

SISTEM INFORMASI PENGELOLAAN STOK OBAT

(Studi Kasus Apotek Joint Farma, Yogyakarta)

SKRIPSI

Diajukan untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Teknik

Program Studi Teknik Informatika



Oleh :

Olivia Dian Kusumawati

NIM : 055314004

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS SANATA DHARMA

YOGYAKARTA

2009

**INFORMATION SYSTEM OF DRUG STOCKS MANAGEMENT
(Case Study on JOINT FARMA DRUGSTORE, Yogyakarta)**

A THESIS

**Presented as Partial Fulfillment of the Requirements
To Obtain Sarjana Teknik Degree
In Informatics Engineering Department**



By:

Olivia Dian Kusumawati

055314004

**INFORMATICS ENGINEERING STUDY PROGRAM
INFORMATICS ENGINEERING DEPARTMENT
FACULTY OF SCIENCE AND TECHNOLOGY
SANATA DHARMA UNIVERSITY
YOGYAKARTA
2009**

SKRIPSI
SISTEM INFORMASI PENGELOLAAN STOK OBAT
(Studi Kasus Apotek Joint Farma, Yogyakarta)

Oleh :

Olivia Dian Kusumawati

NIM : 055314004



Telah disetujui oleh :

Pembimbing

Eko Hari

Eko Hari Parmadi, S.Si,M.Kom.

Tanggal : *29 Sept 2009*

SKRIPSI
SISTEM INFORMASI PENGELOLAAN STOK OBAT
(Studi Kasus Apotek Joint Farma, Yogyakarta)

Dipersiapkan dan ditulis oleh :

Olivia Dian Kusumawati

NIM : 055314004

Telah dipertahankan di depan Panitia Penguji

Pada tanggal 17 September 2009

Dan dinyatakan memenuhi syarat

Susunan Panitia Penguji

Ketua

Nama Lengkap

Agnes Maria Polina, S.Kom.,M.Sc.

Sekretaris

Eko Hari Parmadi, S.Si,M.Kom.

Anggota

Stevanus Wisnu Wijaya, S.T.,M.T.

Tanda Tangan

.....
.....
.....

Yogyakarta, 29 September 2009

Fakultas Sains dan Teknologi

Universitas Sanata Dharma

Dekan,



Yosef Agung Cahyanta S.T.,M.T.

MOTTO

*“ Kita Dapat Memimpikan Suatu Prestasi
dan Membayangkan Suatu Kesuksesan,
Namun Hanya Keteguhanlah yang Membuat
Tujuan Kita Menjadi Kenyataan ”*

INTISARI

Kemajuan ilmu pengetahuan dan teknologi saat ini mulai merubah pola pikir dan cara kerja setiap manusia. Kegiatan-kegiatan manusia seperti kegiatan bisnis yang dulunya dapat dilakukan secara manual mulai tergantikan dengan komputer. Dalam bidang farmasi, komputer ini sangat dibutuhkan untuk mengolah data obat yang jumlahnya sangat banyak sekitar 500 hingga 1000. Oleh karena itu tulisan ini bertujuan memberikan sarana berupa sistem informasi pengelolaan stok obat pada Apotek Joint Farma.

Sistem informasi pengelolaan stok obat ini dirancang dengan pemodelan berorientasi obyek dan dibangun menggunakan bahasa pemrograman Java serta teknologi basisdata MySQL.

Hasil akhirnya adalah sistem informasi pengelolaan stok obat yang berfungsi dengan baik. Sistem ini dilengkapi dengan fasilitas pengingat obat kadaluarsa dan pengingat limit obat yang berjalan secara otomatis pada saat sistem dijalankan.

ABSTRACT

The progress of science and technology began to change the current mindset and workings of each human being. Human activities such as business activities that used to be done manually replaced with a computer. In the pharmaceutical sector, this computer is needed to process the amount of drug data which about 500 to 1000. Therefore, this paper aims to provide a means of information system of drug stocks management on Joint Farma Drugstore.

The system has been developed using by object oriented modeling and is built using Java programming language and MySQL database technologies.

The end result is a information system of drug stocks management which has a well function. This system is equipped with the facility expired medication reminders and medication reminders limit that runs automatically when the system is run.

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yesus Kristus atas berkat dan limpahan kasih karunia yang telah diberikan-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan judul :

“SISTEM INFORMASI PENGELOLAAN STOK OBAT (Studi Kasus Apotek Joint Farma, Yogyakarta)”.

Dorongan serta nasihat dari berbagai pihak sangat membantu sampai tersusunnya skripsi ini. Untuk itu, saya ingin mengucapkan terima kasih kepada :

1. Orang tua saya Paulus Suharwadi dan Rosalia Rape yang telah memberi dukungan moral, spiritual dan finansial dalam penyusunan skripsi.
2. Bapak Yosef Agung Cahyanta, S.T., M.T. selaku Dekan Fakultas Sains dan Teknologi Universitas Sanata Dharma Yogyakarta.
3. Bapak Puspaningtyas Sanjoyo Adi, S.T., M.T. selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Sanata Dharma Yogyakarta.
4. Bapak Eko Hari Parmadi, S.Si,M.Kom. selaku dosen pembimbing Skripsi. Terima kasih telah membimbing dan menyediakan waktu dalam memberikan pengarahan selama penulisan skripsi ini.
5. Kakak dan adikku, terima kasih atas dukungannya sehingga penulis dapat menyelesaikan studi.
6. *My Honey* Taufan, terima kasih atas doa, bantuan, kasih sayang ,dan perhatianmu. Semua itu yang menguatkan dan membuatku mampu bertahan sampai saat ini.
7. Teman-teman kost difa : Ina, Asien, Galih, Tiwi, Dini, Ayu Palembang, dan Grace, terima kasih atas dukungan dan bantuannya selama ini.
8. Teman-teman kontrakan “Sun Rise” : Vicimus, Yuan, Roland, dan Aldo terima kasih atas dukungan dan bantuannya selama ini.

9. Teman-teman TI 05 : Tepan, Mas Goundrex, Tombul, Cahyo, April, Icha, Kingkin, Ami, Ita, Niko, Dimas, dan teman-teman lain yang tidak dapat disebutkan satu-persatu. Terima kasih atas persahabatannya selama ini.

Dan semua teman-teman yang tidak dapat disebutkan satu-persatu.

PERNYATAAN KEASLIAN KARYA

Saya menyatakan sesungguhnya bahwa Tugas Akhir yang saya tulis ini tidak memuat karya atau bagian karya orang lain, kecuali yang telah disebutkan dalam kutipan daftar pustaka, sebagaimana karya ilmiah.

Yogyakarta, 29 September 2009

Penulis



Olivia Dian Kusumawati

LEMBAR PERNYATAAN PERSETUJUAN

PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini, saya mahasiswa Universitas Sanata Dharma :

Nama : Olivia Dian Kusumawati

NIM : 055314004

Demi pengembangan ilmu pengetahuan, saya memberikan kepada Universitas Sanata Dharma karya ilmiah saya yang berjudul :

“SISTEM INFORMASI PENGELOLAAN STOK OBAT (Studi Kasus Apotek Joint Farma, Yogyakarta)”

Beserta perangkat yang diperlukan (bila ada). Dengan demikian saya memberikan kepada Perpustakaan Universitas Sanata Dharma hak untuk menyimpan, mengalihkan dalam bentuk media lain, mengelolanya dalam bentuk pangkalan data, mendistribusikan secara terbatas, dan mempublikasikannya di Internet maupun di media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya maupun memberikan royalti kepada saya selama tetap mencantumkan nama saya sebagai penulis.

Demikian pernyataan ini saya buat sebenarnya.

Dibuat di Yogyakarta

Pada tanggal : 29 September 2009

Yang menyatakan



Olivia Dian Kusumawati

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN JUDUL	ii
HALAMAN PERSETUJUAN PEMBIMBING	iii
HALAMAN PENGESAHAN	Error! Bookmark not defined.
MOTTO	v
INTISARI	vi
ABSTRACT	vii
KATA PENGANTAR	viii
PERNYATAAN KEASLIAN KARYA	Error! Bookmark not defined.
LEMBAR PERNYATAAN.....	Error! Bookmark not defined.
DAFTAR ISI	xii
DAFTAR GAMBAR.....	xv
DAFTAR LISTING PROGRAM	xviii
DAFTAR TABEL.....	xviii
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Batasan Masalah	2
1.4. Tujuan dan Manfaat Penelitian	3
1.5. Metodologi Penelitian.....	3
1.6. Sistematika Penulisan	5
BAB II LANDASAN TEORI.....	7
2.1. Apotek: Definisi, Tugas, dan Fungsi	7
2.1.1. Sumber Daya Manusia di Apotek	8
2.1.2. Pengelolaan Sarana dan Prasarana Apotek	10
2.1.3. Penggolongan Obat.....	10
2.2. Sistem Informasi: Konsep dan Definisi	16

2.3.2.	Desain Sistem	18
2.3.	Java	25
2.3.1.	JDBC.....	25
2.4.	SQL (Structured Query Language).....	27
BAB III ANALISIS DAN PERANCANGAN SISTEM		30
3.1.	Sistem yang Ada Saat Ini.....	30
3.2.	Sistem yang Akan Dibangun	31
3.3.	Hardware dan Software Untuk Membuat Sistem	31
3.4.	Hardware dan Software Untuk Menjalankan Sistem	32
3.5.	Diagram Konteks	33
3.6.	Diagram Use Case	34
3.6.1.	Ringkasan Use Case.....	35
3.6.2.	Narasi Use Case	39
3.7.	Diagram Activity	76
3.8.	Class Diagram.....	94
3.9.	Diagram Sequence	94
3.10.	Diagram Class Desain (lampiran).....	130
3.11.	Desain User Interface.....	130
BAB IV IMPLEMENTASI SISTEM		137
4.1.	Implementasi Form Login.....	137
4.2.	Implementasi Form Utama	138
4.3.	Implementasi Form Pembelian	141
4.4.	Implementasi Form Penjualan	161
4.5.	Implementasi Form Cetak Laporan Penjualan.....	166
4.6.	Implementasi Form Warning Limit	169
4.7.	Implementasi Form Warning ED.....	171
BAB V ANALISIS HASIL DAN PEMBAHASAN		175
5.1.	Pengetesan Program dari <i>Programmer</i>	175
5.2.	Pengetesan Program dari <i>User</i>	176
5.3.	Kelebihan Sistem	176
5.4.	Kelemahan Sistem	177
BAB VI PENUTUP		178
6.1.	Kesimpulan	178

