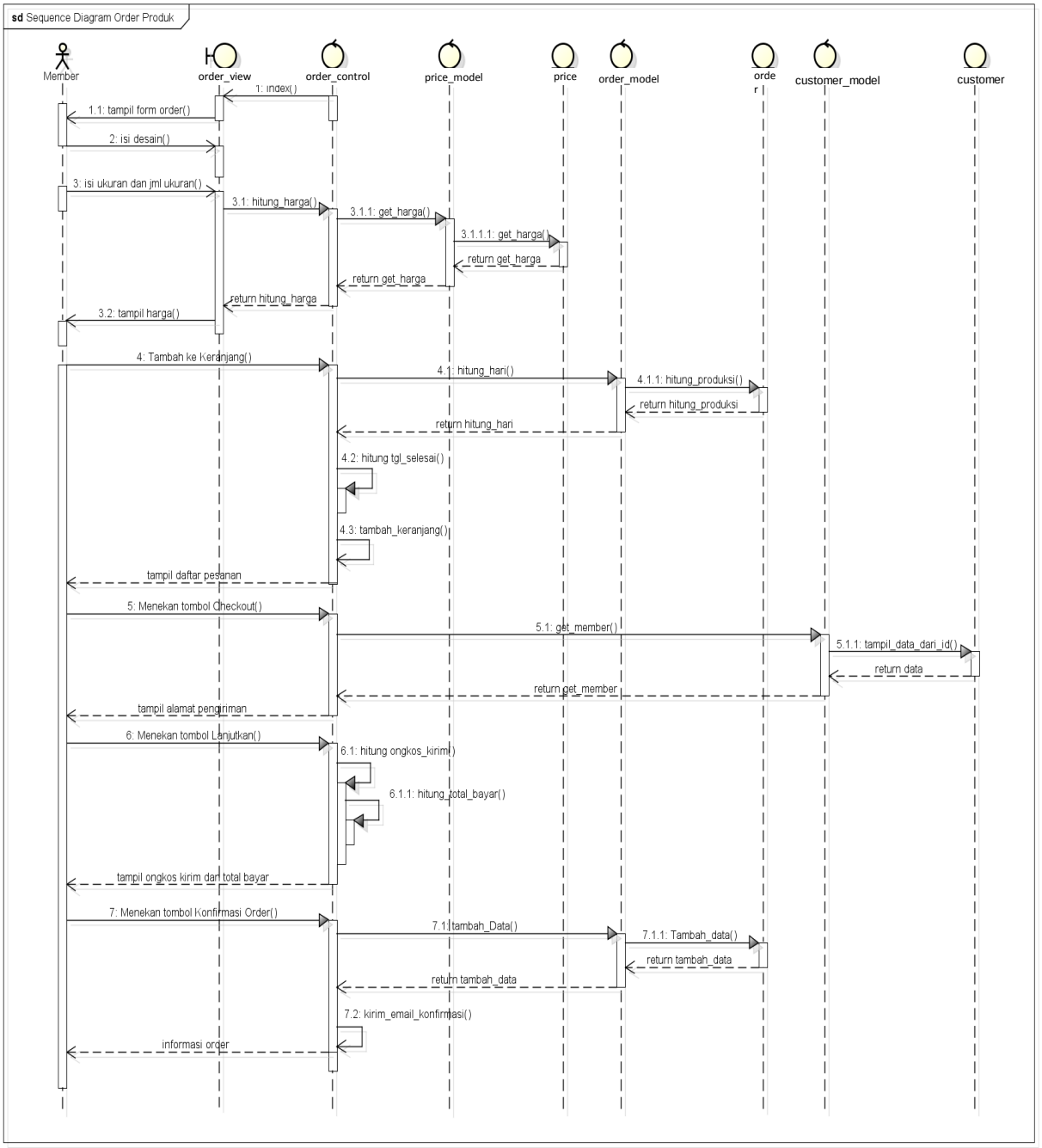
Sequence diagram daftar member dapat dilihat pada gambar 3.15



Gambar 3.15 *Sequence Diagram* Daftar Member

12. *Sequence Diagram* Pemesanan Produk

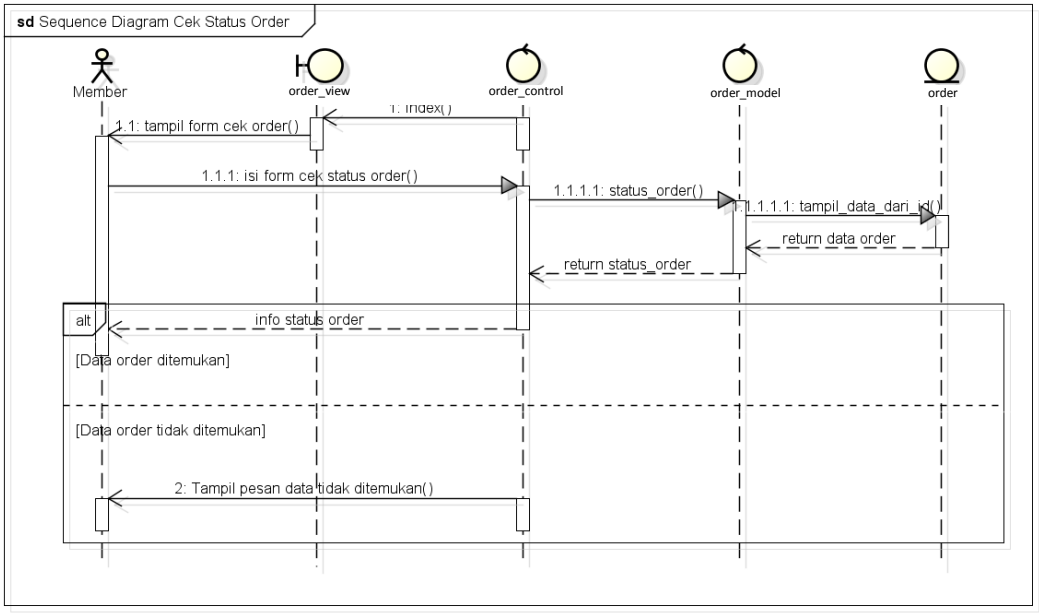
Sequence diagram Pemesanan produk dapat dilihat pada gambar 3.16



Gambar 3.16 *Sequence Diagram* Pemesanan Produk

13. ***Sequence Diagram* Cek Status Order**

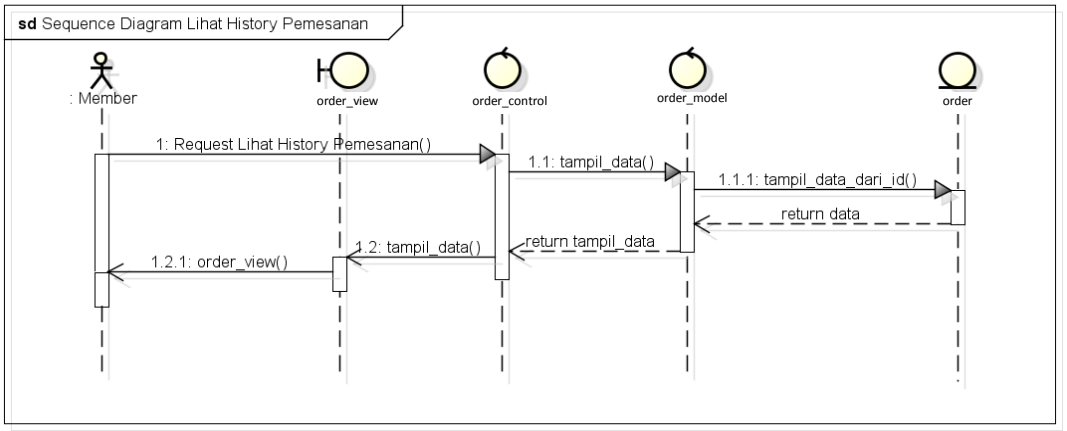
Sequence diagram cek status order dapat dilihat pada gambar 3.17



Gambar 3.17 *Sequence Diagram* Cek Status Order

14. ***Sequence Diagram* Lihat History Pemesanan**

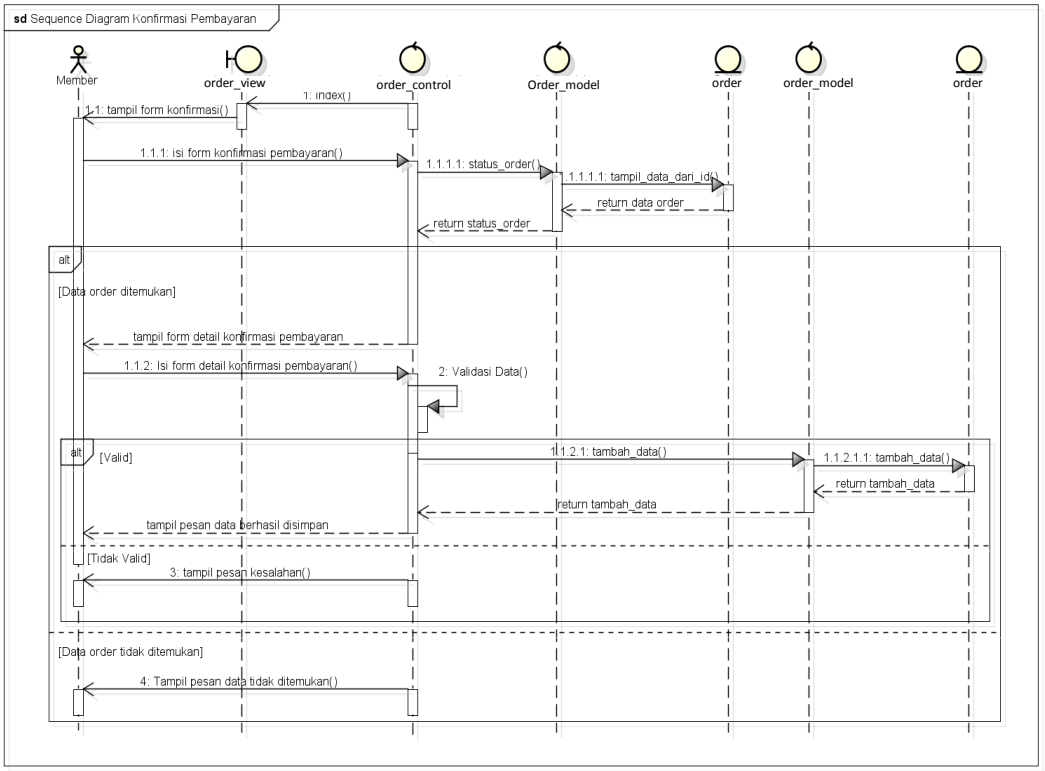
Sequence diagram lihat history pemesanan dapat dilihat pada gambar 3.18



Gambar 3.18 *Sequence Diagram* Lihat History Pemesanan

15. ***Sequence Diagram* Konfirmasi Pembayaran**

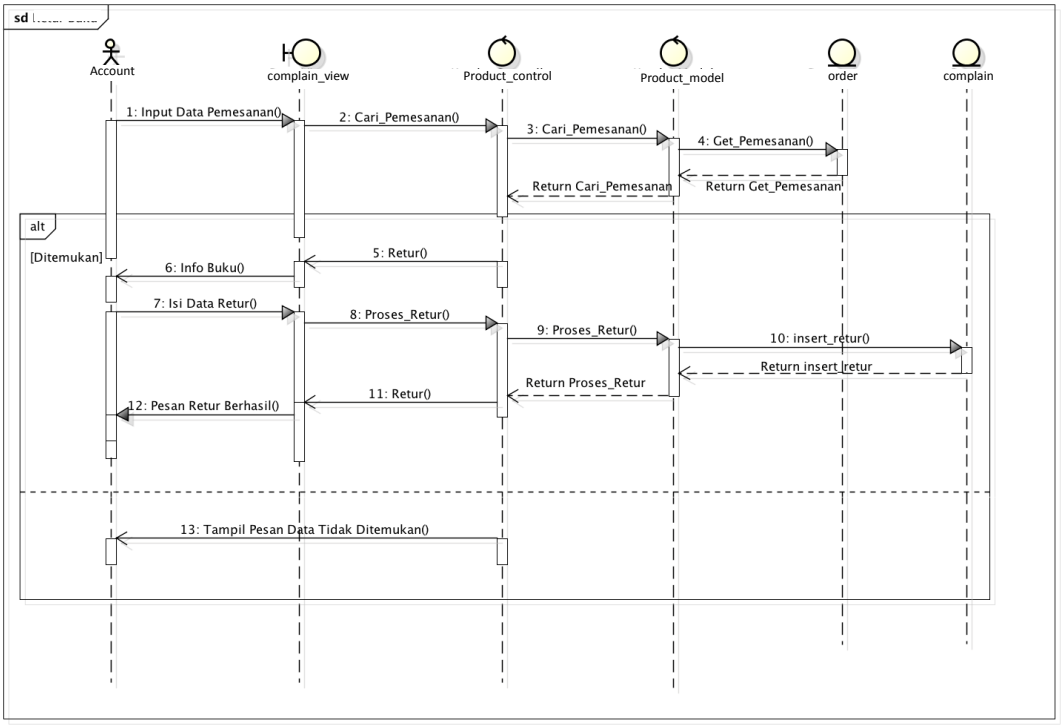
Sequence diagram konfirmasi pembayaran dapat dilihat pada gambar 3.19



Gambar 3.19 *Sequence Diagram* Konfirmasi Pembayaran

16. ***Sequence Diagram* Pengembalian Produk**

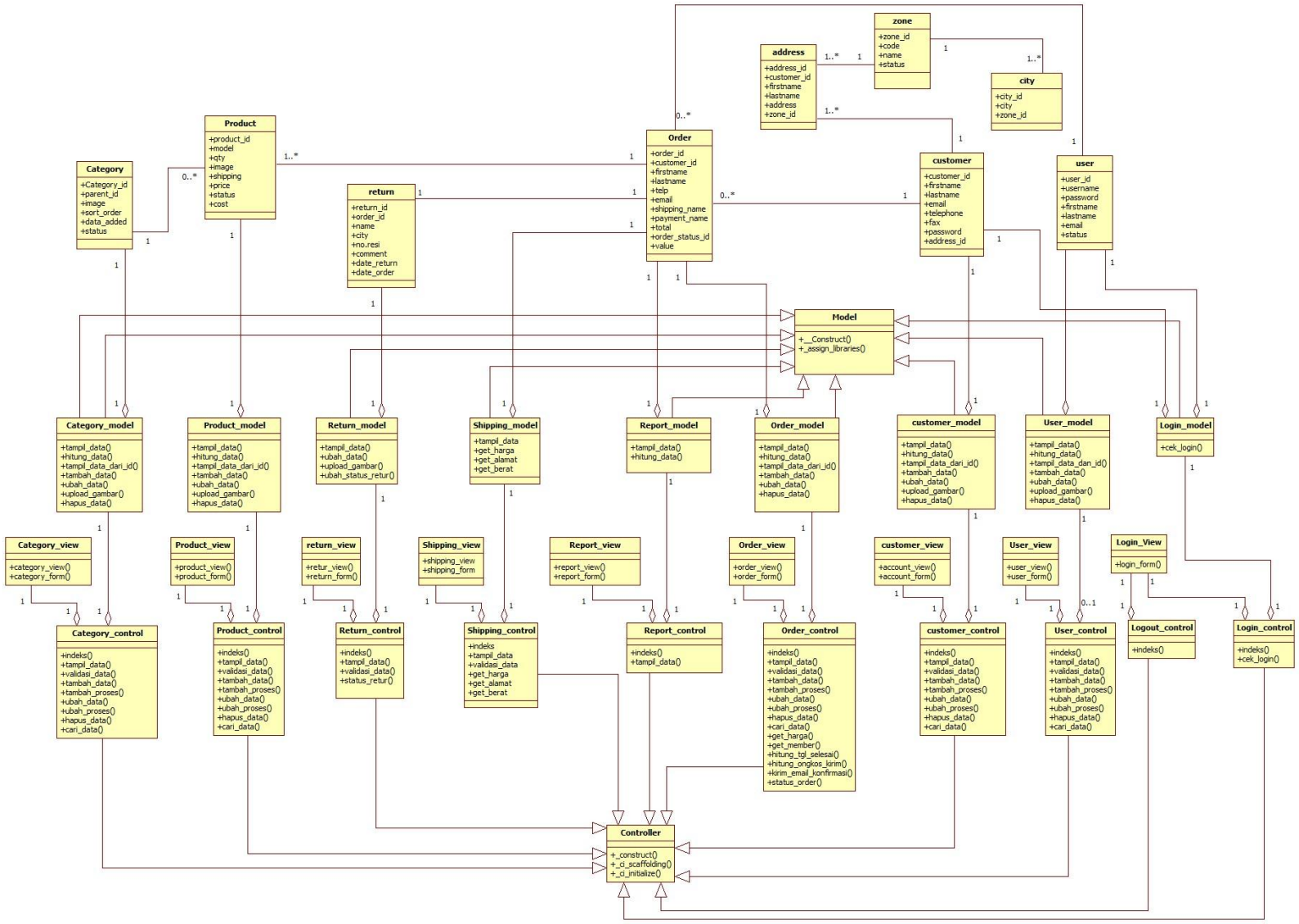
Sequence diagram Pengembalian Produk dapat dilihat pada gambar 3.20



Gambar 3.20 *Sequence Diagram* Pengembalian Produk

III. 3. 3. 8. *Class* Diagram

Class diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class diagram* juga menunjukkan properti dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan objek tersebut. *Class diagram E-commerce* Arsefa Collection dapat dilihat pada gambar 3.20.