6.2. Saran	179
DAFTAR PUSTAKA	
LAMPIRAN	

DAFTAR GAMBAR

KETERANGAN	HALAMAN
Gambar 2.1. Simbol Use Case (Whitten et al, 2004)	18
Gambar 2.2. Simbol Actor (Whitten et al, 2004)	19
Gambar 2.3. Simbol depend on (Whitten et al, 2004)	19
Gambar 2.4 Activity Diagram (Whitten et al, 2004)	20
Gambar 2.5 Sequence Diagram (Whitten et al, 2004)	23
Gambar 2.6 Arsitektur JDBC dan MySQL	26
Gambar 3.1 Diagram Konteks	33
Gambar 3.2 Diagram Use-Case	34
Gambar 3.3 Diagram Activity Log in	77
Gambar 3.4 Diagram Activity Mengisi Data Pegawai	77
Gambar 3.5 Diagram Activity Mengubah Data Pegawai	78
Gambar 3.6 Diagram Activity Menghapus Data Pegawai	78
Gambar 3.7 Diagram Activity Menghapus Daftar Login	79
Gambar 3.8 Diagram Activity Mengisi Data Penjualan	79
Gambar 3.9 Diagram Activity Menghapus Data Penjualan	80
Gambar 3.10 Diagram Activity Mengisi Data Pembelian	81
Gambar 3.11 Diagram Activity Mengubah Data Pembelian	82
Gambar 3.12 Diagram Activity Menghapus Data Pembelian	82
Gambar 3.13 Diagram Activity Mengubah Data Obat	83
Gambar 3.14 Diagram Activity Menghapus Data Obat	84
Gambar 3.15 Diagram Activity Mengisi Data Distributor	85
Gambar 3.16 Diagram Activity Mengubah Data Distributor	86
Gambar 3.17 Diagram Activity Menghapus Data Distributor	87
Gambar 3.18 Diagram Activity Cetak Laporan	88
Gambar 3.19 Diagram Activity Mengisi Data Retur	89
Gambar 3.20 Diagram Activity Menghapus Data Retur	90
Gambar 3.21 Diagram Activity Mengisi Data Embalase	90
Gambar 3.22 Diagram Activity Mengubah Data Embalase	91
Gambar 3.23 Diagram Activity Menghapus Data Embalase	91
Gambar 3.24 Diagram Activity Mengisi Data Toeslag	92
Gambar 3.25 Diagram Activity Mengubah Data Toeslag	92
Gambar 3.26 Diagram Activity Menghapus Data Toeslag	93
Gambar 3.27 Diagram Activity Pengingat Kadaluaarsa	93

Gambar 3.28 Diagram Activity Peningat Limit	94
Gambar 3.29 Diagram Activity LOG OUT	94
Gambar 3.30 Class Diagram System	95
Gambar 3.31 Diagram sequence Log in	102
Gambar 3.32 Diagram sequence Mengisi Data Pegawai	103
Gambar 3.33 Diagram sequence Mengubah Data Pegawai	104
Gambar 3.34 Diagram sequence Menghapus Data Pegawai	105
Gambar 3.35 Diagram sequence Menghapus Daftar Login	106
Gambar 3.36 Diagram sequence Mengisi Data Penjualan	107
Gambar 3.37 Diagram sequence Menghapus Data Penjualan	108
Gambar 3.38 Diagram sequence Mengisi Data Pembelian	109
Gambar 3.39 Diagram sequence Mengubah Data Pembelian	110
Gambar 3.40 Diagram sequence Menghapus Data Pembelian	111
Gambar 3.41 Diagram sequence Mengubah Data Obat	112
Gambar 3.42 Diagram Sequence Menghapus Data Obat	113
Gambar 3.43 Diagram Sequence Mengisi Data Distributor	114
Gambar 3.44 Diagram Sequence Mengubah Data Distributor	115
Gambar 3.45 Diagram Sequence Menghapus Data Distributor	116
Gambar 3.46 Diagram Sequence Cetak Laporan	117
Gambar 3.47 Diagram Sequence Mengisi Data Retur	118
Gambar 3.48 Diagram Sequence Menghapus Data Retur	119
Gambar 3.49 Diagram Sequence Mengisi Data Embalase	120
Gambar 3.50 Diagram Sequence Mengubah Data Embalase	121
Gambar 3.51 Diagram Sequence Menghapus Data Embalase	122
Gambar 3.52 Diagram Sequence Mengisi Data Toeslag	123
Gambar 3.53 Diagram Sequence Mengubah Data Toeslag	124
Gambar 3.54 Diagram Sequence Menghapus Data Toeslag	125
Gambar 3.55 Diagram Sequence Peningat Kadaluarsa	126
Gambar 3.56 Diagram Sequence Peningat Limit	127
Gambar 3.57 Diagram Sequence LOG OUT	128
Gambar 3.58 Form Login	129
Gambar 3.59 Form Utama	129
Gambar 3.60 Form Obat-Obatan	130
Gambar 3.61 Form Kepegawaian	130
Gambar 3.62 Form Disributor	131

Gambar 3.63 Form Daftar Login	131
Gambar 3.64 Form Penjualan	132
Gambar 3.65 Form Total Harga	132
Gambar 3.66 Form Penjualan dengan Resep	133
Gambar 3.67 Form Retur	133
Gambar 3.68 Form Reminder	134
Gambar 3.69 Laporan Pembelian	134
Gambar 3.70 Laporan Pembelian	135
Gambar 3.71 Laporan Register Psikotropika	135
Gambar 3.72 Laporan Obat Wajib Apotek (OWA)	136
Gambar 3.73 Laporan Obat Wajib Apotek (OWA)	136
Gambar 3.74 Nota Penjualan	137
Gambar 4.1 Form Login	138
Gambar 4.2 Peringatan Gagal Login	139
Gambar 4.3 form utama	140
Gambar 4.4 form utama pilihan menu	140
Gambar 4.5 form utama pilihan setup	141
Gambar 4.6 form utama pilihan cetak	141
Gambar 4.7 form utama pilihan Bantuan	142
Gambar 4.8 form pembelian	143
Gambar 4.9 form untuk mencari data distributor	146
Gambar 4.10 form detail pembelian	146
Gambar 4.11 form untuk mencari data obat	147
Gambar 4.12 Form Penjualan Counter	165
Gambar 4.13 Form Penjualan Resep	165
Gambar 4.14 Form untuk mencari data obat	166
Gambar 4.15 Form untuk mencari data embalase	166
Gambar 4.16 Form untuk mencari data toeslag	167
Gambar 4.17 Form cetak laporan penjualan harian	167
Gambar 4.18 Form cetak laporan penjualan bulanan	168
Gambar 4.19 preview laporan penjualan berdasarkan bulan	168
Gambar 4.20 form warning limit	170
Gambar 4.21 form warning ed	172

DAFTAR LISTING PROGRAM

KETERANGAN	HALAMAN
Listing 4.1 Kelas DataModelPembelian.java	149
Listing 4.2 Kelas Pembelian.java	151
Listing 4.3 Kelas DataModelDetailPembelian.java	154
Listing 4.4 Kelas DetailPembelian.java	155
Listing 4.5 Store Procedure spInsertDetailBeli	161
Listing 4.6 Store Procedure spInsertJual	164
Listing 4.7 Kelas MyJasperViewe.java	169
Listing 4.8 Kelas DataModelWarningLimit.java	171
Listing 4.9 Kelas Obat.java	171
Listing 4.9 Kelas DataModelWarningED.java	173
Listing 4.10 Kelas Obat.java	174

DAFTAR TABEL

KETERANGAN	HALAMAN
Tabel 3.1 Ringkasan Use Case	35
Tabel 3.2 Narasi Use-Case Log in	39
Tabel 3.3 Narasi Use-Case Mengisi Data Pegawai	41
Tabel 3.4 Narasi Use-Case Mengubah Data Pegawai	42
Tabel 3.5 Narasi Use-Case Menghapus Data Pegawai	44
Tabel 3.6 Narasi Use-Case Menghapus Daftar Login	45
Tabel 3.7 Narasi Use-Case Mengisi Data Penjualan	46
Tabel 3.8 Narasi Use-Case Menghapus Data Penjualan	47
Tabel 3.9 Narasi Use-Case Mengisi Data Pembelian	49
Tabel 3.10 Narasi Use-Case Mengubah Data Pembelian	51
Tabel 3.11 Narasi Use-Case Menghapus Data Pembelian	52
Tabel 3.12 Narasi Use-Case Mengubah Data Obat	53
Tabel 3.13 Narasi Use-Case Menghapus Data Obat	55
Tabel 3.14 Narasi Use-Case Mengisi Data Distributor	57
Tabel 3.15 Narasi Use-Case Mengubah Data Distributor	58
Tabel 3.16 Narasi Use-Case Menghapus Data Distributor	59
Tabel 3.17 Narasi Use-Case Cetak Laporan	61
Tabel 3.18 Narasi Use-Case Mengisi Data Retur	62
Tabel 3.19 Narasi Use-Case Menghapus Data Retur	64

Tabel 3.20 Narasi Use-Case Mengisi Data Embalase	65
Tabel 3.21 Narasi Use-Case Mengubah Data Embalase	67
Tabel 3.22 Narasi Use-Case Menghapus Data Embalase	68
Tabel 3.23 Narasi Use-Case Mengisi Data Toeslag	69
Tabel 3.24 Narasi Use-Case Mengubah Data Toeslag	71
Tabel 3.25 Narasi Use-Case Menghapus Data Toeslag	72
Tabel 3.26 Narasi Use-Case Pengingat Kadaluarsa	73
Tabel 3.27 Narasi Use-Case Pengingat Limit	74
Tabel 3.28 Narasi Use-Case LOG OUT	75

BAB I

PENDAHULUAN

1.1. Latar Belakang

Kemajuan ilmu pengetahuan dan teknologi saat ini mulai merubah pola pikir dan cara kerja setiap manusia. Kegiatan-kegiatan manusia seperti kegiatan bisnis yang dulunya dapat dilakukan secara manual mulai tergantikan dengan komputer. Penggunaan komputer pada berbagai perusahaan swasta juga mulai berkembang, termasuk dalam bidang obat-obatan. Komputer ini sangat dibutuhkan untuk menghasilkan informasi yang cepat, akurat, dan dapat mengefisiensikan pekerjaan.

Apotek Joint Farma merupakan suatu usaha dagang yang bergerak dalam bidang penjualan obat. Apotek ini bertempat di Jl. Jogja Wonosari Km.17 Bukit Pathuk Gunung Kidul. Dalam kegiatan pendataan obat, pencetakan laporan, dan segala kegiatan bisnisnya masih dilakukan secara manual. Sedangkan obat yang ada di apotek Joint Farma terdiri dari berbagai jenis obat, antara lain adalah obat dengan resep, obat wajib apotek, obat generik berlogo, dan obat bebas. Obat-obatan tersebut secara keseluruhan jumlahnya sangat banyak sekitar 500 hingga 1000 jenis obat, sehingga apabila dilakukan pendataan obat secara manual akan membutuhkan waktu yang cukup lama. Selain itu pencetakan laporan secara manual, juga akan memakan waktu yang lama, demikian halnya dengan pengecekan obat yang hampir kadaluarsa dan

pencarian data obat. Hal tersebut akan menghambat kinerja dari apotek Joint Farma.

Berdasarkan hal-hal tersebut di atas, Apotek Joint Farma menginginkan sebuah sistem informasi pengelolaan stok obat yang dapat membantu kinerja dari apotek tersebut.