Gambar 3.20 *Class diagram E-commerce* Arsefa Collection

Setiap kelas pada gambar 3.20 memiliki jenis kelasnya masing-masing seperti yang terlihat pada tabel 3.20.

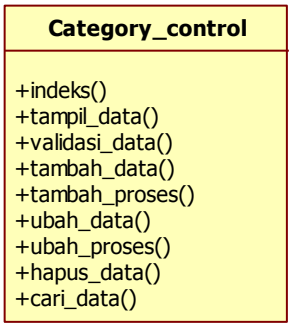
Tabel 3.20 Deskrpsi *Class*

No	Nama	Jenis
1.	category_control	Control
2.	product_control	Control
3.	report_control	Control
4.	order_control	Control
5.	customer_control	Control
6.	login_control	Control
7.	logout_control	Control
8.	return_control	Control
9.	shipping_control	Control
10.	user_control	Control
11.	category_model	Model
12.	product _model	Model
13.	report _model	Model
14.	order _model	Model
15.	customer _model	Model
16.	login_model	Model
17.	return_model	Model
18.	shipping_model	Model
19.	user_model	Model
20.	category _view	Boundary
21.	product _view	Boundary
22.	report _view	Boundary
23.	order _view	Boundary
24.	customer _view	Boundary
25.	login_view	Boundary
26.	return_view	Boundary
27.	shipping_view	Boundary
28.	user_view	Boundary
29.	category	Entity
30.	product	Entity
31.	order	Entity
32.	customer	Entity
33.	return	Entity
34.	address	Entity
35.	zone	Entity
36.	city	Entity
37.	user	Entity

Deskripsi dari setiap *class* yang ada dijelaskan sebagai berikut :

1. *Class* Category_control

Class Category_control merupakan *class* yang digunakan untuk mengatur proses pengolahan pada data kategori dan bertugas untuk memanggil *class-class* yang lainnya. Atribut dan operasi yang dimiliki *class* Category_control dapat dilihat pada gambar 3.21.



Gambar 3.21 Deskripsi *Class* Category_control

a. Deskripsi Atribut

Class category_control tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* category_control dapat dilihat pada tabel 3.21

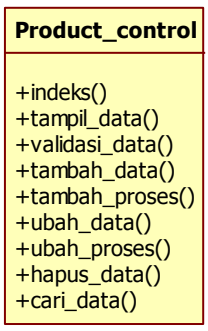
Tabel 3.21 Deskripsi Operasi *Class* category_control

Nama	Keterangan
index() :	Merupakan Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data kategori
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* Category_model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data kategori yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
tambah_data() : void	Prosedur untuk melakukan tambah data. Memanggil Category_View untuk menampilkan form tambah.

Tabel 3.21 Deskripsi Operasi *Class* category_control (lanjutan)	
tambah_proses() : void	Prosedur untuk melakukan proses penambahan data kategori berdasarkan data masukan dari form tambah dan akan memanggil *class* Category_Model untuk melakukan proses tambah data kategori dan menyimpannya pada *database*.
ubah_data() : void	Prosedur untuk melakukan perubahan data kategori, akan memanggil *class* Category_Model untuk mengambil data kemudian memanggil *class* Category View untuk menampilkan form ubah.
ubah_proses() : void	Prosedur untuk memproses data masukan dari form ubah, memanggil Category_Model untuk menyimpan data yang sudah diubah.
hapus_data() : void	Prosedur untuk menghapus data kategori, akan menampilkan pesan konfirmasi apakah data category akan dihapus, jika ya maka akan memanggil Category_Model untuk melakukan penghapusan data pada *database*.
cari_data() : void	Prosedur untuk mencari data kategori pada *database* dan menampilkannya.

2. *Class* Product_control

Class Product_control merupakan *class* yang digunakan untuk mengatur proses pengolahan pada data produk dan bertugas untuk memanggil *class-class* yang berkaitan. Atribut dan operasi yang dimiliki *class* Product_control dapat dilihat pada gambar 3.22.



Gambar 3.22 Deskripsi *Class* Product_control

a. Deskripsi Atribut

Class Product_control tidak memiliki atribut.

b. Deskripsi Operasi

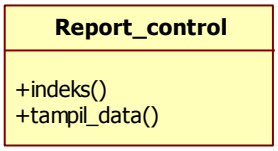
Deskripsi operasi *class* product_control dapat dilihat pada tabel 3.22

Tabel 3.22 Deskripsi Operasi *Class* product_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data produk
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* Product_Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data produk yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
tambah_data() : void	Prosedur untuk melakukan tambah data. Memanggil Product_View untuk menampilkan form tambah.
tambah_proses() : void	Prosedur untuk melakukan proses penambahan data produk berdasarkan data masukan dari form tambah dan akan memanggil *class* Product_Model untuk melakukan proses tambah data produk dan menyimpannya pada *database*.
ubah_data() : void	Prosedur untuk melakukan perubahan data produk, akan memanggil *class* Product_Model untuk mengambil data kemudian memanggil *class* Product_View untuk menampilkan form ubah.
ubah_proses() : void	Prosedur untuk memproses data masukan dari form ubah, memanggil Product_Model untuk menyimpan data yang sudah diubah.
hapus_data() : void	Prosedur untuk menghapus data produk, akan menampilkan pesan konfirmasi apakah data produk akan dihapus, jika ya maka akan memanggil Product_Model untuk melakukan penghapusan data pada *database*.
cari_data() : void	Prosedur untuk mencari data produk pada *database* dan menampilkannya.

3. *Class* Report_control

Class Report_control merupakan *class* yang digunakan untuk mengatur proses pembuatan laporan dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* Report_control dapat dilihat pada gambar 3.23



Gambar 3.23 Deskripsi *Class* Report_control

a. Deskripsi Atribut

Class Report_control tidak memiliki atribut.

b. Deskripsi Operasi

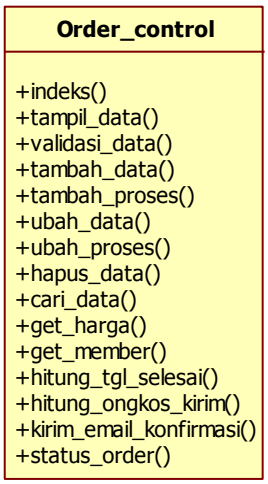
Deskripsi operasi *class* report_control dapat dilihat pada tabel 3.23.

Tabel 3.23 Deskripsi Operasi *Class* report_control

Nama	Keterangan
Index() : void	Prosedur yang pertama kali dijalankan, akan memanggil form Periode pada Report_View.
tampil_data() : void	Prosedur untuk melakukan pembuatan laporan form yang sudah diisi sebelumnya dan kemudian menampilkannya.

4. *Class* Order_control

Class Order_control merupakan *class* yang digunakan untuk mengatur proses pengolahan pada data order dan bertugas untuk memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* Order_control dapat dilihat pada gambar 3.24



Gambar 3.24 Deskripsi *Class* order_control

a. Deskripsi Atribut

Class Order_control tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* order_control dapat dilihat pada tabel 3.24

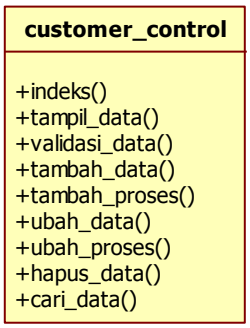
Tabel 3.24 Deskripsi Operasi *Class* order_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampilData untuk menampilkan data order
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* Order_Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data order yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
tambah_data() : void	Prosedur untuk melakukan tambah data. Memanggil Order_View untuk menampilkan form tambah.
tambah_proses() : void	Prosedur untuk melakukan proses penambahan data order berdasarkan data masukan dari form tambah dan akan memanggil *class* Order_Model untuk melakukan proses tambah data order dan menyimpannya pada *database*.

Tabel 3.24 Deskripsi Operasi *Class* order_control (lanjutan)	
ubah_data() : void	Prosedur untuk melakukan perubahan data order, akan memanggil *class* Order_Model untuk mengambil data kemudian memanggil *class* Order_View untuk menampilkan form ubah.
ubah_proses() : void	Prosedur untuk memproses data masukan dari form ubah, memanggil Order_Model untuk menyimpan data yang sudah diubah.
hapus_data() : void	Prosedur untuk menghapus data order, akan menampilkan pesan konfirmasi apakah data order akan dihapus, jika ya maka akan memanggil Order_Mode untuk melakukan penghapusan data pada *database*.
cari_data() : void	Prosedur untuk mencari data order pada *database* dan menampilkannya.
Get_harga() : void	Prosedur untuk mengambil harga dari tabel harga
Get_account : void	Prosedur untuk mengambil data member
Hitung_tgl_selesai() : void	Prosedur untuk menghitung tanggal selesai
Hitung_ongkos_kirim() :void	Prosedur untuk menghitung ongkos kirim
Hitung_total_bayar() : void	Prosedur untuk menghitung total bayar
Kirim_email_konfirmasi(): void	Prosedur untuk mengirim email konfirmasi
Status_order(): void	Prosedur untuk melihat status suatu order/pesanan

5. *Class* Customer_control

Class customer_control merupakan *class* yang digunakan untuk mengatur proses pengolahan pada data member dan bertugas untuk memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* customer_control dapat dilihat pada gambar 3.25



Gambar 3.25 Deskripsi *Class* customer_control

a. Deskripsi Atribut

Class customer _control tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* customer _control dapat dilihat pada tabel 3.25

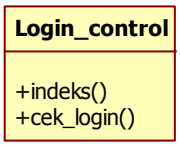
Tabel 3.25 Deskripsi Operasi *Class* customer_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data member
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* customer _Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data member yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
tambah_data() : void	Prosedur untuk melakukan tambah data. Memanggil customer _View untuk menampilkan form tambah.
tambah_proses() : void	Prosedur untuk melakukan proses penambahan data member berdasarkan data masukan dari form tambah dan akan memanggil *class* customer _Model untuk melakukan proses tambah data member dan menyimpannya pada *database*.
ubah_data() : void	Prosedur untuk melakukan perubahan data member, akan memanggil *class* customer _Model untuk mengambil data kemudian memanggil *class* Account View untuk menampilkan form ubah.

Tabel 3.25 Deskripsi Operasi *Class* account_control (Lanjutan)	
ubah_proses() : void	Prosedur untuk memproses data masukan dari form ubah, memanggil customer _Model untuk menyimpan data yang sudah diubah.
hapus_data() : void	Prosedur untuk menghapus data member, akan menampilkan pesan konfirmasi apakah data member akan dihapus, jika ya maka akan memanggil customer _Model untuk melakukan penghapusan data pada *database*.
cari_data() : void	Prosedur untuk mencari data member pada *database* dan menampilkannya.

6. *Class* Login_control

Class Login_control merupakan *class* yang digunakan untuk mengatur proses login untuk pengguna atau member dan bertugas untuk memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* Login_control dapat dilihat pada gambar 3.26



Gambar 3.26 Deskripsi *Class* Login_control

a. Deskripsi Atribut

Class Login_control tidak memiliki atribut.

b. Deskripsi Operasi

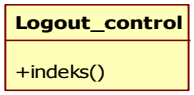
Deskripsi operasi *class* login_control dapat dilihat pada tabel 3.26

Tabel 3.26 Deskripsi Operasi *Class* login_control

Nama	Keterangan
Index() : void	Prosedur yang pertama dijalankan dan akan memanggil Login_view untuk menampilkan form login.
Cek_login() : void	Prosedur yang dipanggil ketika user melakukan proses login, memvalidasi dan memanggil *class* Login_Model untuk mencari data user yang ada pada *database*.

7. *Class* Logout_control

Class Logout_control merupakan *class* yang digunakan untuk melakukan proses logout admin atau member dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* Logout_control dapat dilihat pada gambar 3.27



Gambar 3.27 Deskripsi *Class* Logout_control

a. Deskripsi Atribut

Class Logout_control tidak memiliki atribut.

b. Deskripsi Operasi

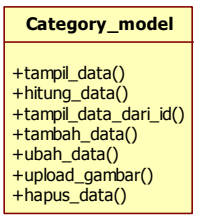
Deskripsi operasi *class* logout_control dapat dilihat pada tabel 3.27

Tabel 3.27 Deskripsi Operasi *Class* logout_control

Nama	Keterangan
index() : void	Prosedur ketika memanggil logout_control, digunakan untuk keluar dari sistem dan kembali menampilkan form login.

8. *Class* Category_model

Class Category_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data kategori. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* Category_model dapat dilihat pada gambar 3.28



Gambar 3.28 Deskripsi *Class* Category_model

a. Deskripsi Atribut

Class Category_model tidak memiliki atribut.

b. Deskripsi Operasi

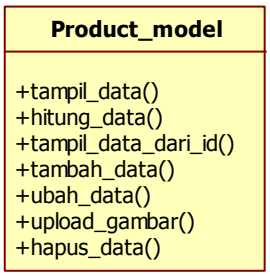
Deskripsi operasi *class* category_model dapat dilihat pada tabel 3.28

Tabel 3.28 Deskripsi Operasi *Class* category_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data kategori.
hitung_data() : void	Prosedur menghitung jumlah data pada *database*
Tampil_Data_dari_Id() : void	Prosedur untuk mengambil data dari *database* berdasarkan id kategori.
tambah_data() : void	Prosedur yang dipanggil ketika proses tambah data kategori, melakukan perintah simpan data pada *database.*
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data kategori, melakukan perintah update data pada *database*.
upload_gambar() : void	Prosedur untuk menyimpan gambar yang diupload dan menyimpan alamatnya pada *database*.
hapus_data() : void	Prosedur yang dipanggil ketika hapus data, menghapus data kategori pada *database*.

9. *Class* Product_model

Class Product_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data produk. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* Product_model dapat dilihat pada gambar 3.29



Gambar 3.29 Deskripsi *Class* Product_model

a. Deskripsi Atribut

Class Product_model tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* product_model dapat dilihat pada tabel 3.29

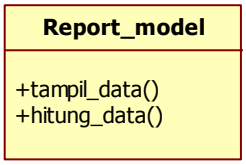
Tabel 3.29 Deskripsi Operasi *Class* product_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data produk.
Hitung_data() : void	Prosedur menghitung jumlah data pada *database*
Tampil_Data_dari_Id() : void	Prosedur untuk mengambil data dari *database* berdasarkan id produk.
tambah_data() : void	Prosedur yang dipanggil ketika proses tambah data produk, melakukan perintah simpan data pada *database*.
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data produk, melakukan perintah update data pada *database*.
upload_gambar() : void	Prosedur untuk menyimpan gambar yang diupload dan menyimpan alamatnya pada *database*.
hapus_data() : void	Prosedur yang dipanggil ketika hapus data, menghapus data produk pada *database*.

10. *Class* Report_model

Class Report_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data laporan. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang

dimiliki *class* Report_model dapat dilihat pada gambar 3.30



Gambar 3.30 Deskripsi *Class* Report_model

a. Deskripsi Atribut

Class Report_model tidak memiliki atribut.

b. Deskripsi Operasi

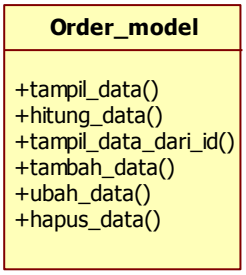
Deskripsi operasi *class* report_model dapat dilihat pada tabel 3.30

Tabel 3.30 Deskripsi Operasi *Class* report_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data berdasarkan jangka waktu yang dimasukan.
Hitung_data() : void	Prosedur menghitung jumlah data pada *database*

11. *Class* Order_model

Class Order_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data order. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* Order_model dapat dilihat pada gambar 3.31



Gambar 3.31 Deskripsi *Class* Order_model

a. Deskripsi Atribut

Class Order_model tidak memiliki atribut.

b. Deskripsi Operasi

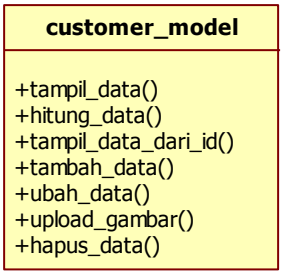
Deskripsi operasi *class* order_model dapat dilihat pada tabel 3.31.