1.2. Rumusan Masalah

Berdasarkan latar belakang masalah di atas maka dapat dirumuskan menjadi beberapa masalah sebagai berikut:

1. Bagaimana membangun sistem informasi pengelolaan data obat pada Apotek Joint Farma yang mampu memberikan kemudahan untuk mendapatkan informasi dengan cepat dan tepat?
2. Bagaimana membuat sebuah pengingat otomatis agar tidak terjadi keterlambatan dalam meretur obat yang hampir kadaluarsa dan memesan obat sehingga stok obat tidak kosong?

1.3. Batasan Masalah

Beberapa batasan masalah pada “Sistem Informasi Pengelolaan Stok Obat Apotek Joint Farma” adalah sebagai berikut:

1. Sistem informasi yang dibuat ini mempunyai kemampuan untuk melakukan proses pencatatan data obat baik obat yang masuk maupun obat yang keluar, pembuatan laporan-laporan yang terkait dengan kegiatan bisnis apotek joint farma, pembuatan label-label obat, dan mengingatkan

secara otomatis setiap 6 bulan sebelum tanggal kadaluarsa obat, serta mengingatkan stok obat yang telah mendekati limit.

2. Bahasa pemrograman yang digunakan adalah Java yang bekerja di bawah sistem operasi Windows.
3. Database yang digunakan untuk menyimpan data obat adalah MySQL.

1.4. Tujuan dan Manfaat Penelitian

Penelitian ini bertujuan untuk memberikan sarana berupa sistem informasi pengelolaan stok obat pada Apotek Joint Farma.

Selain itu penelitian ini juga bermanfaat untuk memudahkan Apotek Joint Farma agar dapat memperoleh informasi dengan cepat dan tepat dengan memanfaatkan aplikasi yang dibuat.

1.5. Metodologi Penelitian

Metodologi yang digunakan dalam penelitian ini adalah metode FAST (*Framework for the Application of Systems Thinking*), yang meliputi tahap-tahap:

- *Scope Definition Phase*, fase ini merupakan tahap pertama dalam melakukan pengembangan suatu sistem. Dalam fase ini dilakukan penentuan batasan-batasan sistem.
- *Problem Analysis Phase*, dalam fase ini dilakukan analisis menyeluruh terhadap permasalahan dari sistem yang akan dikembangkan dengan cara melakukan analisa-analisa terhadap permasalahan sistem, penyebab

permasalahan tersebut, serta menentukan apakah permasalahan tersebut dapat diselesaikan.

- *Requirement Analysis Phase*, dalam fase ini dilakukan analisa terhadap *business requirement* sesuai dengan *requirement* yang dibutuhkan dan diinginkan *user* yang menggunakan sistem tersebut.
- *Logical Design Phase*, dalam fase ini *business requirement* yang ada diterjemahkan dalam bentuk gambar-gambar yang disebut *system model*.
- *Decision Analysis Phase*, dalam fase ini permasalahan yang dihadapi sistem biasanya dapat diselesaikan dengan berbagai solusi.
- *Physical Design and Integration*, dalam fase ini dilakukan desain database, desain input, desain output, dan desain interface.
- *Construction and Testing*, dalam fase ini dilakukan pembuatan program aplikasi (*pengimplementasian interface*) dengan menggunakan bahasa pemrograman tertentu, dalam kasus ini digunakan bahasa pemrograman Java.
- *Installation and Delivery*, dalam fase ini dilakukan training kepada *user* dan membantu dalam proses penginstalan.

1.6. Sistematika Penulisan

Struktur dari laporan Tugas Akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini membahas tentang deskripsi umum isi penulisan yang meliputi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisi dasar teori yang digunakan untuk pembahasan dalam penulisan laporan tugas akhir ini.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tentang sistem lama dan sistem baru beserta rancangan sistemnya.

BAB IV IMPLEMENTASI SISTEM

Bab ini berisi implementasi sistem dari perancangan yang telah dibuat sebelumnya.

BAB V ANALISIS HASIL DAN PEMBAHASAN

Bab ini berisi analisa hasil dari implementasi sistem dengan cara melakukan testing kepada user.

BAB VI PENUTUP

Bab ini berisi kesimpulan dan saran dari pembahasan dan implementasi yang telah dilakukan dalam penulisan tugas akhir ini.

BAB II

LANDASAN TEORI

Bab ini akan membahas beberapa teori yang digunakan dalam membangun sistem pengelolaan stok obat apotek joint farma. Teori-teori tersebut adalah sebagai berikut:

2.1. Apotek: Definisi, Tugas, dan Fungsi

Berdasarkan PP No.25 tahun 1980 tentang perubahan atas PP No. 26 tahun 1965, maka pengertian apotik menjadi “Apotik adalah suatu tempat tertentu, tempat dilakukan pekerjaan kefarmasian dan penyaluran obat kepada masyarakat.”.

Selain itu apotik juga memiliki fungsi dan tugas, antara lain sebagai berikut:

1. Tempat pengabdian profesi seorang apoteker yang telah mengucapkan sumpah jabatan.
2. Sarana farmasi yang melaksanakan peracikan, pengubahan bentuk, pencampuran dan penyerahan obat atau bahan obat.
3. Sarana penyalur perbekalan farmasi yang harus menyebarkan obat yang diperlukan masyarakat secara meluas dan merata.

2.1.1. Sumber Daya Manusia di Apotek

Sebuah apotek dapat beranggotakan orang-orang dengan profesi sebagai berikut:

➤ Apoteker.

Dalam kepmenkes no. 1027 tahun 2004 tentang Standar Pelayanan Kefarmasian di Apotek, Apoteker di apotek senantiasa harus memiliki kemampuan menyediakan dan memberikan pelayanan yang baik, mengambil keputusan yang tepat, kemampuan berkomunikasi antar profesi, menempatkan diri sebagai pemimpin dalam situasi multidisipliner, kemampuan mengelola SDM secara efektif, selalu belajar sepanjang karier, dan membantu memberi pendidikan dan memberi peluang untuk meningkatkan pengetahuan.

Di apotek, seorang apoteker juga dapat bertugas sebagai:

- a. Apoteker Pengelola Apotek (APA) adalah Apoteker yang telah diberi Surat izin Apotek (SIA). Setiap satu apotek harus ada 1 APA dan seorang Apoteker hanya dapat menjasi APA di satu apotek saja.
- b. Apoteker Pendamping adalah Apoteker yang bekerja di apotek di samping APA dan/atau menggantikannya pada jam-jam tertentu pada hari buka apotek.
- c. Apoteker Pengganti adalah Apoteker yang menggantikan APA selama APA tersebut tidak berada

di tempat lebih dari 3 bulan secara terus menerus, apoteker pengganti ini harus memiliki SIK dan tidak bertindak sebagai APA di apotek lain.

➤ Asisten Apoteker (AA).

Asisten Apoteker adalah tenaga kesehatan yang berijazah Sekolah Asisten Apoteker / Sekolah Menengah Farmasi, Akademi Farmasi Jurusan Farmasi Politeknik Kesehatan, Akademi Analis Farmasi dan Makanan Jurusan Analis Farmasi dan Makanan Politeknik Kesehatan sesuai dengan peraturan perundang-undangan yang berlaku.

➤ Pemilik Sarana Apotek (PSA)

Pemilik sarana apotek ini tidak harus ada. Apoteker Pengelola Apotek (APA) dapat menjadi pemilik sarana apotek sekaligus.

➤ Juru resep (reseptir), kasir, akuntan, petugas kebersihan dan karyawan lain tidak harus ada, sesuai dengan kebutuhan apotek saja.

2.1.2. Pengelolaan Sarana dan Prasarana Apotek

Komoditas di apotek dapat berupa sediaan farmasi, perbekalan kesehatan, alat kesehatan maupun yang lainnya. Sediaan farmasi adalah obat, bahan obat, obat tradisional, dan kosmetik. Perbekalan kesehatan adalah semua bahan selain obat dan peralatan yang diperlukan untuk menyelenggarakan upaya kesehatan. Sedangkan alat kesehatan adalah bahan, instrument apparatus, mesin, implant yang tidak mengandung obat yang digunakan untuk mencegah, mendiagnosis, menyembuhkan dan meringankan penyakit, merawat orang sakit serta memulihkan kesehatan dan untuk membentuk struktur dan memperbaiki fungsi tubuh.

2.1.3. Penggolongan Obat

Untuk peningkatan keamanan dan ketepatan penggunaan serta pengamanan lalu lintas obat dan hubungannya dengan aksi obat yang ditimbulkan didalam badan, dan bahayanya obat tersebut bagi pasien, telah dikeluarkan peraturan mengenai penggolongan obat sebagai berikut:

1. Obat Bebas dan Bebas Terbatas atau Daftar W (*Warschuwing*) atau OTC (*Over The Counter*)

Obat jadi yang termasuk dalam obat ini banyak dijumpai di pasaran, baik di Apotek, Toko Obat Berijin ataupun di tempat lain.

Menurut Surat Edaran dari Direktorat Jendral Pengawasan Obat dan Makanan Dep.Kes.RI., No.: 02469/A/VI/1983 tentang Obat yang boleh dijual oleh Toko Obat Berijin, sesuai dengan SK. Menteri Kesehatan RI. Nomor : 2380/A/SK/VI/83 tanggal 15 Juni 1983 tentang Tanda Khusus Untuk Obat Bebas dan Obat Bebas Terbatas, maka sejak diberlakukannya SK tersebut yang boleh dijual oleh Toko Obat Berijin hanyalah obat yang dalam bungkus luar dan etiketnya tertera **Tanda khusus:**

- **Lingkaran hijau** dengan garis tepi berwarna hitam, yaitu tanda khusus untuk OBAT BEBAS.
- **Lingkaran biru** dengan garis tepi berwarna hitam, yaitu tanda khusus untuk OBAT BEBAS TERBATAS.

Selain itu merupakan pelengkap dari keharusan mencantumkan Tanda Peringatan yang ditetapkan dalam SK. Menteri Kesehatan No. 6355/Dir.Jend/SK/1969 tanggal 28 Oktober 1969. yaitu:

- a) P.No. 1 Awas Obat Keras, Bacalah aturan Memakainya.
- b) P.No. 2 Awas Obat Keras, Hanya untuk kumur, jangan ditelan.
- c) P.No. 3 Awas Obat Keras, Hanya untuk bagian luar badan.
- d) P.No. 4 Awas Obat Keras, Hanya untuk Dibakar.
- e) P.No. 5 Awas Obat Keras, Tidak boleh ditelan.
- f) P.No. 6 Awas Obat Keras, Obat Wasir, jangan ditelan.

2. Obat Wajib Apotek (OWA)

Obat Wajib Apotek (OWA) adalah obat keras yang dapat diserahkan oleh Apoteker kepada pasien di Apotek tanpa resep dokter.

Obat Wajib Apotek merupakan program pemerintah dengan tujuan untuk meningkatkan kemampuan masyarakat dalam menolong dirinya sendiri guna mengatasi masalah kesehatan. Selain tujuan di atas tersebut, pemerintah juga mengharapkan dengan adanya pelayanan OWA oleh Apoteker dimasyarakat dapat meningkatkan pelayanan KIE (Komunikasi, Informasi, dan Edukasi).

Pelaksanaan OWA tersebut oleh apoteker harus sesuai dengan yang diwajibkan pada diktum kedua SK. Menteri Kesehatan Nomor : 347/Men.Kes./SK/VII/1990 tentang Obat Wajib Apotek yaitu sebagai berikut:

- a. Memenuhi ketentuan dan batasan tiap jenis obat per pasien yang disebutkan dalam OWA yang bersangkutan.
- b. Membuat catatan pasien serta obat yang telah diserahkan.
- c. Memberikan informasi meliputi dosis dan aturan pakainya, kontradiksi, efek samping, dan lain-lain yang perlu diperhatikan oleh pasien.

Obat Wajib Apotek dapat diperoleh tanpa resep dokter baik di Apotek maupun Apotek Rumah Sakit.

3. Obat Keras atau Daftar G (*Gevaarlijk*)

Obat keras diatur menurut Undang-undang obat keras St.No.419, tanggal 22 Desember 1949. Pada pasal 1 butir a, disebutkan bahwa :

Obat-obat keras, yaitu obat-obatan yang tidak digunakan untuk keperluan teknik, yang mempunyai khasiat mengobati, menguatkan, membaguskan, mendesinfeksi-kan dan lain-lain tubuh manusia, baik dalam bungkus maupun tidak, yang ditetapkan oleh Sekretaris Van Staat, Hoofd van het Departement van Gesonheid, menurut ketentuan dalam pasal 2.

Pasal 1 butir k : obat-obatan G, adalah obat-obat keras yang oleh Sec.V.St. didaftar pada obat-obatan berbahaya (*gevaarlijk*; daftar G).

Pada tanggal 15 Maret 1977, Direktur Jendral Pengawasan Obat dan Makanan atas nama Menteri Kesehatan RI., mengeluarkan SK Nomor : 197/A/SK/77 tentang Pembungkusan dan penandaan Obat Keras sebagai berikut:

Pasal 1

- Pada bungkus luar,, etiket pada setiap pembungkus obat jadi seperti blister, strip aluminium / cellophane, botol, kotak,

doos, kaleng, tube, vial, ampul dari semua obat keras, harus dicantumkan kalimat : **“Harus dengan resep dokter”**.

- Pencantuman kalimat tersebut di atas harus dalam bahasa Indonesia, tercetak jelas dengan warna kontras dibandingkan dengan warna dasar pembungkus atau wadahnya.

Obat keras hanya diperoleh dengan resep dokter di Apotek, Apotek Rumah Sakit, Puskesmas, dan Balai Pengobatan.

Berdasarkan SK. Menteri Kesehatan RI., Nomor 02396/A/SK/VII/86, tentang Tanda Khusus Obat Keras Daftar G, disebutkan bahwa tanda khusus untuk obat keras adalah lingkaran berwarna merah dengan garis tepi berwarna hitam dengan huruf **K** yang menyentuh garis tepi. Oleh karena itu Tanda khusus untuk golongan obat keras selain keharusan mencantumkan kalimat **“Harus dengan resep dokter”** juga diharuskan mencantumkan tanda khusus **lingkaran bulat berwarna merah** dengan huruf **K** didalam lingkaran tersebut.

4. Obat Psikotropika

Berdasarkan Undang-undang RI., Nomor : 5 Tahun 1997 tentang Psikotropika, disebutkan bahwa yang dimaksud Psikotropika adalah zat atau obat, baik alamiah maupun sintesis bukan narkotika yang berkhasiat psikoaktif melalui pengaruh

selektif pada susunan saraf pusat yang menyebabkan perubahan khas pada aktivitas mental dan perilaku. Adapun tujuan pengaturan di bidang psikotropika adalah :

- a. Menjamin ketersediaan psikotropika guna kepentingan pelayanan kesehatan dan ilmu pengetahuan.
- b. Mencegah terjadinya penyalahgunaan psikotropika.
- c. Memberantas peredaran gelap psikotropika.

Sehubungan dengan banyaknya penyalahgunaan obat golongan psikotropika ini di masyarakat, maka pemerintah telah mengeluarkan peraturan yang menyangkut tentang pengadaan, distribusi, dan pelayanan obat psikotropika. Termasuk dalam hal ini adalah pemesanan obat oleh Apotek dengan Surat Pesanan Psikotropika, pelayanan resep obat psikotropika, dan laporan rutin penggunaan obat psikotropika oleh sarana pelayanan kesehatan (Apotek, Rumah Sakit, PBF, maupun Industri Farmasi).

5. Obat Narkotika

Narkotika atau Narkotics berasal dari kata Narcosis yang berarti narkose atau menidurkan yaitu zat atau obat-obatan yang membiuskan. Dalam pengertian ini, Narkotika adalah zat atau obat yang dapat mengakibatkan ketidaksadaran atau pembiusan,

karena zat-zat tersebut bekerja langsung mempengaruhi susunan saraf pusat.

Untuk lebih meningkatkan pengendalian dan pengawasan serta meningkatkan upaya pencegahan dan pemberantasan penyalahgunaan dan peredaran gelap narkoba, maka pemerintah memandang perlu untuk memperbaharui Undang-undang Nomor 9 Tahun 1976 tentang Narkoba dengan membentuk Undang-undang baru yaitu Undang-undang Nomor 22 Tahun 1997 tentang Narkoba. Undang-undang Narkoba terbaru ini mempunyai tujuan sebagai berikut:

- a. Menjamin ketersediaan narkoba untuk kepentingan pelayanan kesehatan dan/atau pengembangan ilmu pengetahuan.
- b. Mencegah terjadinya penyalahgunaan narkoba.
- c. Memberantas peredaran gelap narkoba.

2.2. Sistem Informasi: Konsep dan Definisi

Sistem adalah kumpulan berbagai elemen yang saling bekerjasama untuk mencapai suatu tujuan tertentu. (Whitten et al, 2004)

Sistem informasi adalah kumpulan orang, data, proses, dan teknologi informasi yang berinteraksi untuk mengumpulkan, memproses, menyimpan, dan menyediakan informasi yang dibutuhkan untuk mendukung suatu organisasi. (Whitten et al, 2004)

Sistem informasi memiliki komponen-komponen dasar, antara lain:

- a) **Piranti Keras (*Hardware*)** adalah serangkaian peralatan seperti prosesor , monitor, keyboard, dan printer. Secara bersama-sama, berbagai peralatan tersebut menerima data serta informasi, memprosesnya, dan menampilkannya.
- b) **Piranti Lunak (*software*)** adalah sekumpulan program yang memungkinkan piranti keras untuk memproses data.
- c) **Basis Data (*database*)** adalah sekumpulan arsip (*file*), tabel, relasi, dan lain-lain yang saling berkaitan dan menyimpan data serta berbagai hubungan di antaranya.
- d) **Jaringan (*network*)** adalah sistem koneksi (dengan kabel atau nirkabel) yang memungkinkan adanya berbagai sumber daya antar komputer yang berbeda.
- e) **Prosedur** adalah serangkaian instruksi mengenai bagaimana menggabungkan berbagai komponen di atas agar dapat memproses informasi dan menciptakan hasil yang diinginkan.
- f) **Orang** adalah berbagai individu yang bekerja dengan sistem informasi, berinteraksi dengannya, atau menggunakan hasilnya.

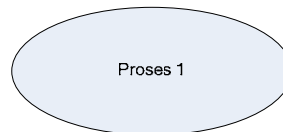
2.3.2. Desain Sistem

Untuk mendesain sebuah sistem informasi yang menggunakan pendekatan berbasis objek, maka digunakan beberapa diagram berikut untuk fase desain, yaitu:

A. Use Case Diagram

Use case diagram adalah sebuah diagram yang menggambarkan interaksi antara sistem, eksternal sistem dan pemakai. Use case merupakan bagian dari keseluruhan sistem. Digambarkan secara grafik dengan elips yang horizontal dengan nama dari use case tertera di atas, di bawah atau di dalam ellips.

Gambar 2.1 merupakan simbol use case.



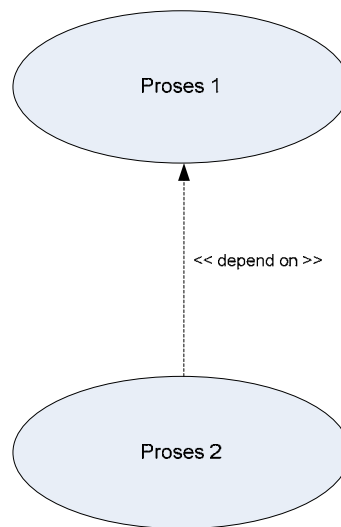
Gambar 2.1. Simbol Use Case (Whitten et al, 2004)

Aktor merupakan segala sesuatu yang dibutuhkan untuk berinteraksi dengan sistem untuk mengubah informasi. Dapat berupa orang, organisasi atau sistem informasi yang lain atau juga suatu waktu kejadian. *Gambar 2.2* merupakan simbol dari aktor .



Gambar 2.2. Simbol Actor (Whitten et al, 2004)

Use case depends on relationship merupakan sebuah relasi use case yang menentukan bahwa use case yang lain harus dibuat sebelum use case yang sekarang. Digambarkan sebagai anak panah yang dimulai dari satu use case dan menunjuk ke use case yang *depend on* kepadanya. Setiap relasi *depend on* diberi label “<<depend on>>”. *Gambar 2.3* merupakan simbol *depend on*.



Gambar 2.3. Simbol depend on (Whitten et al, 2004)

B. Activity Diagram

Activity diagram digunakan untuk menggambarkan proses bisnis, langkah-langkah use case, dan logika perilaku obyek/metode. Gambar 2.4 merupakan contoh dari *activity diagram*.



Gambar 2.4 Activity Diagram (Whitten et al, 2004)

Keterangan Gambar 2.4 adalah:

1. **Node awal / *Initial node*** merupakan lingkaran penuh yang menyatakan awal proses.
2. **Aksi / *Actions*** merupakan kotak berujung bulat yang menyatakan langkah tunggal. Sederetan aksi akan membentuk aktivitas total yang diperlihatkan dengan diagram.
3. **Alur / *Flow*** merupakan panah pada diagram menunjukkan alur aksi. Tidak perlu keterangan kecuali jika alur tsb keluar dari notasi keputusan.
4. **Keputusan / *Decision*** merupakan bentuk belah ketupat dengan satu alur masuk dan dua atau lebih alur keluar. Alur keluar diberi keterangan untuk mengindikasikan kondisi.
5. **Penggabungan / *Merge*** merupakan bentuk belah ketupat dengan banyak alur masuk dan satu alur keluar. Notasi ini menggabungkan alur yang sebelumnya dipisah dengan keputusan. Proses berlanjut dengan banyak alur masuk ke penggabungan.
6. **Pemisah / *Fork*** merupakan garis hitam dengan satu alur masuk dan dua atau lebih alur keluar. Aksi pada alur paralel dibawah pemisah dapat terjadi dalam beberapa urutan atau secara bersamaan.