Tabel 3.31 Deskripsi Operasi *Class* order_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data order.
Hitung_data() : void	Prosedur menghitung jumlah data pada *database*
Tampil_data_dari_id() : void	Prosedur untuk mengambil data dari *database* berdasarkan id order.
tambah_data() : void	Prosedur yang dipanggil ketika proses tambah data order, melakukan perintah simpan data pada *database*.
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data order, melakukan perintah update data pada *database*.
hapus_data() : void	Prosedur yang dipanggil ketika hapus data, menghapus data order pada *database*.

12. *Class* Customer_model

Class customer_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data member. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* customer _model dapat dilihat pada gambar 3.32



Gambar 3.32 Deskripsi *Class* customer_model

a. Deskripsi Atribut

Class customer _model tidak memiliki atribut.

b. Deskripsi Operasi

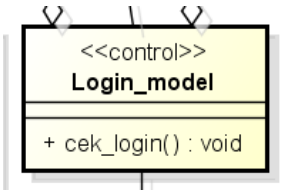
Deskripsi operasi *class* customer _model dapat dilihat pada tabel 3.32

Tabel 3.32 Deskripsi Operasi *Class* customer_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data member.
hitung_data() : void	Prosedur menghitung jumlah data pada *database*
tampil_data_dari_Id() : void	Prosedur untuk mengambil data dari *database* berdasarkan id member.
tambah_data() : void	Prosedur yang dipanggil ketika proses tambah data member, melakukan perintah simpan data pada *database*.
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data member, melakukan perintah update data pada *database*.
Upload_gambar() : void	Prosedur untuk menyimpan gambar yang diupload dan menyimpan alamatnya pada *database*.
hapus_data() : void	Prosedur yang dipanggil ketika hapus data, menghapus data member pada *database*.

13. *Class* Login_model

Class Login_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data login. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* Login_model dapat dilihat pada gambar 3.33



Gambar 3.33 Deskripsi *Class* Login_model

a. Deskripsi Atribut

Class Login_model tidak memiliki atribut.

b. Deskripsi Operasi

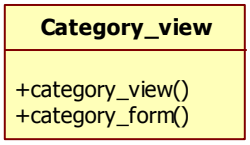
Deskripsi operasi *class* login_model dapat dilihat pada tabel 3.33.

Tabel 3.33 Deskripsi Operasi *Class* login_model

Nama	Keterangan
Cek_login() : void	Prosedur ketika user akan login, mengecek apakah data login yang dimasukan terdaftar pada *database*.

14. *Class* Category_view

Class Category_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* Category_view dapat dilihat pada gambar 3.34



Gambar 3.34 Deskripsi *Class* Category_view

a. Deskripsi Atribut

Class Category_view tidak memiliki atribut.

b. Deskripsi Operasi

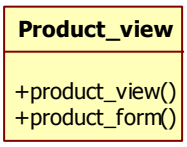
Deskripsi operasi *class* category_view dapat dilihat pada tabel 3.34

Tabel 3.34 Deskripsi Operasi *Class* category_view

Nama	Keterangan
Category_view() : void	Prosedur untuk menampilkan data kategori yang hasil prosedur dari *class* Category_model dan Category_control.
Category_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambah data dan ubah data pada *class* Category_control.

15. *Class* Product_view

Class Product_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan pengunjung. Atribut dan operasi yang dimiliki *class* Product_view dapat dilihat pada gambar 3.35



Gambar 3.35 Deskripsi *Class* Product_view

a. Deskripsi Atribut

Class Product_view tidak memiliki atribut.

b. Deskripsi Operasi

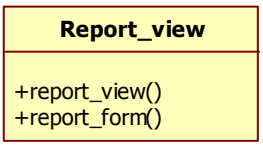
Deskripsi operasi *class* product_view dapat dilihat pada tabel 3.35

Tabel 3.35 Deskripsi Operasi *Class* product_view

Nama	Keterangan
Product_view() : void	Prosedur untuk menampilkan data produk yang hasil prosedur dari *class* Product_model dan Product_control.
Product_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambah data dan ubah data pada *class* Product_control.

16. *Class* Report_view

Class Report_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* Report_view dapat dilihat pada gambar 3.36



Gambar 3.36 Deskripsi *Class* Report_view

a. Deskripsi Atribut

Class Report_view tidak memiliki atribut.

b. Deskripsi Operasi

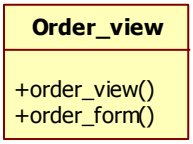
Deskripsi operasi *class* report_view dapat dilihat pada tabel 3.36

Tabel 3.36 Deskripsi Operasi *Class* report_view

Nama	Keterangan
Report_view() : void	Prosedur untuk menampilkan data laporan yang hasil prosedur dari *class* Report_model dan Report_control.
Report_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* Report_control.

17. *Class* Order_view

Class Order_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* Order_view dapat dilihat pada gambar 3.37



Gambar 3.37 Deskripsi *Class* Order_view

a. Deskripsi Atribut

Class Order_view tidak memiliki atribut.

b. Deskripsi Operasi

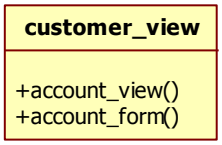
Deskripsi operasi *class* order_view dapat dilihat pada tabel 3.37

Tabel 3.37 Deskripsi Operasi *Class* order_view

Nama	Keterangan
Order_view() : void	Prosedur untuk menampilkan data order yang hasil prosedur dari *class* Order_model dan Order_control.
Order_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* Order_control.

18. *Class* customer_view

Class customer_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* customer _view dapat dilihat pada gambar 3.38



Gambar 3.38 Deskripsi *Class* customer_view

a. Deskripsi Atribut

Class customer _view tidak memiliki atribut.

b. Deskripsi Operasi

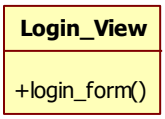
Deskripsi operasi *class* customer _view dapat dilihat pada tabel 3.38

Tabel 3.38 Deskripsi Operasi *Class* customer_view

Nama	Keterangan
customer _view() : void	Prosedur untuk menampilkan data member berdasarkan hasil prosedur *class* customer _model dan customer _control.
customer _form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* customer _Control.

19. *Class* Login_view

Class Login_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* Login_view dapat dilihat pada gambar 3.39



Gambar 3.39 Deskripsi *Class* Login_view

a. Deskripsi Atribut

Class Login_view tidak memiliki atribut.

b. Deskripsi Operasi

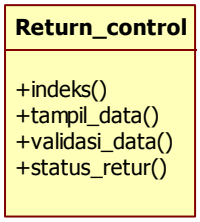
Deskripsi operasi *class* login_view dapat dilihat pada tabel 3.39

Tabel 3.39 Deskripsi Operasi *Class* login_view

Nama	Keterangan
login_form() : void	Prosedur untuk menampilkan form yang akan digunakan untuk proses login.

20. *Class* Return_control

Class return_control merupakan *class* yang digunakan untuk melakukan proses pengembalian barang oleh member dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* return_control dapat dilihat pada gambar 3.40



Gambar 3.40 Deskripsi *Class* return_control

a. Deskripsi Atribut

Class return_control tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* return_control dapat dilihat pada tabel 3.40

Tabel 3.40 Deskripsi Operasi *Class* return_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data return
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* return _Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data return yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
status_return() : void	Prosedur untuk mengubah status retur

21. *Class* Shipping Control

Class shipping_control merupakan *class* yang digunakan untuk melakukan proses pengiriman barang oleh admin dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* shipping_control dapat dilihat pada gambar 3.41



Gambar 3.41 Deskripsi *Class* shipping_control

a. Deskripsi Atribut

Class shipping_control tidak memiliki atribut.

b. Deskripsi Operasi

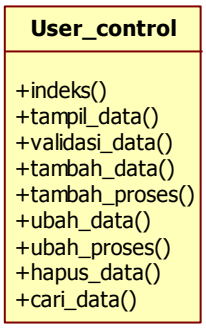
Deskripsi operasi *class* shipping_control dapat dilihat pada tabel 3.41

Tabel 3.41 Deskripsi Operasi *Class* shipping_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data return
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* return _Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data return yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
get_harga() : void	Prosedur untuk mengambil data harga
get_alamat() : void	Prosedur untuk mengambil data alamat
get_berat() : void	Prosedur untuk mengambil data berat

22. *Class* User_Control

Class user_control merupakan *class* yang digunakan untuk melakukan proses pengolahan data admin dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* user_control dapat dilihat pada gambar 3.42



Gambar 3.42 Deskripsi *Class* user_control

a. Deskripsi Atribut

Class user_control tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* user_control dapat dilihat pada tabel 3.42

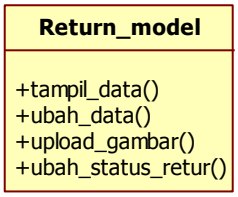
Tabel 3.42 Deskripsi Operasi *Class* user_control

Nama	Keterangan
index() :	Prosedur yang dijalankan ketika *class* pertama kali dipanggil, prosedur ini akan memanggil prosedur tampil Data untuk menampilkan data user
tampil_data() : void	prosedur ini akan memberi perintah kepada *class* user _Model untuk mengambil data pada *database*.
validasi_data() : void	Prosedur yang dipanggil untuk memvalidasi data user yang dimasukan pada form tambah atau ubah, akan menampilkan kesalahan jika data yang dimasukan tidak valid.
tambah_data() : void	Prosedur untuk melakukan tambah data. Memanggil user _View untuk menampilkan form tambah.
tambah_proses() : void	Prosedur untuk melakukan proses penambahan data user berdasarkan data masukan dari form tambah dan akan memanggil *class* user _Model untuk melakukan proses tambah data user dan menyimpannya pada *database*.
ubah_data() : void	Prosedur untuk melakukan perubahan data user, akan memanggil *class* user _Model untuk mengambil data kemudian memanggil *class* user_View untuk menampilkan form ubah.
ubah_proses() : void	Prosedur untuk memproses data masukan dari form ubah, memanggil user_Model untuk menyimpan data yang sudah diubah.
hapus_data() : void	Prosedur untuk menghapus data user, akan menampilkan pesan konfirmasi apakah data user akan dihapus, jika ya maka akan memanggil user_Model untuk melakukan penghapusan data pada *database*.
cari_data() : void	Prosedur untuk mencari data user pada *database* dan menampilkannya.

23. *Class* Return_Model

Class return_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data pengembalian produk. *Class* ini berhubungan langsung dengan *database* untuk proses pengembalian barang. Atribut dan operasi yang

dimiliki *class* return_model dapat dilihat pada gambar 3.43



Gambar 3.43 Deskripsi *Class* return_model

a. Deskripsi Atribut

Class return_model tidak memiliki atribut.

b. Deskripsi Operasi

Deskripsi operasi *class* return_model dapat dilihat pada tabel 3.43

Tabel 3.43 Deskripsi Operasi *Class* return_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data return produk.
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data return produk, melakukan perintah update data pada *database*.
upload_gambar() : void	Prosedur untuk menyimpan gambar yang diupload dan menyimpan alamatnya pada *database*.
ubah_status_retur() : void	Prosedur yang dipanggil ketika mengubah status retur produk pada *database*.

24. *Class* Shipping_Model

Class shipping_model merupakan *class* yang digunakan untuk melakukan proses pengiriman barang oleh admin dan memanggil *class-class* lain yang berkaitan. Atribut dan operasi yang dimiliki *class* shipping_model dapat dilihat pada gambar 3.44



Gambar 3.44 Deskripsi *Class* shipping_model

a. Deskripsi Atribut

Class shipping_model tidak memiliki atribut.

b. Deskripsi Operasi

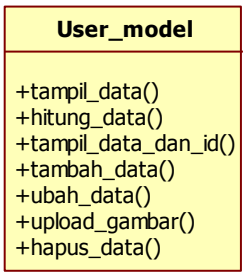
Deskripsi operasi *class* shipping_model dapat dilihat pada tabel 3.44

Tabel 3.44 Deskripsi Operasi *Class* shipping_model

Nama	Keterangan
tampil_data() : void	Prosedur yang dipanggil ketika user meminta untuk menampilkan data, prosedur ini akan memberi perintah kepada *class* return _Model untuk mengambil data pada *database*.
get_harga() : void	Prosedur untuk mengambil data harga
get_alamat() : void	Prosedur untuk mengambil data alamat
get_berat() : void	Prosedur untuk mengambil data berat

25. *Class* User_Model

Class user_model merupakan *class* yang berisi prosedur-prosedur untuk mengolah data user. *Class* ini berhubungan langsung dengan *database* untuk proses mengambil, menyimpan dan menghapus data. Atribut dan operasi yang dimiliki *class* user_model dapat dilihat pada gambar 3.45



Gambar 3.45 Deskripsi *Class* user_model

a. Deskripsi Atribut

Class user _model tidak memiliki atribut.

b. Deskripsi Operasi

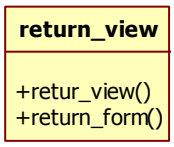
Deskripsi operasi *class* user _model dapat dilihat pada tabel 3.45

Tabel 3.45 Deskripsi Operasi *Class* user_model

Nama	Keterangan
tampil_data() : void	Prosedur mengambil data dari *database*, digunakan untuk menampilkan seluruh data user.
hitung_data() : void	Prosedur menghitung jumlah data pada *database*
tampil_data_dari_Id() : void	Prosedur untuk mengambil data dari *database* berdasarkan id user.
tambah_data() : void	Prosedur yang dipanggil ketika proses tambah data user, melakukan perintah simpan data pada *database*.
ubah_data() : void	Prosedur yang dipanggil ketika proses ubah data user, melakukan perintah update data pada *database*.
Upload_gambar() : void	Prosedur untuk menyimpan gambar yang diupload dan menyimpan alamatnya pada *database*.
hapus_data() : void	Prosedur yang dipanggil ketika hapus data, menghapus data user pada *database*.

26. *Class* Return_View

Class return_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* return_view dapat dilihat pada gambar 3.46



Gambar 3.46 Deskripsi *Class* return_view

a. Deskripsi Atribut

Class return_view tidak memiliki atribut.

b. Deskripsi Operasi

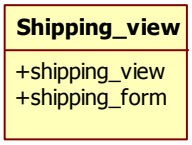
Deskripsi operasi *class* return_view dapat dilihat pada tabel 3.46

Tabel 3.46 Deskripsi Operasi *Class* return_view

Nama	Keterangan
return_view() : void	Prosedur untuk menampilkan data return yang hasil prosedur dari *class* return_model dan return_control.
return_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* return_control.

27. *Class* Shipping_View

Class shipping_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* shipping_view dapat dilihat pada gambar 3.47



Gambar 3.47 Deskripsi *Class* shipping_view

a. Deskripsi Atribut

Class shipping_view tidak memiliki atribut.

b. Deskripsi Operasi

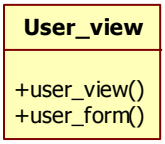
Deskripsi operasi *class* shipping_view dapat dilihat pada tabel 3.47

Tabel 3.47 Deskripsi Operasi *Class* shipping_view

Nama	Keterangan
shipping_view() : void	Prosedur untuk menampilkan data return yang hasil prosedur dari *class* shipping _model dan shipping _control.
shipping_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* shipping_control.

28. *Class* User_View

Class user_view merupakan *class* yang berisi prosedur-prosedur untuk mengatur tampilan yang berinteraksi langsung dengan user. Atribut dan operasi yang dimiliki *class* user_view dapat dilihat pada gambar 3.48



Gambar 3.48 Deskripsi *Class* user_view

a. Deskripsi Atribut

Class user_view tidak memiliki atribut.

b. Deskripsi Operasi

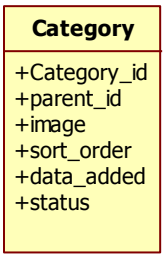
Deskripsi operasi *class* user_view dapat dilihat pada tabel 3.48

Tabel 3.48 Deskripsi Operasi *Class* user_view

Nama	Keterangan
user_view() : void	Prosedur untuk menampilkan data return yang hasil prosedur dari *class* user _model dan user _control.
user_form() : void	Prosedur untuk menampilkan form ketika memanggil prosedur tambahdata dan ubahdata pada *class* user _control.

29. *Class* Category

Class Category merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* Category dapat dilihat pada gambar 3.49.



Gambar 3.49 Deskripsi *Class* category

30. *Class* Product

Class Product merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* Product dapat dilihat pada gambar 3.50.

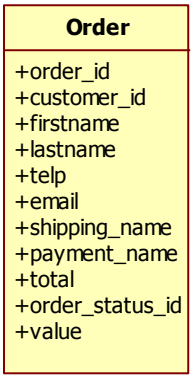


Gambar 3.50 Deskripsi *Class* product

31. *Class* Order

Class order merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang

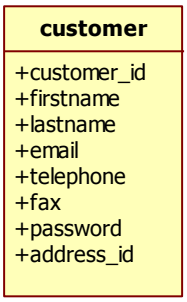
dimiliki *class* order dapat dilihat pada gambar 3.51



Gambar 3.51 Deskripsi *Class* order

32. *Class* Customer

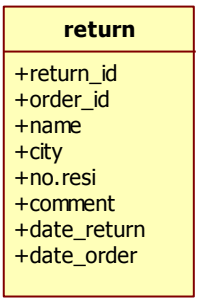
Class customer merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* customer dapat dilihat pada gambar 3.52



Gambar 3.52 Deskripsi *Class* customer

33. *Class* Return

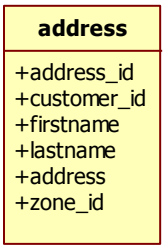
Class return merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* return dapat dilihat pada gambar 3.53



Gambar 3.53 Deskripsi *Class* return

34. *Class* Address

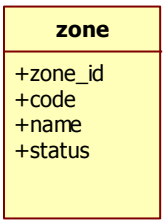
Class address merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* address dapat dilihat pada gambar 3.54



Gambar 3.54 Deskripsi *Class* address

35. *Class* Zone

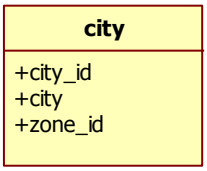
Class zone merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* zone dapat dilihat pada gambar 3.55



Gambar 3.55 Deskripsi *Class* zone

36. *Class* City

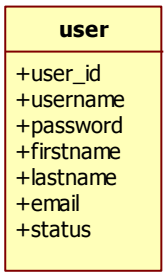
Class city merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* city dapat dilihat pada gambar 3.56



Gambar 3.56 Deskripsi *Class* city

37. *Class*

Class user merupakan entitas yang berisi atribut-atribut yang digunakan dalam proses pengolahan data yang akan dilakukan. Atribut dan operasi yang dimiliki *class* user dapat dilihat pada gambar 3.57

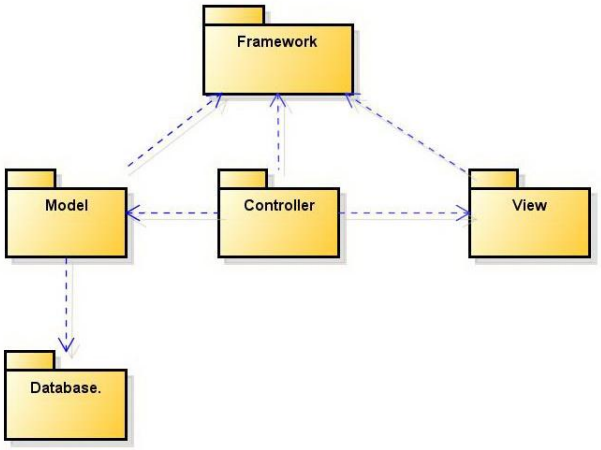


Gambar 3.57 Deskripsi *Class* user

III.3.3.9 *Package Diagram*

Package diagram adalah sebuah bentuk pengelompokkan yang memungkinkan untuk mengambil setiap bentuk di UML dan mengelompokkan elemen-elemennya dalam tingkatan unit yang lebih tinggi.

Package diagram untuk pengelompokkan *class* pada *E-commerce* Arsefa Collection dapat dilihat pada gambar 3.58.



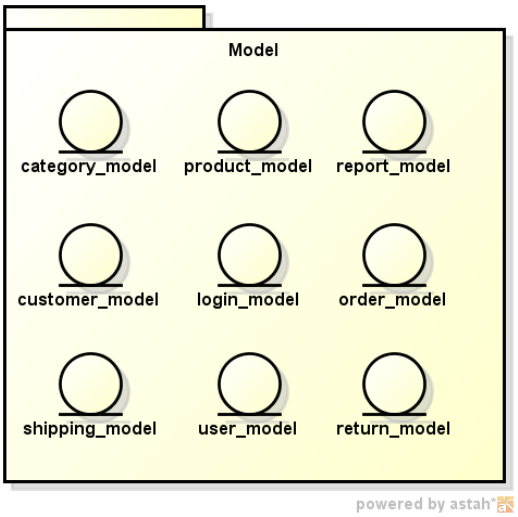
Gambar 3.58 *Package* Diagram *E-commerce* Arsefa Collection

Deskripsi untuk *package* diagram adalah sebagai berikut :

1. ***Package Model***

   Package Model berisi *class-class* model yang mengakses langsung

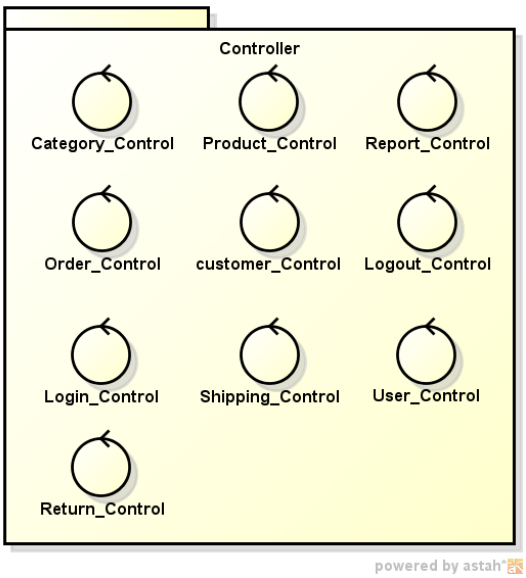
pada *database* untuk dapat mengolah data. Isi dari *package* Models dapat dilihat pada gambar 3.59



Gambar 3.59 *Package* Model

2. *Package Controller*

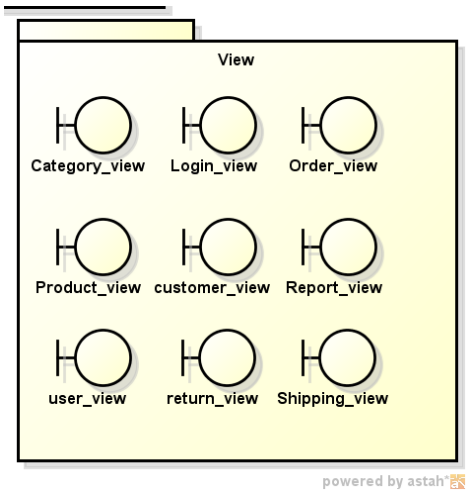
Package Controller berisi *class-class control* yang merupakan pengatur dalam setiap proses yang ada. Isi dari *package Controller* dapat dilihat pada gambar 3.60



Gambar 3.60 *Package* Controller

3. Package View

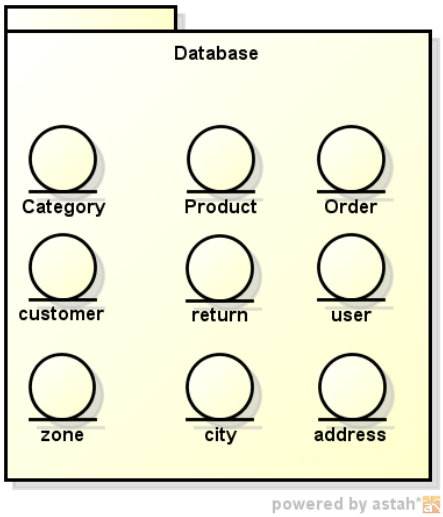
Package View berisi *class-class* view yang digunakan untuk menampilkan interface yang dapat berupa tampilan tabel atau form. Isi dari *Package* View dapat dilihat pada gambar 3.61



Gambar 3.61 *Package* View

4. Package Database

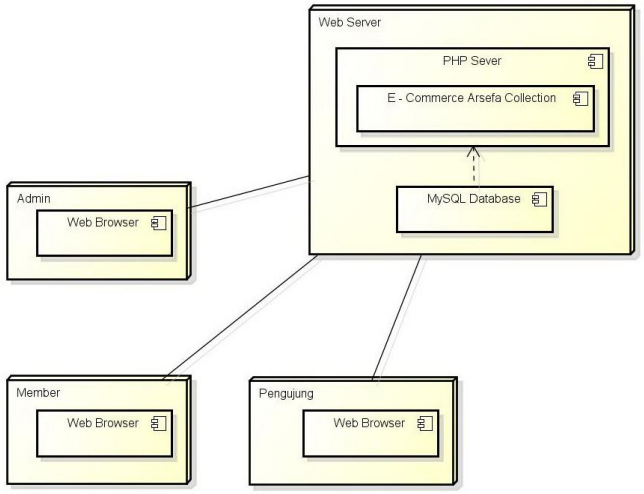
Package Database berisi *class-class* entity yang berisi atribut-atribut yang akan digunakan pada proses pengolahan data dan penyimpanan pada *database*. Isi dari *Package* Database dapat dilihat pada gambar 3.62



Gambar 3.62 *Package* Database

III. 3. 3. 10. *Deployment* Diagram

Deployment diagram menunjukkan susunan fisik sebuah sistem, menunjukkan bagian perangkat lunak mana yang berjalan pada perangkat keras mana. *Deployment* diagram *E-commerce* pada Arsefa Collection dapat dilihat pada gambar 3.63



Gambar 3.63 *Deployment* diagram

III. 4. Perancangan Sistem

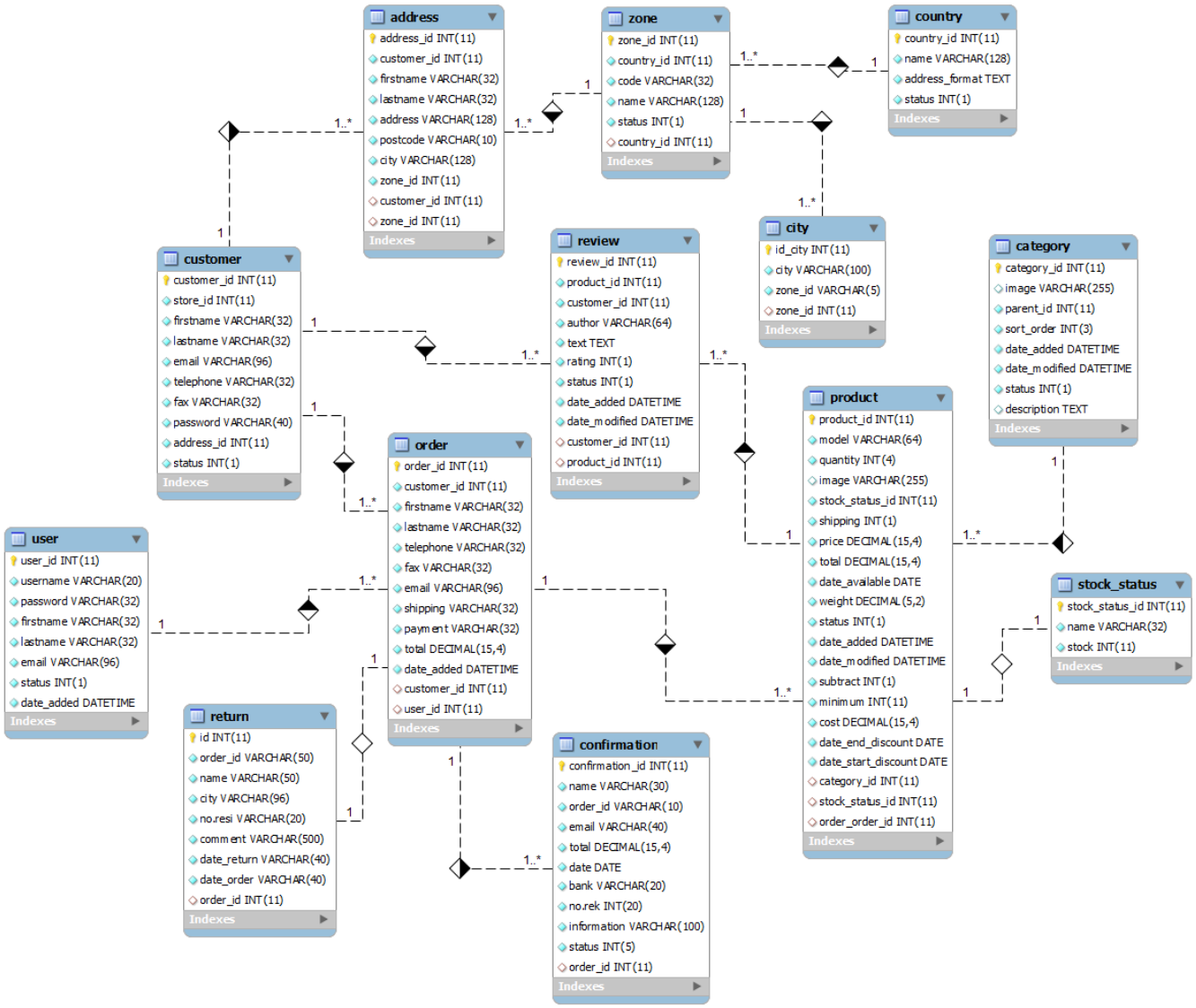
Perancangan merupakan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam suatu kesatuan yang utuh dan berfungsi. Tahapan ini meliputi mengkonfigurasi komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem sehingga setelah instalasi sistem akan mendapatkan hasil yang baik berdasarkan rancang bangun yang telah ditetapkan pada tahap akhir analisis sistem.

III. 4. 1. Perancangan Data

Perancangan data merupakan tahapan untuk memetakan model konseptual ke model basis data yang akan dipakai.