7. **Penghubung / Join** merupakan garis hitam dengan dua atau lebih alur masuk dan satu alur keluar. Menandai akhir dari proses bersamaan. Semua aksi yang masuk ke join harus diselesaikan sebelum proses berlanjut.
8. **Aktivitas akhir / Activity final** merupakan lingkaran padat didalam lingkaran berlubang menyatakan akhir proses.
9. **Indikator subaktivitas / Subactivity indicator** merupakan simbol dalam aksi ini menandakan bahwa aksi dipecah menjadi diagram aktivitas yang terpisah. Hal ini untuk membantu diagram aktivitas agar tidak menjadi kompleks.
10. **Penghubung / Connector** merupakan huruf didalam lingkaran yang membantu untuk mengatur kompleksitas. Alur masuk ke dalam konektor akan melompat ke alur keluar dengan huruf yang sesuai.

C. Class Diagram Analysis

Class diagram analysis merupakan gambaran grafis dari struktur obyek statis sistem. Class diagram ini menunjukkan kelas-kelas obyek yang menyusun sistem serta relasi diantara kelas-kelas obyek. Obyek pada class diagram ini dapat disimpan dalam dua 2 kelas, yaitu:

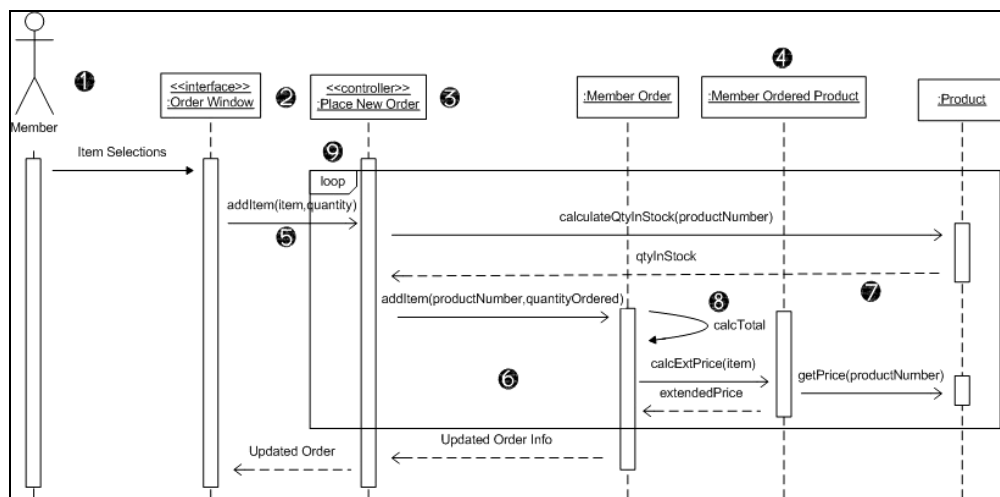
- **Kelas Persisten** adalah sebuah kelas yang mendeskripsikan obyek yang akan tetap ada meskipun eksekusi program

sudah selesai dengan kata lain obyek tersebut disimpan secara permanen di dalam basis data.

- **Kelas obyek Transien** adalah sebuah kelas yang mendeskripsikan obyek yang dibuat secara temporer dan hanya dikenali selama program dieksekusi.

D. Sequence Diagram

Sequence diagram merupakan diagram UML yang memodelkan logika dari use case dengan menggambarkan interaksi pesan-pesan antara obyek dalam urutan waktu. Sequence diagram terdiri-dari beberapa bagian seperti yang terlihat pada *Gambar 2.5*.



Gambar 2.5 Sequence Diagram (Whitten et al, 2004)

Keterangan Gambar 2.5 adalah:

1. Actor
2. Interface class
3. Controller class
4. Entity classes
5. Messages
6. Activation bars
7. Return messages
8. Self-call
9. Frame

E. Class Diagram Desain

Class diagram desain merupakan sebuah diagram yang menggambarkan kelas-kelas yang berhubungan dengan komponen software yang digunakan untuk membangun aplikasi software. Diagram kelas ini berisi:

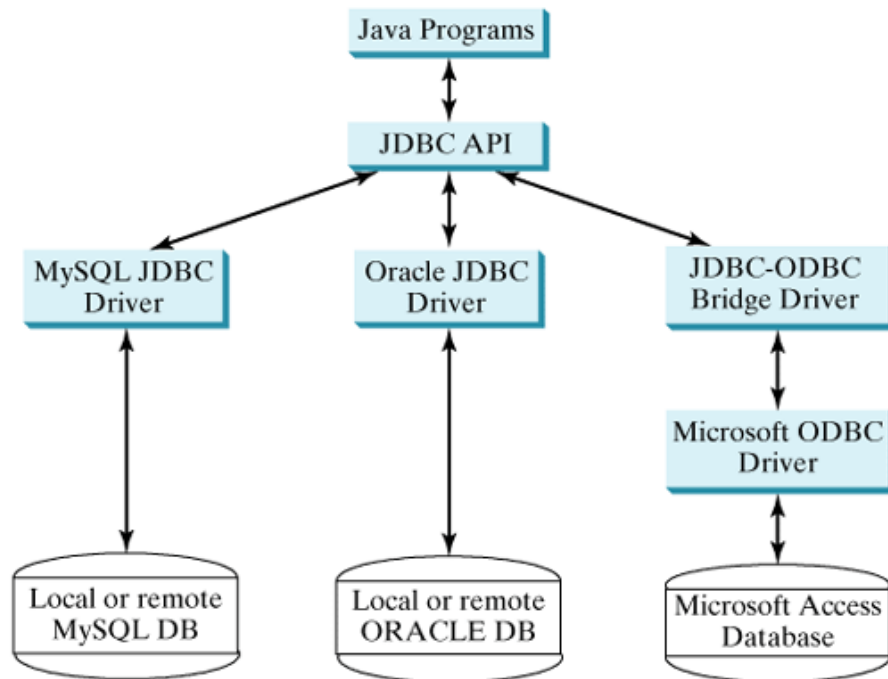
- Kelas.
- Relasi asosiasi, generalization/specialization, dan agregasi.
- Informasi atribut dan tipe atribut.
- Metode dengan parameter.
- *Navigability*.
- Ketergantungan (dependensi).

2.3. Java

Java merupakan suatu bahasa pemrograman yang bersifat object oriented, multiplatform, dan aman. Object oriented merupakan suatu metode pengembangan perangkat lunak dimana sebuah program merupakan sekelompok obyek yang bekerja bersama. Multiplatform berarti dapat dijalankan diberbagai macam sistem operasi jika mempunyai interpreter java yang dapat membaca bytecode.

2.3.1. JDBC

JDBC merupakan teknologi Java yang ditujukan untuk pengolahan data. JDBC ini merupakan Java API yang mendukung program Java untuk mengakses relational databases. Dengan menggunakan JDBC API, aplikasi yang menggunakan bahasa pemrograman Java dapat melakukan eksekusi perintah SQL, mendapatkan hasil, menampilkan data dengan tampilan yang *user-friendly* dan mengembalikan perubahan data ke database. *Gambar 2.1 Arsitektur JDBC dan MySQL* di bawah menunjukkan hubungan antara program Java, JDBC API dan JDBC Driver. JDBC Driver ini menghubungkan antara JDBC API dengan database seperti MySQL, Oracle dan Microsoft Access sehingga program Java dapat mengakses database tersebut.



Gambar 2.6 Arsitektur JDBC dan MySQL

Dalam pemrograman JDBC dikenal komponen-komponen sebagai berikut :

1. Driver

Interface ini menangani komunikasi dengan database server dan mengenkapsulasi proses internal dalam interaksi dengan database. Driver ini akan langsung berurusan dengan DriverManager.

2. DriverManager

Menggunakan object DriverManager untuk menangani objek Driver dimana objek DriverManager juga mengabstraksi detail dari proses kerja objek Driver.

3. Connection

Objek ini merepresentasikan koneksi fisik ke database. Kita dapat mengatur sifat result set dan operasi transaksi dengan object Connection ini.

4. Statement

Kita akan menggunakan objek dari interface ini untuk mengirimkan perintah SQL ke database. Interface turunan dari Statement memungkinkan untuk menerima parameter untuk mengeksekusi store procedure

5. ResultSet

Objek ini akan menyimpan data yang di dapat dari database setelah menjalankan query sql dengan menggunakan objek Statement. Objek ini bertindak sebagai iterator untuk menavigasi data.

6. SQLException

Objek ini merupakan objek turunan dari Exception yang sering digunakan untuk penanganan error dari pemrograman JDBC.

2.4. SQL (Structured Query Language)

SQL merupakan suatu bahasa yang digunakan untuk mengakses basis data. SQL dapat digunakan untuk menjelaskan struktur dari suatu data,

modifikasi data pada basis data dan menetapkan batasan keamanan. SQL mempunyai terbagi atas beberapa bagian, yaitu :

- Data-Definition Language(DDL) yang menyediakan perintah untuk menjelaskan relasi, menghapus relasi dan memodifikasi relasi. DDL menyediakan perintah-perintah seperti :

1. CREATE nama_objek
2. ALTER nama_objek
3. DROP nama_objek

- Data-Manipulation Language (DML) yang merupakan bahasa query berbasis relational algebra dan tuple relational calculus. DML menyediakan perintah-perintah seperti :

1. SELECT

Digunakan untuk membaca data dari basis data. Bentuk umum perintah ini adalah :

```
SELECT * | {[DISTINCT|DISTINCTROW] column | expression[alias], ...}
FROM table
[WHERE condition(s)] [GROUP BY condition(s)] [HAVING condition(s)]
[ORDER BY condition(s) [ASC|DEC]]
```

2. INSERT

Digunakan untuk menambahkan satu atau lebih data dari basis data. Bentuk umum perintah ini adalah :

```
INSERT INTO table (column1, column2, [columnN]) VALUES (value1,
value2, [valueN])
```

3. UPDATE

Digunakan untuk mengubah data pada satu atau lebih baris data pada tabel. Bentuk umum perintah ini adalah :

```
UPDATE table SET column1 = value1, column2 = value2, [columnN = valueN]
[WHERE id_column = value]
```

4. DELETE

Digunakan untuk menghapus satu atau lebih data dari suatu tabel.