III. 4. 1. 1. Diagram Relasi

Diagram relasi merupakan rangkaian hubungan antara dua tabel atau lebih pada sistem basis data. Penjelasan rangkaian basis data aplikasi *e-commerce* pada Arsefa Collection dapat dilihat pada gambar 3.64



Gambar 3.64 Diagram relasi *e-commerce* pada Arsefa Collection

III. 4. 1. 2. Struktur Tabel

Struktur tabel adalah spesifikasi *file* dari tiap tabel yang digunakan dalam sistem yang akan dibangun. Struktur tabel dari sistem yang dibangun adalah sebagai berikut:

1. Customer

Nama table : customer

Primary Key : customer_id

Tabel 3.49 Struktur Tabel customer

Nama Field	Tipe	Ukuran	Keterangan
customer_id	INT	11	Primary Key
firstname	Varchar	32	
lastname	Varchar	32	
email	Varchar	96	
telepone	Varchar	32	
fax	Varchar	32	
password	Varchar	40	
address_id	INT	11	
status	INT	1	
approved	INT	1	

2. zone

Nama table : zone

Primary Key : zone_id

Tabel 3.50 Struktur Tabel zone

Nama Field	Tipe	Ukuran	Keterangan
zone_id	INT	11	Primary Key
country_id	INT	11	
code	Varchar	255	
name	Varchar	255	
status	INT	1	

3. stock_status

Nama table : stock_status

Primary Key : stock_status _id

Tabel 3.51 Struktur Tabel stock_status

Nama Field	Tipe	Ukuran	Keterangan
stock_status _id	INT	11	Primary Key
name	Varchar	32	

4. city

Nama table : city

Primary Key : id_city

Tabel 3.52 Struktur Tabel city

Nama Field	Tipe	Ukuran	Keterangan
id_city	INT	11	Primary Key
city	Varchar	100	
zone_id	Varchar	5	

5. country

Nama table : country

Primary Key : country_id

Tabel 3.53 Struktur Tabel country

Nama Field	Tipe	Ukuran	Keterangan
country_id	INT	11	Primary Key
name	Varchar	128	
iso_code_2	Varchar	5	
iso_code_3	Varchar	3	
address_format	Text		
postcode_required	INT	1	
status	INT	1	

6. user

Nama table : user

Primary Key : user_id

Tabel 3.54 Struktur Tabel user

Nama Field	Tipe	Ukuran	Keterangan
user_id	INT	11	Primary Key
username	Varchar	20	
password	Varchar	32	
firstname	Varchar	32	
lastname	Varchar	32	
email	Varchar	96	
status	INT	1	
ip	Varchar	15	
date_added	DATETIME		

7. address

Nama table : address

Primary Key : address_id

Tabel 3.55 Struktur Tabel address

Nama Field	Tipe	Ukuran	Keterangan
address_id	INT	11	Primary Key
customer_id	Varchar	12	
firstname	Varchar	32	
lastname	Varchar	32	
address_1	Varchar	128	
address_2	Varchar	128	
postcode	Varchar	10	
city	Varchar	128	
zone_id	INT	11	

8. product

Nama tabel : product

Primary Key : product_id

Tabel 3.56 Struktur Tabel product

Nama Field	Tipe	Ukuran	Keterangan
product_id	INT	11	Primary Key
model	Varchar	64	
sku	Varchar	64	
quantity	INT	4	
stock_status_id	INT	11	
image	Varchar	255	
shipping	INT	1	
price	Decimal	15,4	
date_available	Date		
weight	Decimal	5,2	
status	INT	1	
date_added	Datetime		
date_modified	Datetime		
viewed	INT	5	
sort_order	INT	11	
subtract	INT	1	
minimum	INT	1	
cost	Decimal	15,4	
description	Text		

9. Category

Nama table : category

Primary Key : category_id

Tabel 3.57 Struktur Tabel category

Nama Field	Tipe	Ukuran	Keterangan
category_id	INT	11	Primary Key
product_id	Int	11	
image	Varchar	255	
parent_id	INT	11	
sort_order	INT	3	
date_added	Datetime		
date_modified	Datetime		
status	INT	1	
description	Text		

10. Return

Nama table : return

Primary Key : return_id

Tabel 3.58 Struktur Tabel return

Nama Field	Tipe	Ukuran	Keterangan
return_id	INT	11	Primary Key
order_id	INT	11	
date_ordered	Date		
costumer_id	INT	11	
firstname	Varchar	32	
lastname	Varchar	32	
email	Varchar	96	
telephone	Varchar	32	
status	INT	11	
comment	Text		
date_added	Datetime		
date_modified	Datetime		
reason	INT	11	

11. confirmation

Nama table : confirmation

Primary Key : confirmation_id

Tabel 3.59 Struktur Tabel tbl_konfirmasi

Nama Field	Tipe	Ukuran	Keterangan
confirmation_id	INT	11	Primary Key
name	Varchar	30	
Order_id	Varchar	10	
Email	Varchar	40	
total	Decimal	15,4	
date	Date		
bank	Varchar	20	
no.rek	INT	20	
information	Varchar	100	
status	INT	5	

12. order

Nama table : order

Primary Key : order_id

Tabel 3.60 Struktur Tabel order

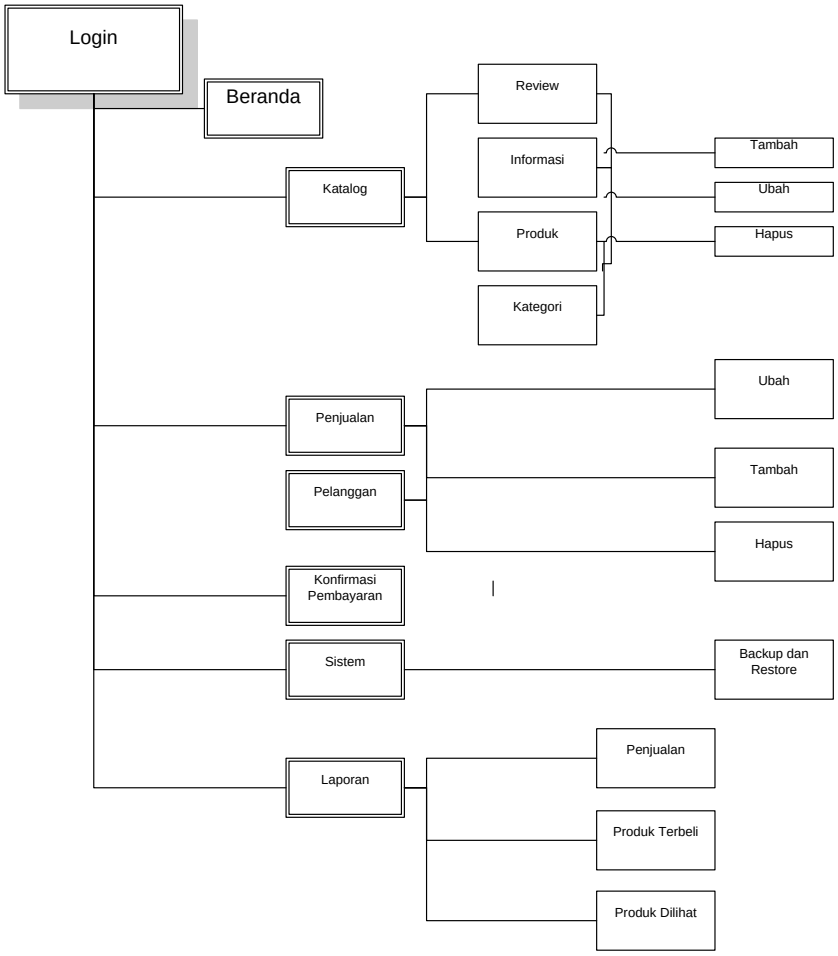
Nama Field	Tipe	Ukuran	Keterangan
Order_id	INT	11	Primary Key
Costumer_id	INT	11	
Firstname	Varchar	32	
Lastname	Varchar	32	
Telephone	Varchar	32	
Fax	Varchar	32	
Email	Varchar	96	
Comment	Text		
Total	Decimal	15,4	
Order_status_id	INT	11	
Value	Decimal	15,8	
Date_modified	Datetime		
Date_added	Datetime		

III. 4. 2. Arsitektur Menu

Menu dirancang untuk menggambarkan segala macam bentuk kegiatan yang bisa dilakukan dalam program aplikasi, seperti *input* data, cari data dan membuat report berdasarkan data *input*. Struktur menu yang dirancang sesuai dengan level pengguna sistem. Terdapat 3 pengguna aplikasi *e-commerce* pada Arsefa Collection yaitu admin, account dan pengunjung. Adapun struktur menu dati setiap level pengguna adalah sebagai berikut :

1. Perancangan struktur menu *admin*

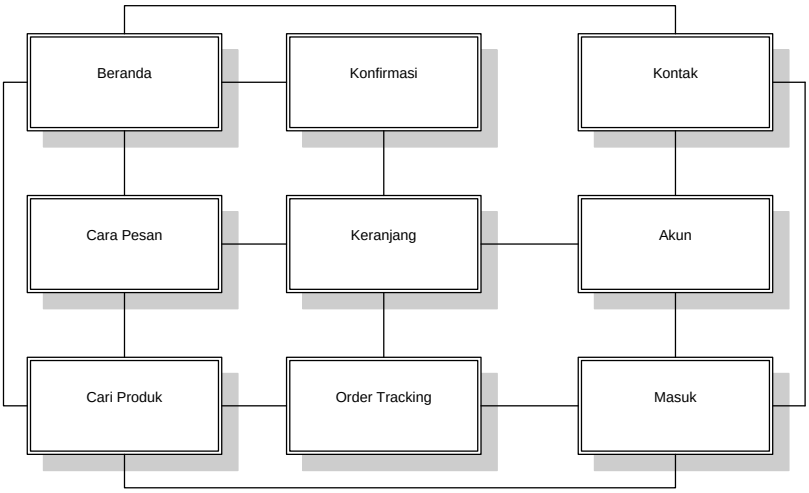
Perancangan struktur menu admin dapat dilihat pada gambar 3.65



Gambar 3.65 struktur menu *admin*

2. Perancangan Struktur Menu Pengunjung

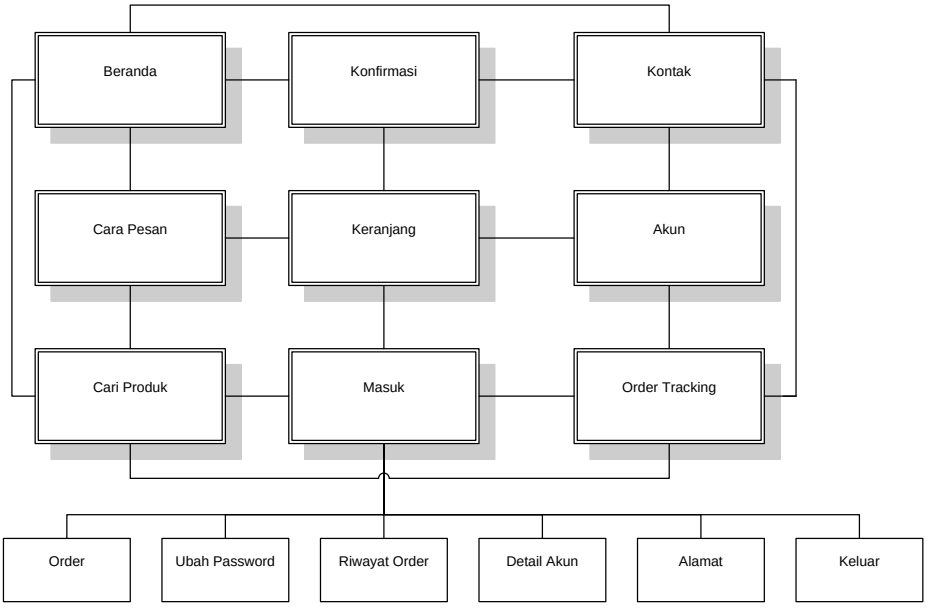
Perancangan struktur menu Pengunjung dapat dilihat pada gambar 3.66



Gambar 3.66 struktur menu pengunjung

3. Perancangan Struktur Menu Account

Perancangan struktur menu Pengunjung dapat dilihat pada gambar 3.67



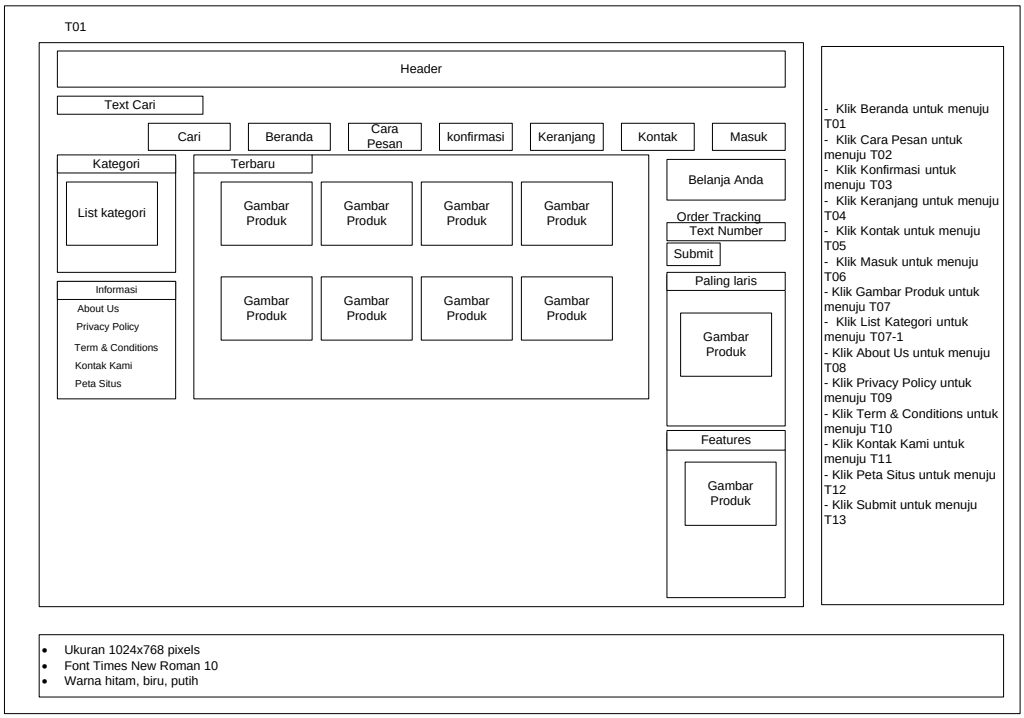
Gambar 3.67 struktur menu Account

III. 4. 3. Perancangan Antarmuka

Interface atau antar muka merupakan tampilan dari suau program aplikasi yang berperan sebagai media komunikasi yang digunakan sebagai sarana berdialog antara program dengan *user*. Sistem yang akan dibangun diharapkan menyediakan *interface* yang mudah dipahami dan digunakan oleh *user*.

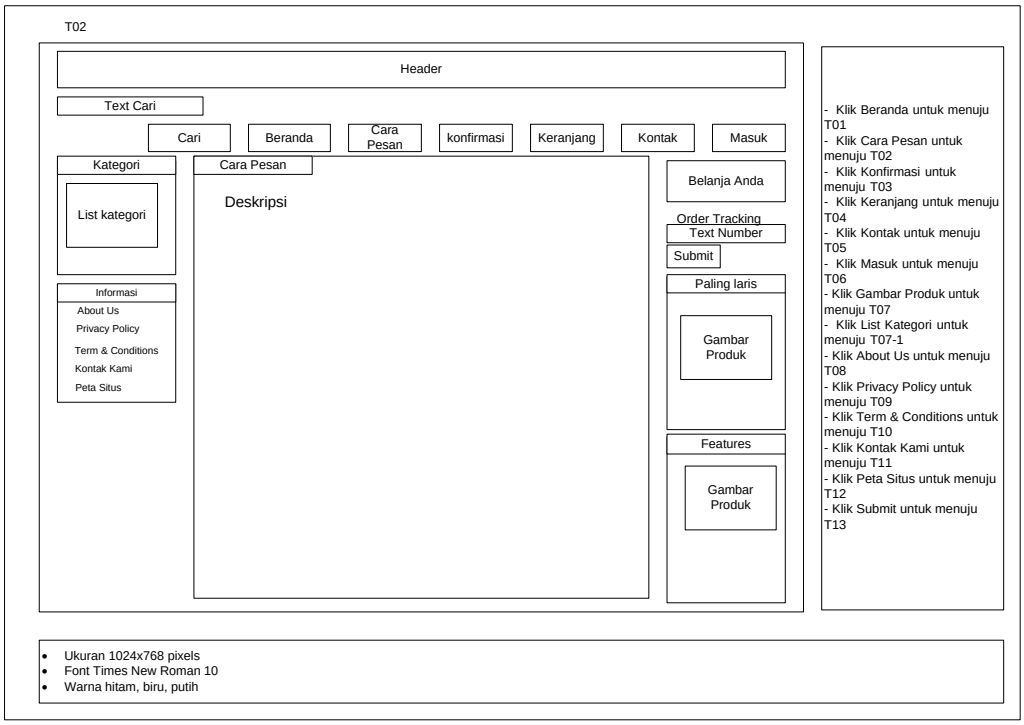
Untuk memudahkan proses pembuatan aplikasi maka terlebih dahulu membuat rancangan desain tampilannya. Perancangan ini meliputi rancanagan sebagai berikut :

1. Perancangan Halaman Beranda (T01)



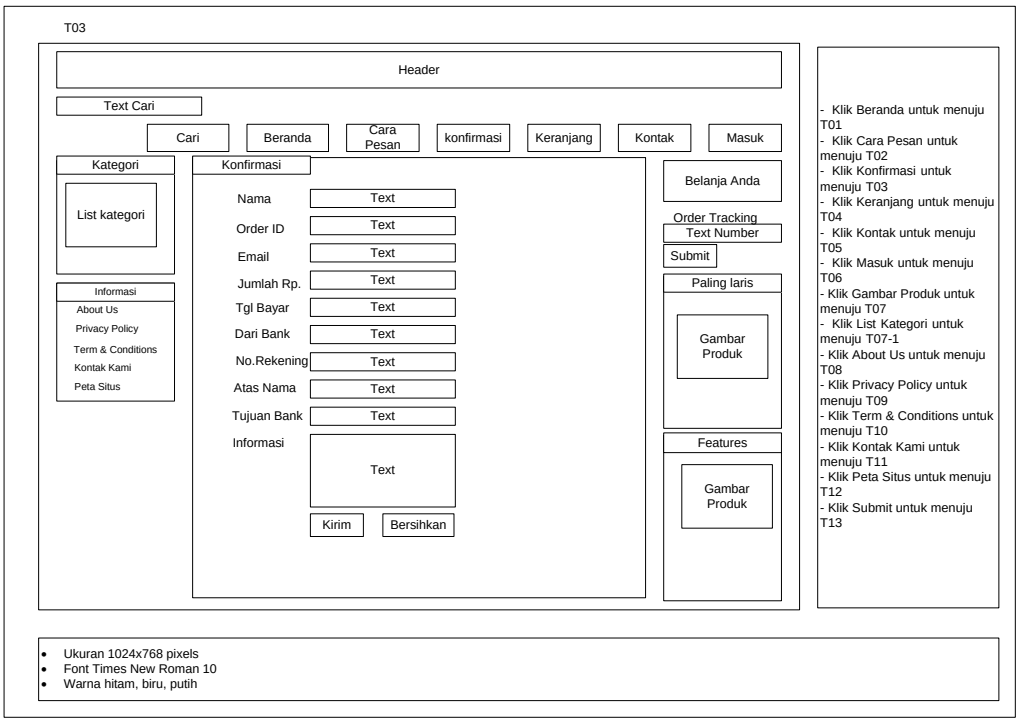
Gambar 3.68 Perancangan Tampilan Halaman Beranda

2. Perancangan Halaman Cara Pesan (T02)



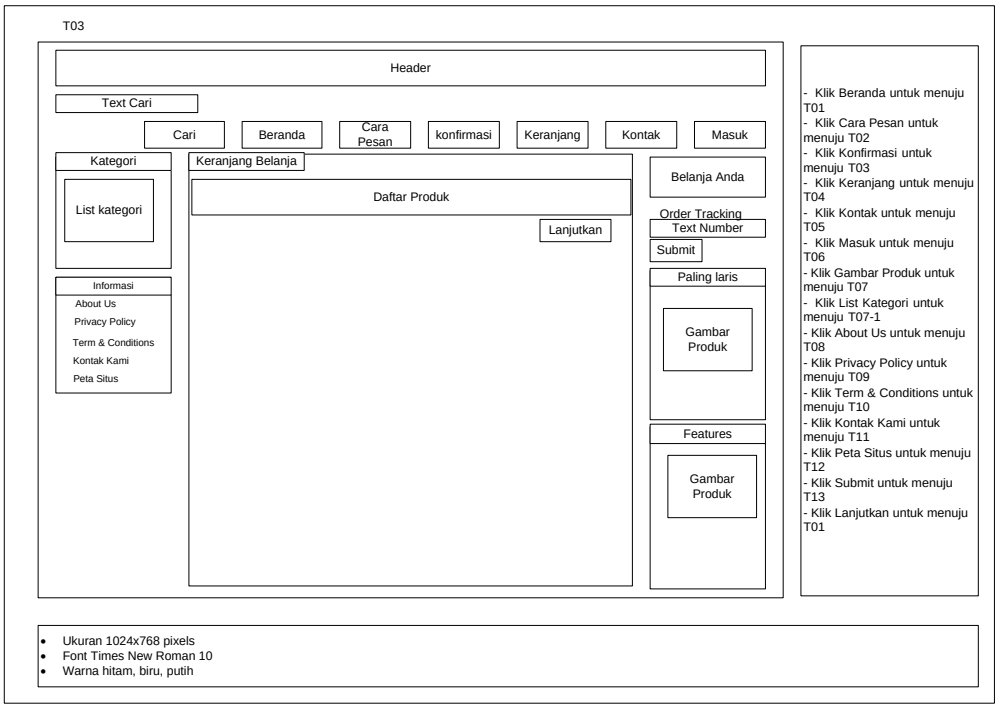
Gambar 3.69 Perancangan Tampilan Halaman Cara Pesan

3. Perancangan Halaman Konfirmasi (T03)



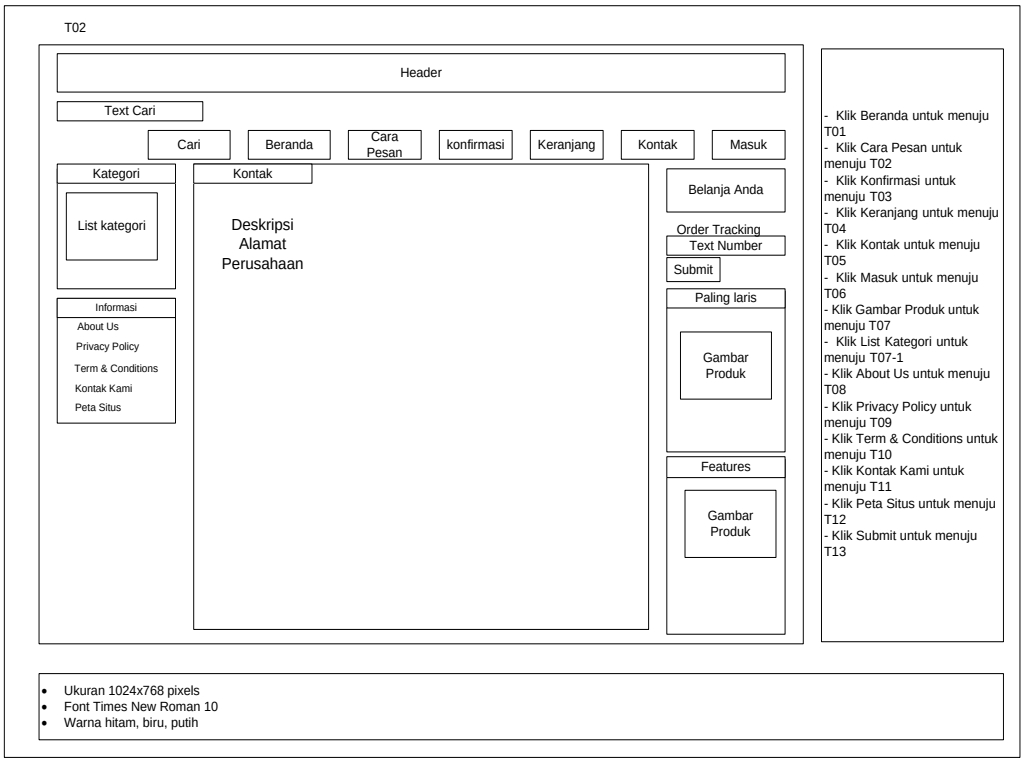
Gambar 3.70 Perancangan Tampilan Halaman Konfirmasi

4. Perancangan Halaman Keranjang (T04)



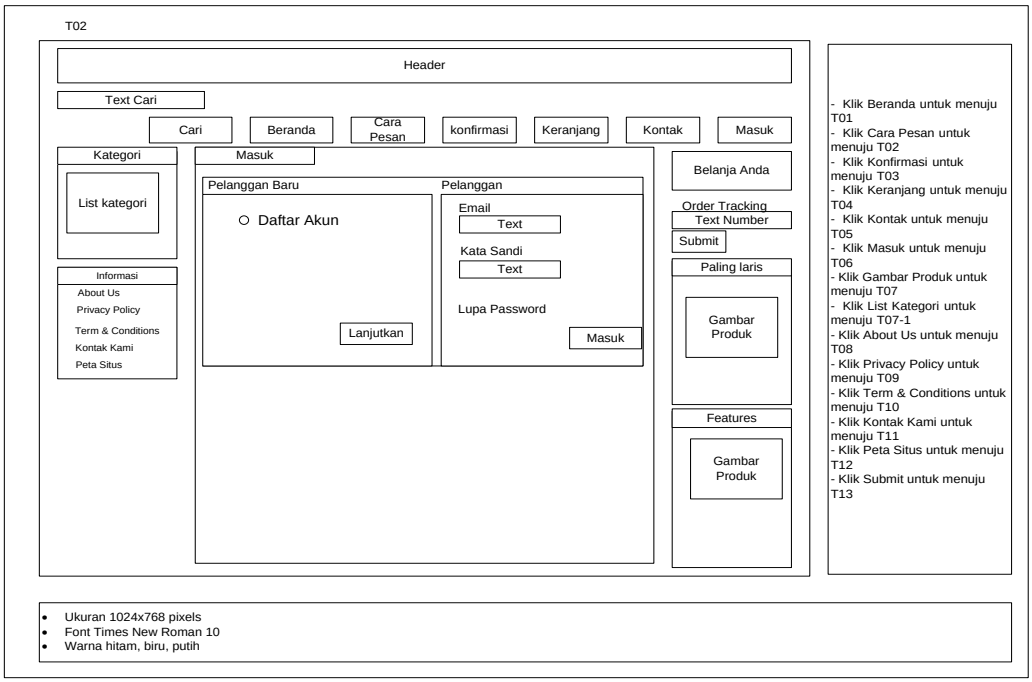
Gambar 3.71 Perancangan Tampilan Halaman Keranjang

5. Perancangan Halaman Kontak (T05)



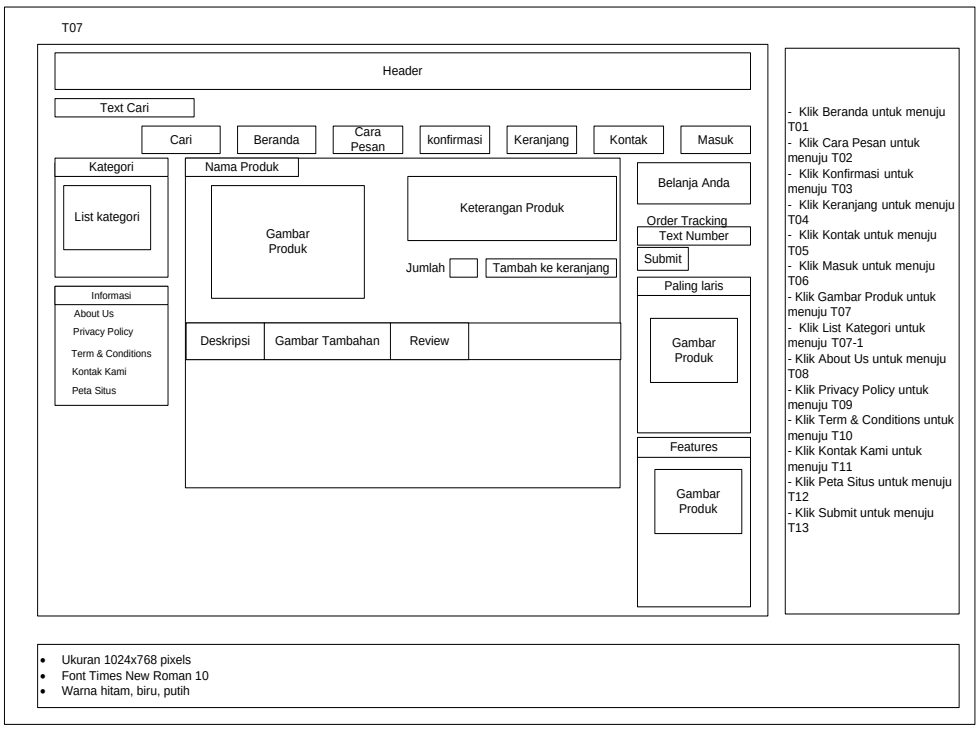
Gambar 3.72 Perancangan Tampilan Halaman Kontak

6. Perancangan Halaman Masuk (T06)



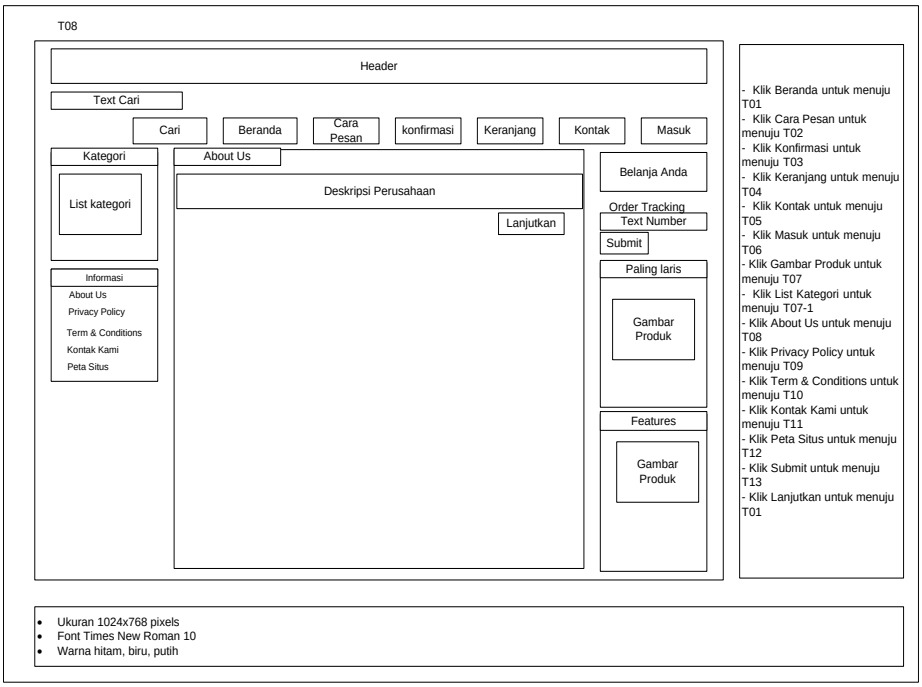
Gambar 3.73 Perancangan Tampilan Halaman Masuk

7. Perancangan Halaman Gambar Product (T07)



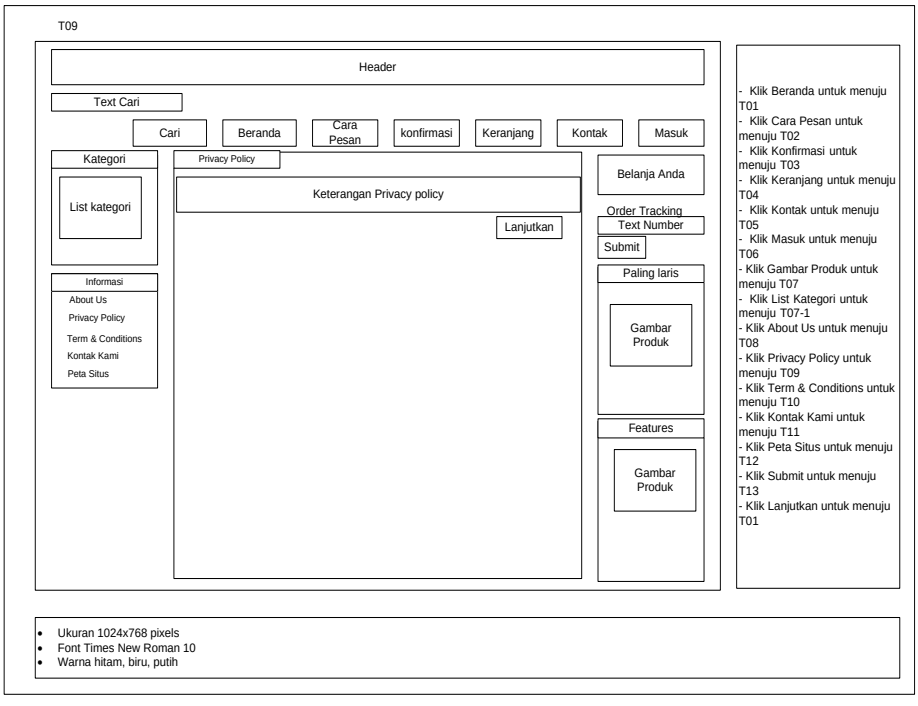
Gambar 3.74 Perancangan Tampilan Halaman Gambar Product

8. Perancangan halaman About Us (T08)



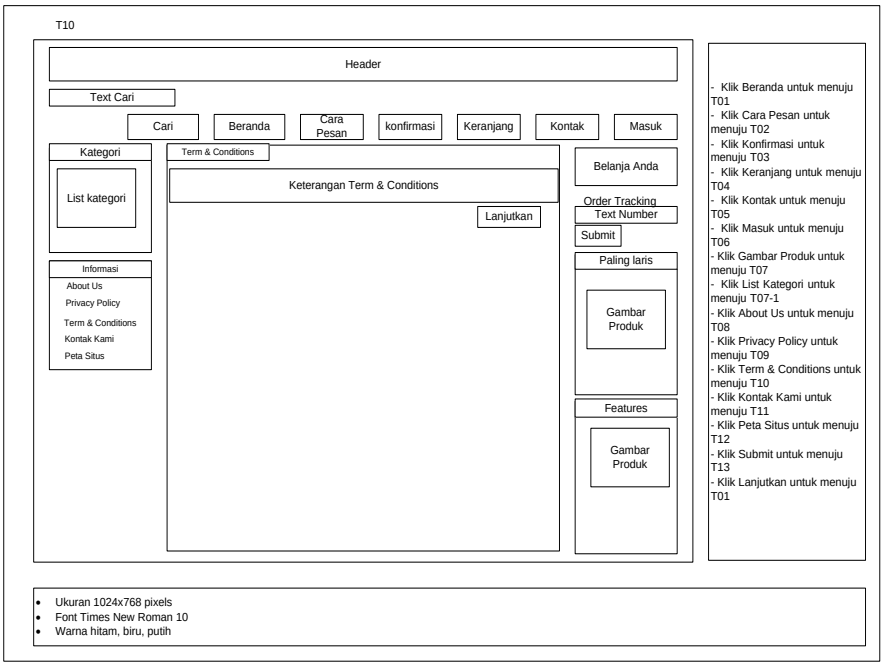
Gambar 3.75 Perancangan Tampilan Halaman About Us