Bentuk umum perintah ini adalah :

```
DELETE FROM tablename [where field1 = value1 [AND | OR] field2 = value2
[AND | OR] fieldN = valueN]
```

- View-Definition yang merupakan bagian dari DDL yang menyediakan perintah view untuk melihat data dari satu table atau lebih.
- Transaction control yang menyediakan perintah untuk memulai dan mengakhiri transaksi.
- Embedded SQL yang menjelaskan di mana perintah SQL dapat diintegrasikan ke dalam bahasa pemrograman seperti C, C++, Java, Cobol, Pascal dan lain-lain.
- Integrity yang merupakan bagian dari DDL yang menyediakan perintah untuk menspesifikasi integritas data yang masuk ke basis data.
- Authorization yang merupakan bagian dari DDL yang menyediakan perintah untuk menspesifikasi aturan akses.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Sistem yang Ada Saat Ini

Obat yang tersedia di Apotek Joint Farma diperoleh dari proses pembelian obat dari beberapa distributor. Satu macam obat dapat diperoleh dari distributor yang berbeda-beda sehingga perlu dicatat data distributor tersebut. Selain proses pembelian, ada juga proses penjualan obat kepada pasien atau pembeli. Obat yang dijual ada beberapa macam, antara lain: obat dengan resep dokter, obat psikotropika obat narkotika, obat wajib apotek, dan obat bebas. Setelah membeli, pegawai apotek akan memberikan nota sebagai bukti pembelian. Untuk mengatasi masalah obat yang hampir kadaluarsa, pemilik atau pegawai apotek mencari obat-obatan tersebut satu per satu dari buku yang mencatat data obat dan distributor.

Setiap akhir bulan baik pemilik maupun pegawai apotek bersama-sama membuat laporan-laporan yang diperlukan, antara lain laporan penggunaan psikotropika dan narkotika, laporan penjualan, laporan pembelian, dan lain sebagainya. Semua kegiatan tersebut di atas dilakukan secara manual tanpa bantuan komputer.

3.2. Sistem yang Akan Dibangun

Sistem Informasi Pengelolaan Stok Obat pada Apotek Joint Farma dapat memudahkan Apotek Joint Farma dalam memperoleh informasi dengan cepat dan tepat. Dalam proses pembelian obat, pemilik atau pegawai perlu mencatat data obat yang dibeli dan data distributor. Untuk mengatasi masalah obat yang hampir kadaluarsa, pemilik atau pegawai apotek hanya perlu menjalankan Sistem Informasi Pengelolaan Stok Obat dan langsung ada peringatan otomatis dari sistem. Kemudian pemilik atau pegawai apotek mengisi data obat tersebut pada salah satu *form* yang tersedia dan mencetak retur pembeliannya.

Dalam proses penjualan obat, seorang pembeli datang ke apotek, kemudian memberitahukan kepada pegawai apotek obat apa yang ingin dibeli. Setelah itu pegawai hanya perlu menginputkan data tersebut dan mencetak nota untuk pembeli. Untuk proses pembuatan laporan bulanan, pemilik atau pegawai yang bertindak sebagai admin apotek tidak perlu mencatat data obat, data pembelian, data penjualan, dan lain sebagainya. Pemilik atau pegawai yang bertindak sebagai admin apotek hanya perlu memilih menu untuk mencetak laporan yang diperlukan dan sistem akan langsung mencetak laporan.

3.3. Hardware dan Software Untuk Membuat Sistem

- Prosesor Intel Core 2 Duo
- RAM 2 GB

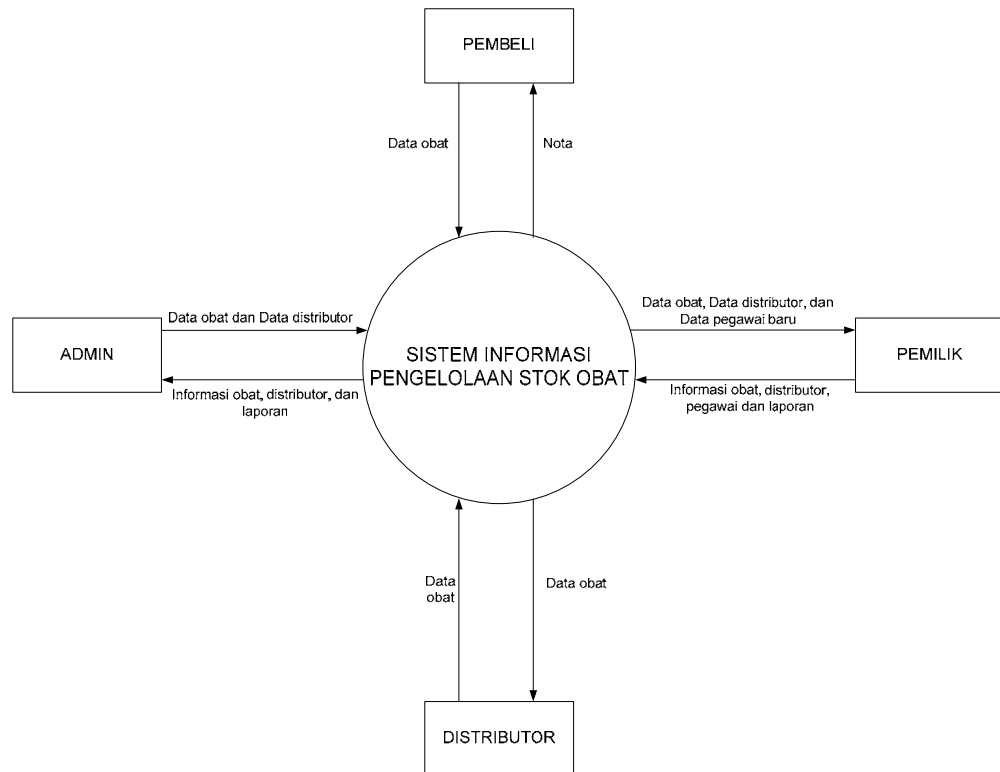
- VGA 128
- HD 160 GB
- MySQL Server 5.0
- SQL yog 5.12
- NetBeans 6.1
- JDK 1.6
- iReport

3.4. Hardware dan Software Untuk Menjalankan Sistem

- Intel Pentium 4
- RAM 512 GB sampai 1 GB
- VGA 128
- HD 200 MB
- MySQL Server 5.0
- JDK 1.6

3.5. Diagram Konteks

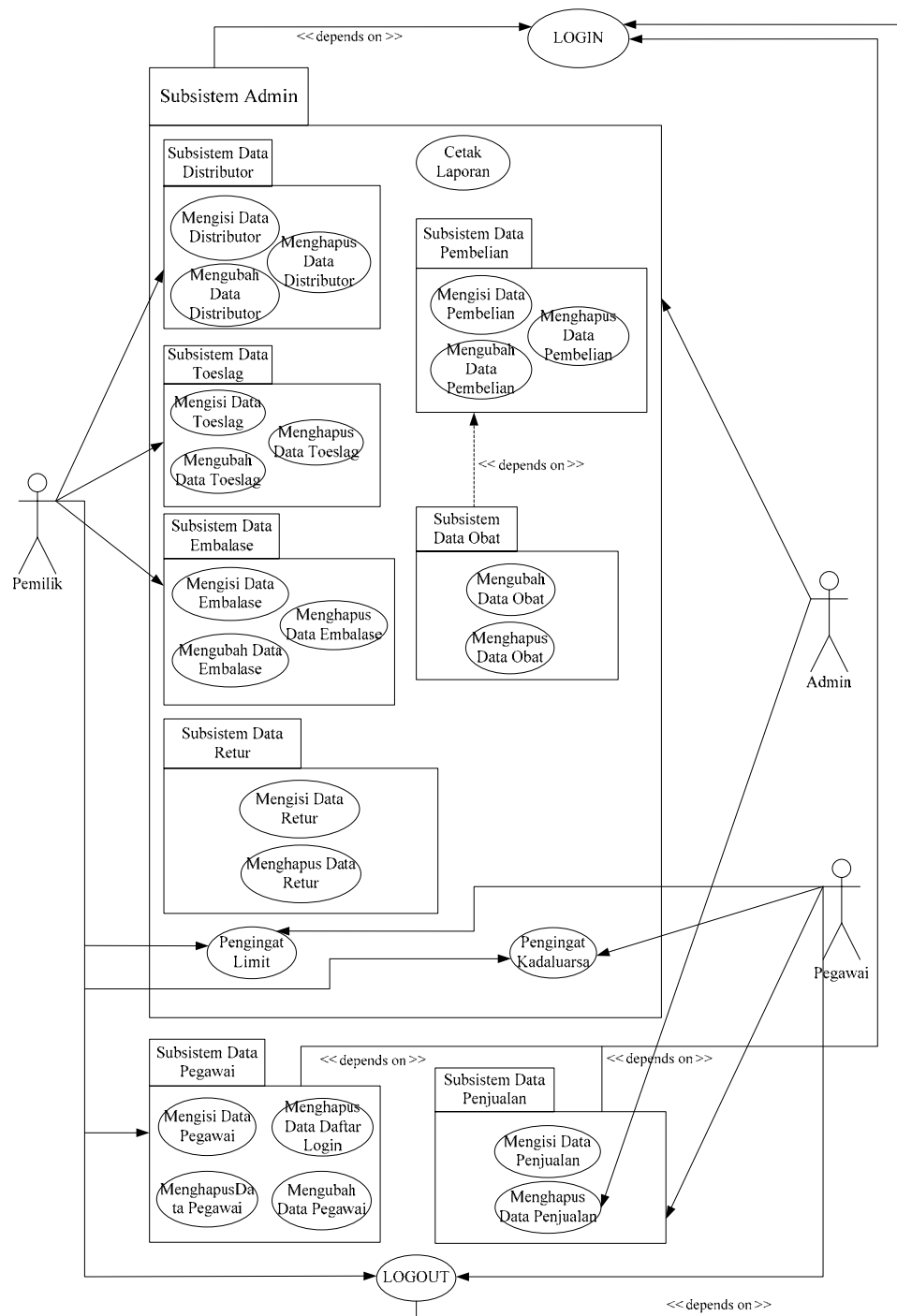
Berikut ini merupakan diagram konteks sistem.



Gambar 3.1 Diagram Konteks

3.6. Diagram Use Case

Berikut ini merupakan diagram use case sistem.



Gambar 3.2 Diagram Use-Case

3.6.1. Ringkasan Use Case

Ringkasan use-case sistem yang memuat use-case, deskripsi dan pelaku yang berpartisipasi dapat dilihat pada tabel di bawah ini:

Tabel 3.1 Ringkasan Use Case

No	Nama Use-Case	Deskripsi Use-Case	Pelaku yang berpartisipasi
1	Log in	Use case ini menggambarkan proses untuk masuk ke sistem.	Pemilik, Pegawai, dan Admin (<i>primary business</i>)
2	Mengisi Data Pegawai	Use case ini menggambarkan proses untuk memasukkan data pegawai.	Pemilik (<i>primary business</i>)
3	Mengubah Data Pegawai	Use case ini menggambarkan proses pengubahan data pegawai.	Pemilik (<i>primary bussines</i>)
4	Menghapus Data Pegawai	Use case ini menggambarkan proses untuk menghapus data pegawai yang sudah tidak diperlukan.	Pemilik (<i>primary business</i>)
5	Menghapus Data Daftar Login	Use case ini menggambarkan proses untuk menghapus data login pegawai dan admin yang	Pemilik (<i>primary business</i>)

		sudah tidak diperlukan.	
6	Mengisi Data Penjualan	Use case ini menggambarkan proses untuk memasukkan data obat yang dijual kepada pembeli.	Pegawai (<i>primary business</i>)
7	Menghapus Data Penjualan	Use case ini menggambarkan proses untuk menghapus data penjualan yang sudah tidak diperlukan.	Pemilik dan Admin (<i>primary business</i>)
8	Mengisi Data Pembelian	Use case ini menggambarkan proses pencatatan data obat yang dibeli dari distributor.	Admin (<i>primary business</i>)
9	Mengubah Data Pembelian	Use case ini menggambarkan proses pengubahan data pembelian. Proses ini dapat berupa penambahan data obat ataupun pengubahan detail data pembelian	Admin (<i>primary business</i>)
10	Menghapus Data Pembelian	Use case ini menggambarkan proses untuk menghapus data pembelian yang sudah tidak diperlukan.	Admin (<i>primary business</i>)

11	Mengubah Data Obat	Use case ini menggambarkan proses pengubahan data obat.	Admin (<i>primary business</i>)
12	Menghapus Data Obat	Use case ini menggambarkan proses untuk menghapus data obat yang tidak diperlukan.	Admin (<i>primary business</i>)
13	Mengisi Data Distributor	Use case ini menggambarkan proses untuk memasukkan data distributor.	Pemilik dan Admin (<i>primary business</i>)
14	Mengubah Data Distributor	Use case ini menggambarkan proses pengubahan data distributor.	Pemilik dan Admin (<i>primary business</i>)
15	Menghapus Data Distributor	Use case ini menggambarkan proses untuk menghapus data distributor yang tidak diperlukan.	Pemilik dan Admin (<i>primary business</i>)
16	Cetak Laporan	Use case ini menggambarkan proses untuk mencetak laporan yang diperlukan.	Admin (<i>primary business</i>)

17	Mengisi Data Retur	Use case ini menggambarkan proses memasukkan data retur.	Admin (<i>primary business</i>)
18	Menghapus Data Retur	Use case ini menggambarkan proses untuk menghapus data retur yang tidak diperlukan.	Admin (<i>primary business</i>)
19	Mengisi Data Embalase	Use case ini menggambarkan proses memasukkan data embalase.	Pemilik dan Admin (<i>primary business</i>)
20	Mengubah Data Embalase	Use case ini menggambarkan proses pengubahan data embalase.	Pemilik dan Admin (<i>primary business</i>)
21	Menghapus Data Embalase	Use case ini menggambarkan proses untuk menghapus data toeslag yang tidak diperlukan.	Pemilik dan Admin (<i>primary business</i>)
22	Mengisi Data Toeslag	Use case ini menggambarkan proses memasukkan data toeslag.	Pemilik dan Admin (<i>primary business</i>)
23	Mengubah Data Toeslag	Use case ini menggambarkan proses pengubahan data toeslag.	Pemilik dan Admin (<i>primary business</i>)
24	Menghapus Data Toeslag	Use case ini menggambarkan proses untuk menghapus data	Pemilik dan Admin (<i>primary business</i>)

		toeslag yang tidak diperlukan.	
25	Pengingat Kadaluarsa	Use case ini menggambarkan proses yang berjalan secara otomatis untuk mengingatkan tanggal kadaluarsa obat.	Pemilik, Pegawai, dan Admin (<i>primary business</i>)
26	Pengingat Limit	Use case ini menggambarkan proses yang berjalan secara otomatis untuk mengingatkan stok obat yang sudah sama dengan limit atau kurang dari limit obat.	Pemilik, Pegawai, dan Admin (<i>primary business</i>)
27	LOG OUT	Use case ini menggambarkan proses untuk keluar sistem.	Pemilik, Pegawai, dan Admin (<i>primary business</i>)

3.6.2. Narasi Use Case

Narasi Use-Case Log in dapat dilihat pada tabel berikut ini:

Tabel 3.2 Narasi Use-Case Log in

Author : Olivia Dian Kusumawati

Date : 21 November 2008

Version : 1.0

Use-case Name:	LOG IN	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>	
Use-case ID:	UC-001		
Priority :	High		
Source:	-		
Primary Business Actor:	Pemilik, admin, dan pegawai.		
Other	-		

Participating Actor:		
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses untuk masuk ke sistem. Use case ini digunakan untuk menjaga agar fasilitas insert data pegawai hanya dapat dilakukan oleh seorang pemilik.	
Precondition:	Pemilik, admin, dan pegawai telah memiliki username dan password.	
Trigger:	Use case ini digunakan apabila pemilik atau admin atau pegawai ingin masuk ke dalam sistem.	
Typical Course of event:	Actor Action	System Response
	<p>Step 1: Pemilik atau admin atau pegawai menjalankan sistem.</p> <p>Step 3: Pemilik atau admin atau pegawai memasukkan username, password, dan status.</p> <p>Step 4: Pemilik atau admin atau pegawai mengklik tombol OK.</p>	<p>Step 2: Sistem meminta untuk memasukkan username, password, dan status.</p> <p>Step 5: Sistem mengecek validasi di database.</p> <p>Step 6: Sistem masuk ke halaman utama untuk pemilik atau admin atau pegawai.</p>
Alternate Course:	<p>Alt-Step 4: Pemilik atau admin atau pegawai mengklik tombol BATAL, sehingga sistem tidak jadi masuk ke halaman utama.</p> <p>Alt-Step 5: Jika username, password, dan status yang dimasukkan tidak sesuai maka sistem akan memberikan pesan kegagalan dan secara otomatis kembali ke menu LOG IN.</p>	
Conclusion:	Use case ini berhenti apabila pemilik atau admin atau pegawai telah berhasil masuk ke dalam halaman utama.	
Postcondition:	<ul style="list-style-type: none"> • Pemilik atau admin atau pegawai berhasil login dan masuk ke halaman utama. • Pemilik atau admin atau pegawai tidak berhasil masuk dan kembali ke menu login. 	
Business Rules:	Pemilik atau admin atau pegawai harus memasukan username, password, dan status dengan benar.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	

Assumptions:	-
Open issues:	Karena tidak hanya satu orang saja yang mengakses data-data yang ada dikhawatirkan data yang harusnya tidak boleh dilihat oleh orang yang tidak memiliki wewenang pada akhirnya juga dapat dilihat.

Narasi Use-Case Mengisi Data Pegawai dapat dilihat pada tabel berikut:

Tabel 3.3 Narasi Use-Case Mengisi Data Pegawai

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengisi Data Pegawai	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-002	
Priority :	High	
Source:	Data kepegawaian.	
Primary Business Actor:	Pemilik.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penginputan data pegawai yang dilakukan oleh pemilik apabila ada pegawai baru.	
Precondition:	Pemilik telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada pegawai baru.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik memilih menu setup pegawai.	Step 2: Sistem menampilkan form untuk menambah data pegawai.
	Step 3: Pemilik mengklik tombol TAMBAH.	
	Step 4: Pemilik mengisikan data username, password, dan status untuk pegawai baru.	
	Step 5: Pemilik mengklik tombol SIMPAN.	
	Step 6: Sistem menambahkan data pegawai baru ke dalam database.	

		Step 7: Sistem akan menampilkan pesan keberhasilan.
Alternate Course:	<p>Alt-Step 5: Pemilik mengklik tombol BATAL, sehingga sistem tidak jadi menambahkan data pegawai yang baru ke dalam database dan akan kembali ke form untuk menambah data pegawai.</p> <p>Alt-Step 7: Jika pemilik tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada pegawai baru.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pemilik untuk menginputkan data.	
Assumptions:	-	
Open issues:	Di Apotek Joint Farma setiap pegawai menggunakan sistem shift, sehingga pemilik merasa perlu untuk membuat log file untuk mencatat identitas pegawai dan waktu mereka melakukan proses login. Oleh karena itu setiap pegawai harus memiliki username yang berbeda.	


Narasi Use-Case Mengubah Data Pegawai dapat dilihat pada tabel berikut:

Tabel 3.4 Narasi Use-Case Mengubah Data Pegawai

Author : Olivia Dian Kusumawati

Date : 21 November 2008

Version : 1.0

Use-case Name:	Mengubah Data Pegawai	Jenis Use-Case Business Requirements: 	
Use-case ID:	UC-003		
Priority :	Medium		
Source:	Data kepegawaian.		
Primary Business Actor:	Pemilik.		
Other Participating Actor:	-		
Other Interested Stakeholder:	-		
Decription:	Use case ini menggambarkan proses pengubahan data		

	pegawai yang dilakukan oleh pemilik apabila ada data pegawai yang salah.	
Precondition:	Pemilik telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data pegawai yang salah.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik memilih menu setup pegawai.	Step 2: Sistem menampilkan form untuk mengubah data pegawai.
	Step 3: Pemilik memilih data yang akan diganti.	Step 4: Sistem akan menampilkan data yang dipilih pada teksfield yang tersedia.
	Step 5: Pemilik menggantikan data lama dengan yang baru.	
	Step 6: Pemilik mengklik tombol SIMPAN.	Step 7: Sistem akan menyimpan data tersebut ke database.
		Step 8: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 6: Pemilik mengklik tombol BATAL, sehingga sistem tidak jadi menyimpan data baru ke database dan akan kembali ke form untuk menambah data pegawai. Alt-Step 7: Jika pemilik tidak berhasil melakukan proses perubahan data, maka sistem akan menampilkan peringatan.	
Conclusion:	Use case ini berhenti apabila tidak ada data pegawai yang ingin diubah.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pemilik untuk menginputkan data.	
Assumptions:	-	
Open issues:	Di Apotek Joint Farma setiap pegawai mempunyai username dan password yang berbeda, sehingga apabila ada pegawai tersebut lupa dengan password ato username maka pemelik hanya perlu melakukan proses perubahan data pegawai ini.	

Narasi Use-Case Menghapus Data Pegawai dapat dilihat pada tabel berikut:

Tabel 3.5 Narasi Use-Case Menghapus Data Pegawai

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Data Pegawai	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-004	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Pemilik.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Data yang dihapus adalah username dan password.	
Precondition:	Pemilik telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada pegawai yang keluar.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik memilih menu setup pegawai. Step 3: Pemilik memilih data pegawai yang ingin dihapus. Step 4: Pemilik mengklik tombol HAPUS.	Step 2: Sistem menampilkan form untuk menghapus data pegawai. Step 5: Sistem menghapus data. Step 6: Sistem akan menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 4: Pemilik mengklik tombol BATAL, sehingga sistem tidak jadi menghapus data pegawai dan kembali ke form untuk menghapus data pegawai. Alt-Step 6: Jika pemilik tidak berhasil melakukan proses penghapusan data, maka sistem akan menampilkan peringatan.	
Conclusion:	Use case ini berhenti apabila tidak ada pegawai yang keluar.	
Postcondition:	<ul style="list-style-type: none">• Data berhasil dihapus dari database.• Data tidak berhasil dihapus dan tetap berada di dalam	

	database.
Business Rules:	Pemilik harus memilih data yang sesuai.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	Karena ada pegawai yang keluar, sehingga data tentang pegawai yang keluar ini tidak diperlukan lagi.