9. Perancangan Halaman Privacy Policy (T09)



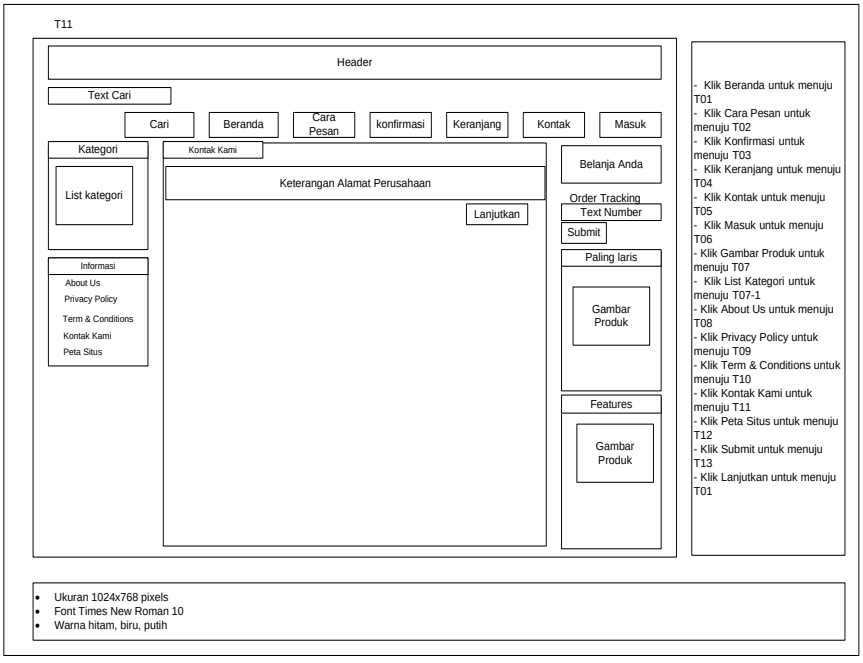
Gambar 3.76 Perancangan Tampilan Halaman Privacy Policy

10. Perancangan Halaman Term & Conditions (T10)



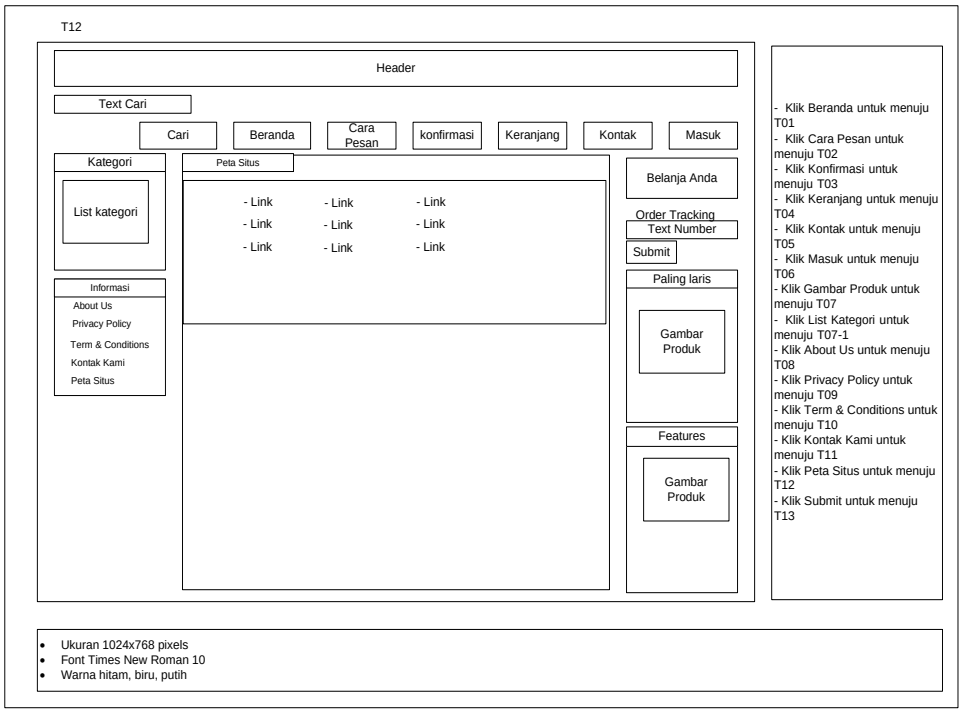
Gambar 3.77 Perancangan Tampilan Halaman Term & Conditions

11. Perancangan Halaman Kontak Kami (T11)



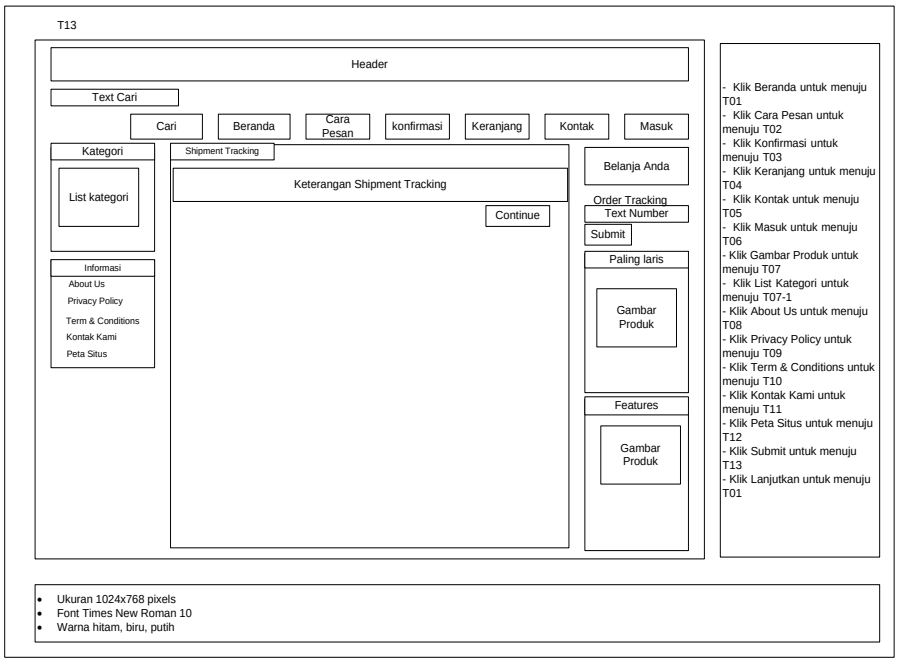
Gambar 3.78 Perancangan Tampilan Halaman Kontak Kami

12. Perancangan Halaman Peta Situs (T12)



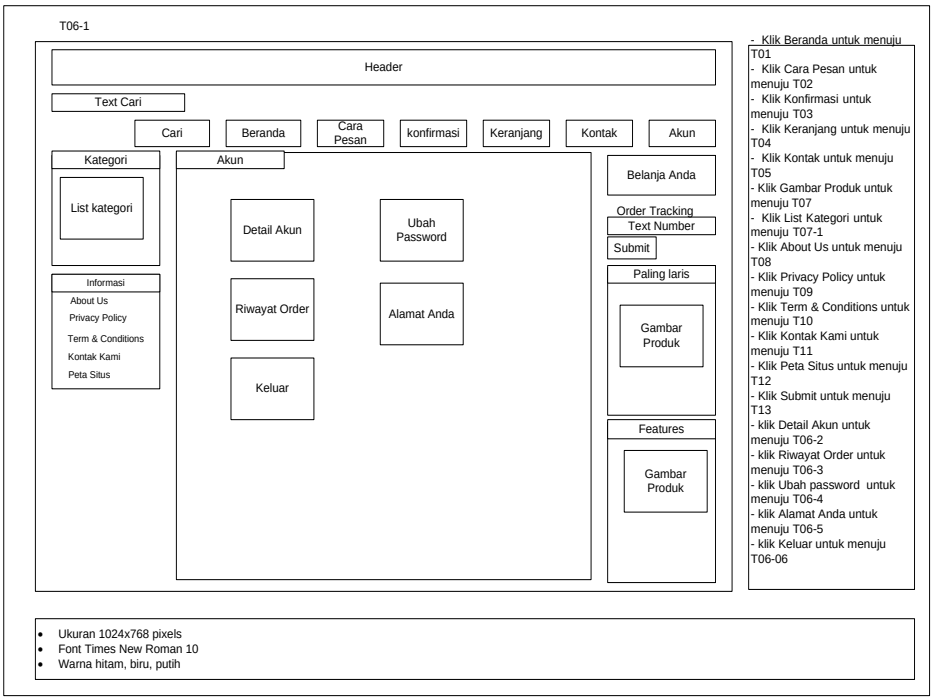
Gambar 3.79 Perancangan Tampilan Halaman Peta Situs

13. Perancangan Halaman Shipment Tracking (T13)



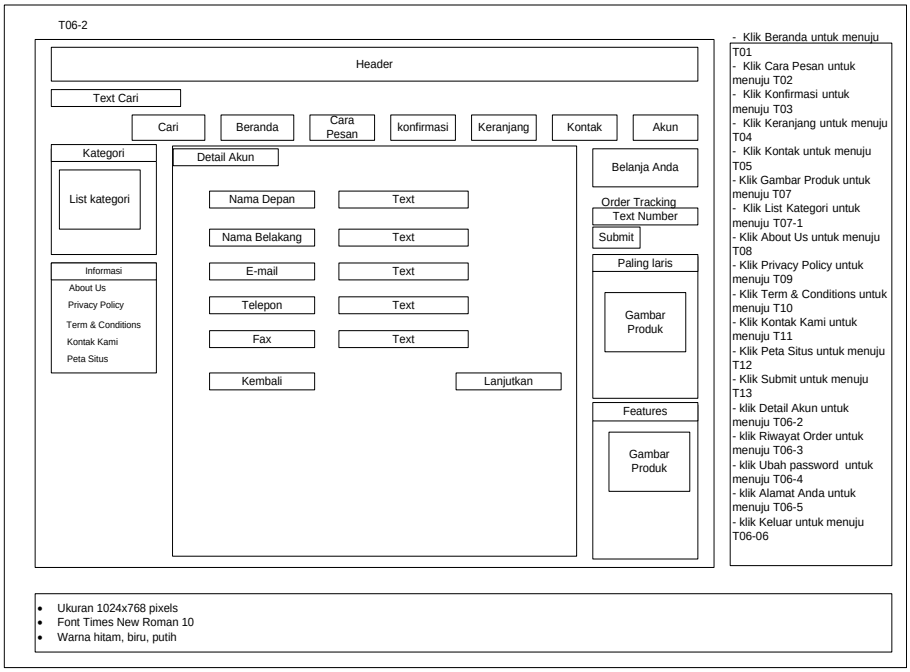
Gambar 3.80 Perancangan Tampilan Halaman Shipment Tracking

14. Perancangan Halaman Akun (T06-1)



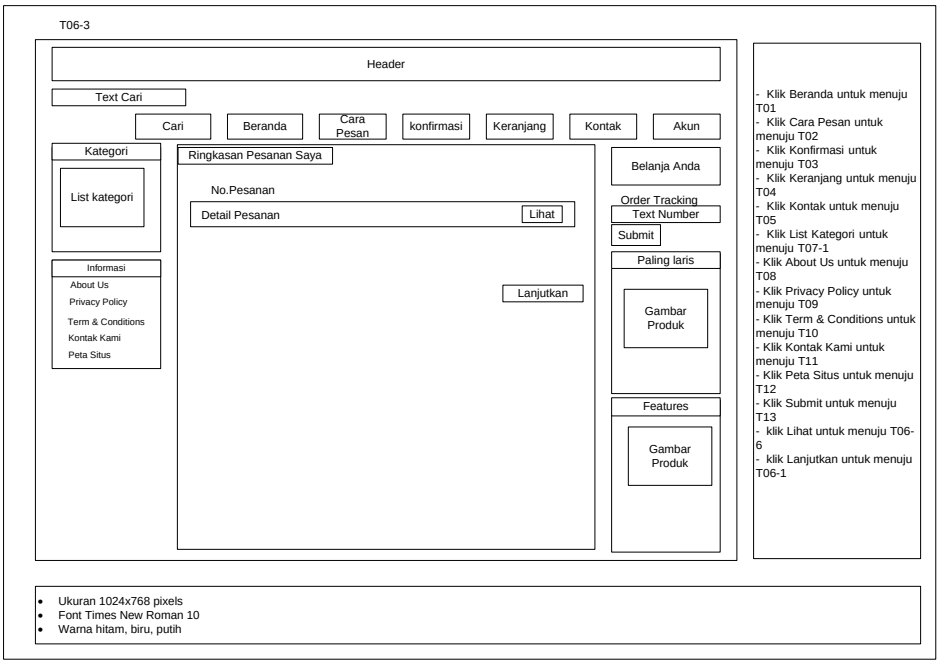
Gambar 3.81 Perancangan Tampilan Halaman Akun

15. Perancangan Halaman Detail Akun (T06-2)



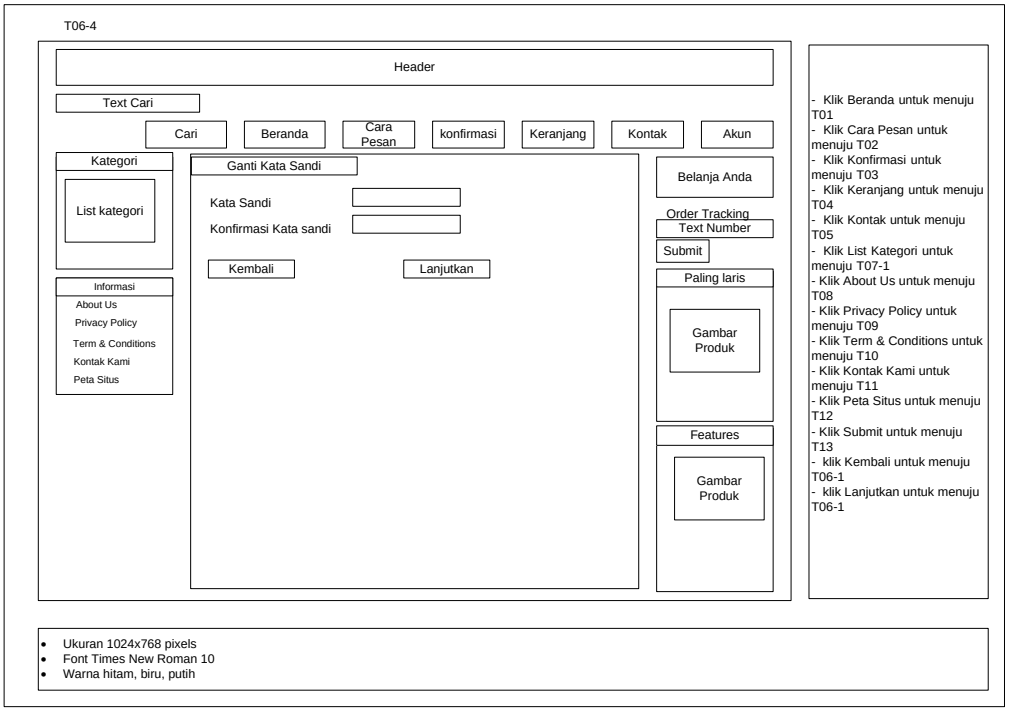
Gambar 3.82 Perancangan Tampilan Halaman Detail Akun

16. Perancangan Halaman Riwayat Order (T06-3)



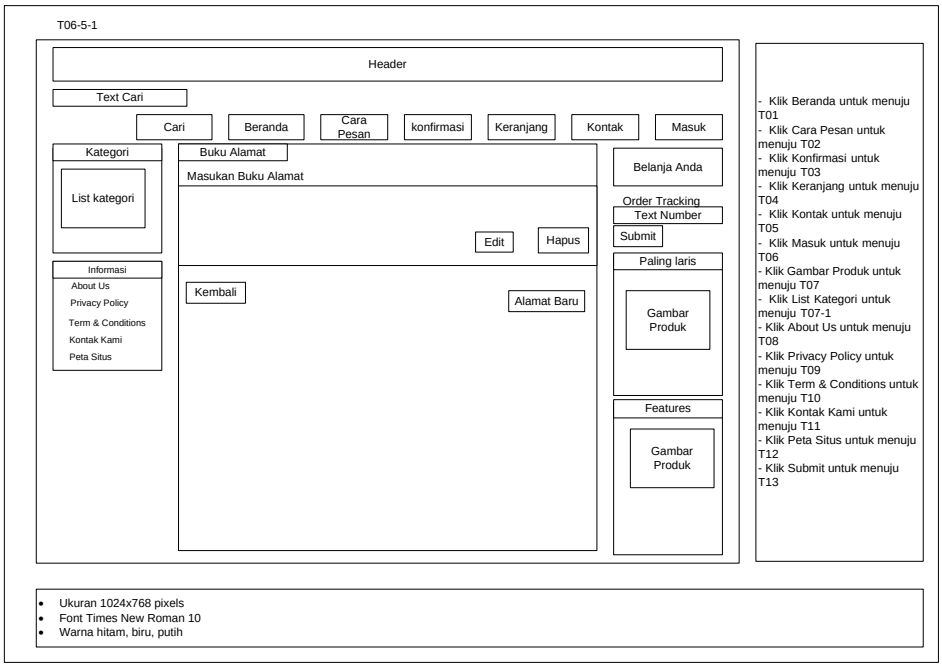
Gambar 3.83 Perancangan Tampilan Halaman Riwayat Order

17. Perancangan Halaman Ubah Password (T06-4)



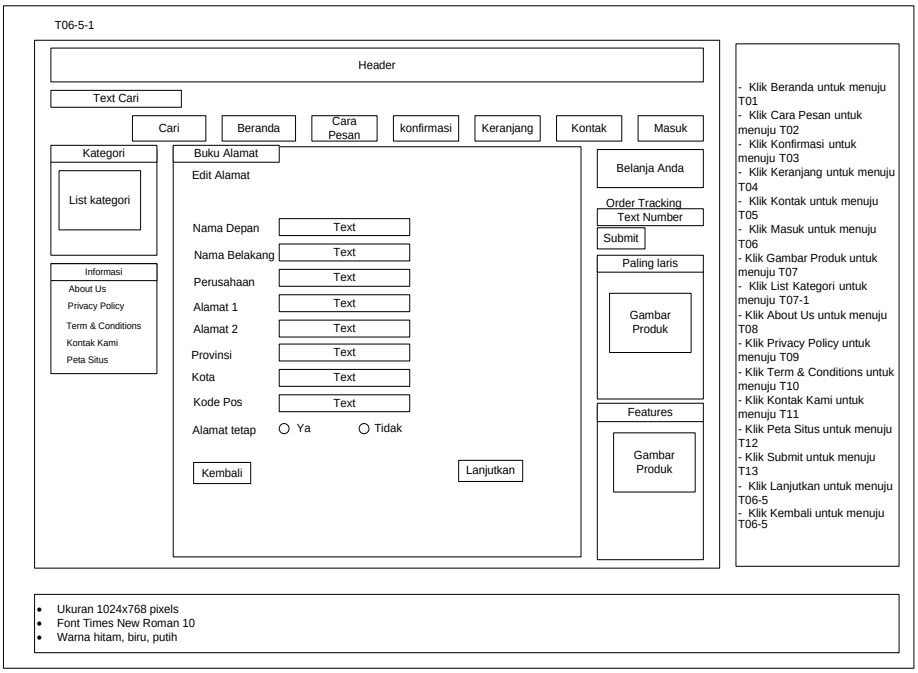
Gambar 3.84 Perancangan Tampilan Halaman Ubah Password

18. Perancangan Halaman Alamat Anda (T06-5)



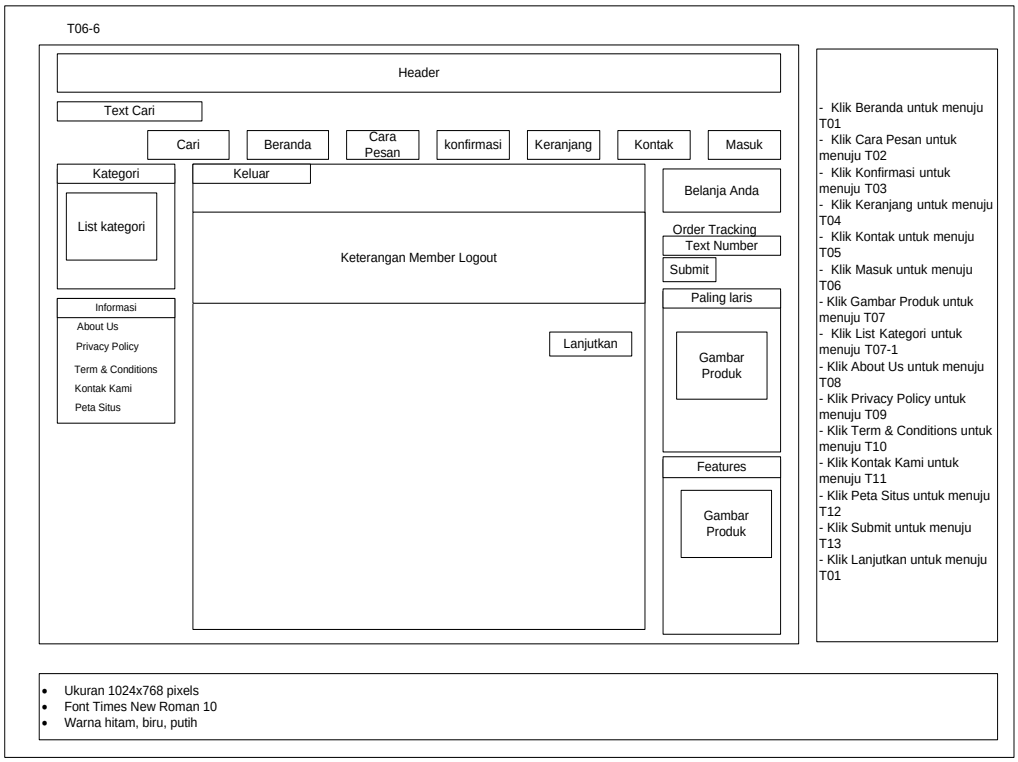
Gambar 3.85 Perancangan Tampilan Halaman Alamat Anda

19. Perancangan Halaman Edit Alamat (T06-5-1)



Gambar 3.86 Perancangan Tampilan Halaman Edit Alamat

20. Perancangan Halaman Logout (T06-6)



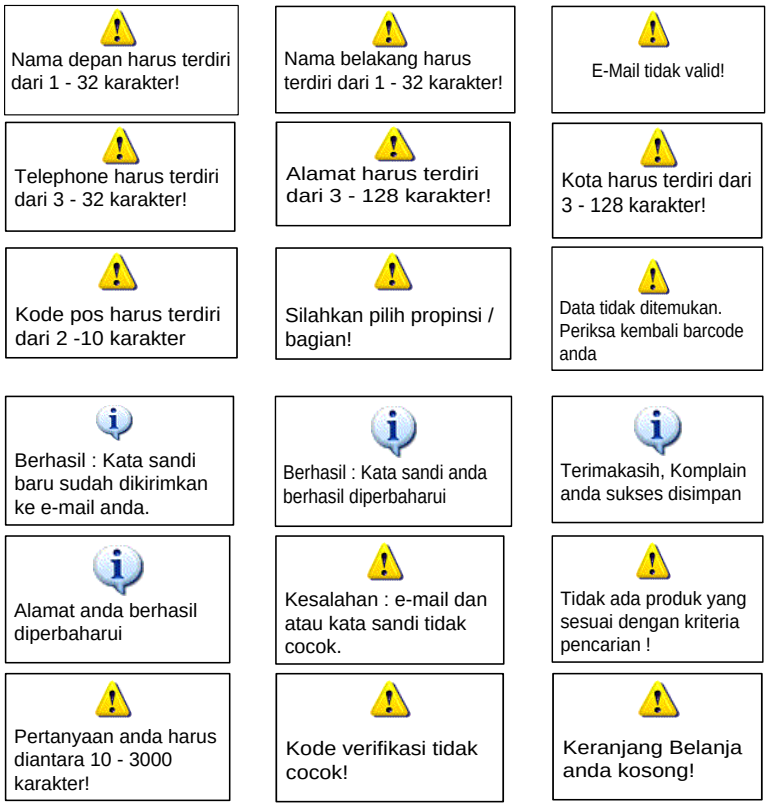
Gambar 3.87 Perancangan Tampilan Halaman Logout

III. 4. 4. Perancangan Pesan

Perancangan pesan merupakan bahasa yang disampaikan oleh sistem kepada pengguna untuk menyampaikan informasi sesuai yang dilakukan oleh *pengguna.* Tampilan pesan yang ada pada aplikasi *e-commerce* padaArsefa Collection adalah sebagai berikut :

1. Perancangan Pesan Halaman *Member*

Perancangan pesan pada halaman *member* dapat dilihat pada gambar pada gambar 3.88



Gambar 3.88 Perancangan Pesan Halaman Member

2. Perancangan Pesan Halaman Admin

Perancangan pesan pada halaman *Admin* dapat dilihat pada gambar pada gambar 3.89



Gambar 3.89 Perancangan pesan pada halaman *Admin*

III. 4. 5. Perancangan Prosedural

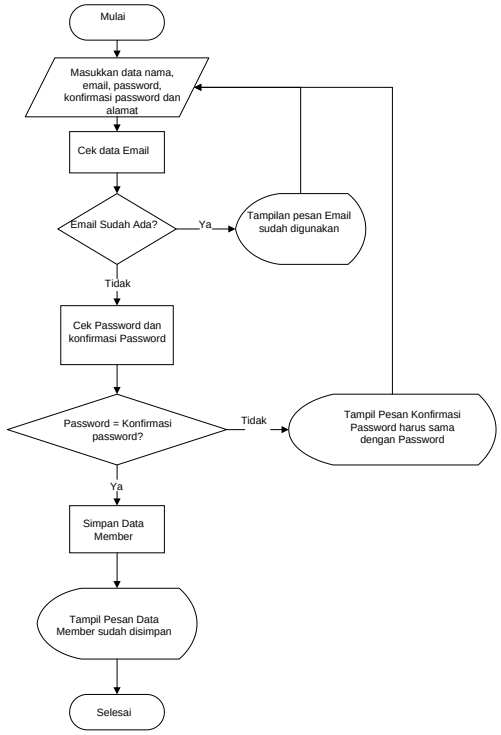
Perancangan prosedural adalah tata cara atau urutan langkah-langkah untuk melakukan suatu proses. Prosedural ini akan digunakan sebagai algoritma dasar dalam mengkodekan prosedur yang ada. *Tools* yang digunakan adalah flowchart program.

Adapun perancangan prosedural adalah sebagai berikut:

1. *Flowchart* Pendaftaran Member

Prosedur pendaftaran member merupakan proses pengisian data member beserta pengecekan data yang dimasukan untuk mendaftar menjadi seorang *member*.

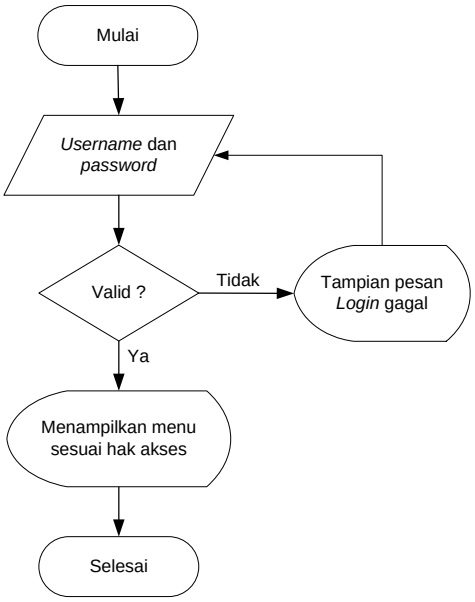
Flowchart pendaftaran *member* dapat dilihat pada gambar 3.90



Gambar 3.90 *Flowchart* pendaftaran *member*

2. *Flowchart Login*

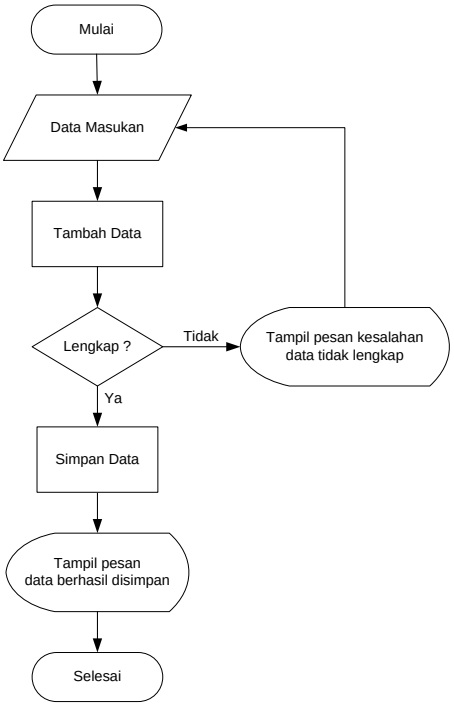
Diagram alir pada prosedur *login* adalah sebagai berikut :



Gambar 3.91 *Flowchart login*

3. *Flowchart* Tambah Data

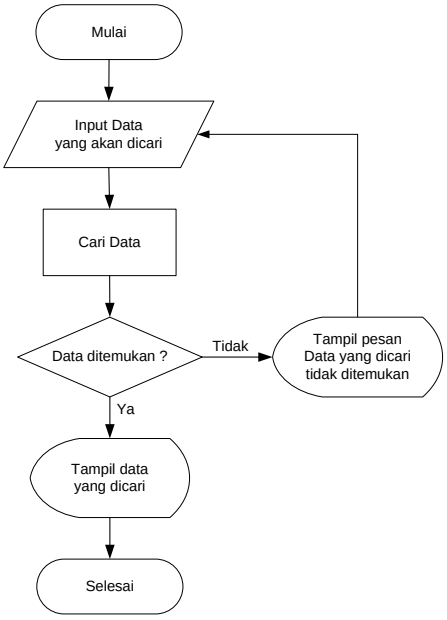
Diagram alir pada prosedur tambah data adalah sebagai berikut :



Gambar 3.92 *Flowchart* Tambah Data

4. *Flowchart* Cari Data

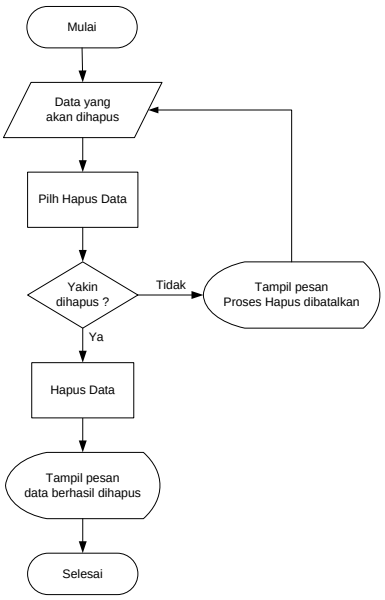
Diagram alir pada prosedur cari data adalah sebagai berikut :



Gambar 3.93 *Flowchart* Cari Data

5. *Flowchart* Hapus Data

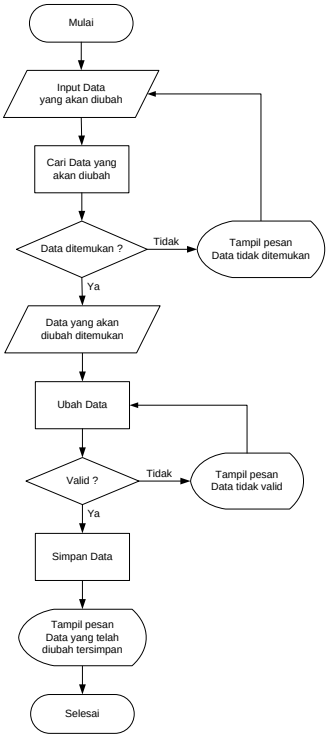
Diagram alir pada prosedur hapus data adalah sebagai berikut :



Gambar 3.94 *Flowchart* Hapus Data

6. *Flowchart* Ubah Data

Diagram alir pada prosedur ubah data adalah sebagai berikut :



Gambar 3.95 *Flowchart* Ubah Data