Narasi Use-Case Menghapus Daftar Login dapat dilihat pada tabel berikut:

Tabel 3.6 Narasi Use-Case Menghapus Daftar Login

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Daftar Login	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-005	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Pemilik.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Data yang dihapus adalah username dan waktu login.	
Precondition:	Pemilik telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada pegawai yang keluar.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik memilih menu hapus daftar login.	Step 2: Sistem menampilkan form untuk menghapus data daftar login.
	Step 3: Pemilik mengklik tombol HAPUS.	Step 4: Sistem menghapus data. Step 5: Sistem akan menampilkan pesan keberhasilan.
Alternate	Alt-Step 3: Pemilik mengklik tombol BATAL, sehingga	

Course:	sistem tidak jadi menghapus data daftar login dan kembali ke form untuk menghapus data daftar login. Alt-Step 5: Jika pemilik tidak berhasil melakukan proses penghapusan data, maka sistem akan menampilkan peringatan.
Conclusion:	Use case ini berhenti apabila tidak ada data yang ingin dihapus.
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dan tetap berada di dalam database.
Business Rules:	Pemilik harus memilih data yang sesuai.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	Karena data login dan log out yang tersimpan dalam jangka waktu tertentu akan bertambah banyak sehingga perlu untuk dihapus.

Narasi Use-Case Mengisi Data Penjualan dapat dilihat pada tabel berikut:

Tabel 3.7 Narasi Use-Case Mengisi Data Penjualan

Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0

Use-case Name:	Mengisi Data Penjualan	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>	
Use-case ID:	UC-006		
Priority :	High		
Source:	Pesanan dan Resep		
Primary Business Actor:	Pegawai.		
Other Participating Actor:	-		
Other Interested Stakeholder:	-		
Decription:	Use case ini menggambarkan proses penginputan data penjualan obat yang dilakukan oleh pegawai apabila terjadi proses pembelian obat oleh pasien atau pembeli.		
Precondition:	Pegawai telah melakukan proses login.		
Trigger:	Use case ini digunakan apabila ada obat yang baru dibeli oleh pasien atau pembeli.		
Typical Course	Actor Action	System Response	

of event:	<p>Step 1: Pegawai memilih menu penjualan.</p> <p>Step 3: Pegawai mengisikan data obat yang ingin diinputkan.</p> <p>Step 4: Pegawai mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk mengisi data obat yang akan dijual.</p> <p>Step 5: Sistem menambahkan data penjualan obat ke dalam database.</p> <p>Step 6: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	Alt-Step 6: Jika pegawai tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.	
Conclusion:	Use case ini berhenti apabila tidak ada data penjualan obat yang harus diinputkan.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pegawai untuk menginputkan data.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Menghapus Data Penjualan dapat dilihat pada tabel berikut:

Tabel 3.8 Narasi Use-Case Menghapus Data Penjualan

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0


Use-case Name:	Menghapus Data Penjualan	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-007	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Admin.	
Other Participating	-	

Actor:		
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Data yang dihapus adalah data penjualan berupa kode penjualan dan tanggal penjualan.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data penjualan yang tidak diperlukan lagi.	
Typical Course of event:	Actor Action	System Response
	<p>Step 1: Admin memilih menu setup penjualan.</p> <p>Step 3: Admin memilih data penjualan yang ingin dihapus.</p> <p>Step 4: Admin mengklik tombol HAPUS.</p>	<p>Step 2: Sistem menampilkan form untuk menghapus data penjualan.</p> <p>Step 5: Sistem menghapus data.</p> <p>Step 6: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 4: admin mengklik tombol BATAL, sehingga sistem tidak jadi menghapus data penjualan dan kembali ke form untuk menghapus data penjualan.</p> <p>Alt-Step 6: Jika admin tidak berhasil melakukan proses penghapusan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data penjualan yang ingin dihapus.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dan tetap berada di dalam database. 	
Business Rules:	Admin harus memilih data yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Setelah pencetakan laporan penjualan per bulan ada data penjualan yang mungkin sudah tidak diperlukan lagi.	

Narasi Use-Case Mengisi Data Pembelian dapat dilihat pada tabel berikut:

Tabel 3.9 Narasi Use-Case Mengisi Data Pembelian

Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0


Use-case Name:	Mengisi Data Pembelian	Jenis Use-Case Business Requirements: 
Use-case ID:	UC-008	
Priority :	High	
Source:	Faktur Pembelian.	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penginputan data pembelian.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada obat yang baru dibeli dari distributor.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin memilih menu pembelian.	Step 2: Sistem menampilkan form untuk mengisi data pembelian.
	Step 3: Admin mengklik tombol TAMBAH.	
	Step 4: Admin mengisikan data pembelian yang ingin diinputkan.	
	Step 5: Admin mengklik tombol SIMPAN.	Step 6: Sistem menambahkan data pembelian ke dalam database.
		Step 7: Sistem akan menampilkan pesan keberhasilan.
	Step 8: Admin mengklik	Step 9: Sistem akan

	<p>tombol Detail.</p> <p>Step 10: Admin mengklik tombol TAMBAH.</p> <p>Step 11: Admin mengisikan data obat yang ingin diinputkan.</p> <p>Step 12: Admin mengklik tombol SIMPAN.</p>	<p>menampilkan form untuk mengisi data obat-obatan yabg baru saja dibeli dari distributor.</p> <p>Step 13: Sistem menambahkan data obat ke dalam database.</p> <p>Step 14: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 5: Jika admin mengklik tombol BATAL, maka sistem tidak jadi menambahkan data pembelian ke database.</p> <p>Alt-Step 7: Jika admin tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p> <p>Alt-Step 12: Jika admin mengklik tombol BATAL, maka sistem tidak jadi menambahkan data obat ke database.</p> <p>Alt-Step 14: Jika admin tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data pembelian obat dari distributor.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Admin harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah Admin untuk menginputkan data.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Mengubah Data Pembelian dapat dilihat pada tabel berikut:

Tabel 3.10 Narasi Use-Case Mengubah Data Pembelian

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0


Use-case Name:	Mengubah Data Pembelian	Jenis Use-Case Business Requirements: 
Use-case ID:	UC-009	
Priority :	Medium	
Source:	Faktur Pembelian.	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pengubahan data pembelian yang dilakukan oleh admin apabila ada data pembelian yang salah.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data pembelian yang salah.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin memilih menu pembelian.	Step 2: Sistem menampilkan form untuk mengubah data pembelian.
	Step 3: Admin memilih data yang akan diganti.	Step 4: Sistem akan menampilkan data yang dipilih pada teksfield yang tersedia.
	Step 5: Admin menggantikan data lama dengan yang baru.	
	Step 6: Admin mengklik tombol SIMPAN.	Step 7: Sistem akan menyimpan data tersebut ke database.
		Step 8: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 6: Admin mengklik tombol BATAL, sehingga sistem tidak jadi menyimpan data baru ke database dan akan kembali ke form untuk menambah data pembelian.	

	Alt-Step 7: Jika admin tidak berhasil melakukan proses pengubahann data, maka sistem akan menampilkan peringatan.
Conclusion:	Use case ini berhenti apabila tidak ada data pembelian yang ingin diubah.
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database.
Business Rules:	Admin harus memasukan data dengan tipe yang sesuai.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah admin untuk menginputkan data.
Assumptions:	-
Open issues:	-

Narasi Use-Case Menghapus Data Pembelian dapat dilihat pada tabel berikut:

Tabel 3.11 Narasi Use-Case Menghapus Data Pembelian

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Data Pembelian	Jenis Use-Case Business Requirements: 	
Use-case ID:	UC-010		
Priority :	Low		
Source:	Data yang sudah tersimpan.		
Primary Business Actor:	Admin.		
Other Participating Actor:	-		
Other Interested Stakeholder:	-		
Decription:	Use case ini menggambarkan proses penghapusan data pembelian.		
Precondition:	Admin telah melakukan proses login.		
Trigger:	Use case ini digunakan apabila ada data pembelian yang tidak diperlukan lagi.		
Typical Course	Actor Action	System Response	

of event:	<p>Step 1: Admin memilih menu pembelian.</p> <p>Step 3: Admin memilih data pembelian yang ingin dihapus.</p> <p>Step 4: Admin mengklik tombol HAPUS.</p>	<p>Step 2: Sistem menampilkan form untuk menghapus data pembelian.</p> <p>Step 5: Sistem menghapus data.</p> <p>Step 6: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 4: Admin mengklik tombol BATAL, sehingga sistem tidak jadi menghapus data pembelian dan kembali ke form untuk menghapus data pembelian.</p> <p>Alt-Step 6: Jika admin tidak berhasil melakukan proses penghapusan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data pembelian yang ingin dihapus.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dan tetap berada di dalam database. 	
Business Rules:	Admin harus memilih data yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Mengubah Data Obat dapat dilihat pada tabel berikut:

Tabel 3.12 Narasi Use-Case Mengubah Data Obat

Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0

Use-case Name:	Mengubah Data Obat	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-011	
Priority :	Medium	
Source:	Daftar harga obat-obatan.	
Primary Business Actor:	Admin.	
Other	-	

Participating Actor:		
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pengubahan data obat yang dilakukan oleh admin apabila terjadi perubahan harga obat-obatan.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada obat yang berubah harganya.	
Typical Course of event:	Actor Action	System Response
	<p>Step 1: Admin memilih menu setup obat.</p> <p>Step 3: Admin memilih kategori dan memasukan kata kunci yang dicari.</p> <p>Step 6: Admin mengklik data yang ditemukan.</p> <p>Step 8: Admin mengganti data lama dengan data yang baru.</p> <p>Step 9: Admin mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk merubah data obat.</p> <p>Step 4: Sistem mencari kata kunci tersebut ke dalam database.</p> <p>Step 5: Sistem menampilkan data yang ditemukan.</p> <p>Step 7: Data yang diklik muncul pada setiap textfield yang tersedia.</p> <p>Step 10: Sistem menyimpan data yang baru tersebut ke dalam database.</p> <p>Step 11: Sistem menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 9: Jika admin mengklik tombol BATAL, maka sistem akan kembali ke form untuk merubah data.</p> <p>Alt-Step 11: Sistem akan menampilkan peringatan apabila sistem tidak dapat merubah data lama dengan data yang baru.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data obat yang harus dirubah.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil diubah dan disimpan ke database. • Data tidak berhasil diubah. 	

Business Rules:	Admin harus memasukan data dengan tipe yang sesuai.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	Data obat yang ada di Apotek Joint Farma jumlahnya banyak sekitar 500 sampai 1000 obat, sehingga apabila dirubah satu per satu secara manual maka proses pencetakan laporan dapat dilaksanakan setelah proses perubahan data obat secara manual selesai dilaksanakan.

Narasi Use-Case Menghapus Data Obat dapat dilihat pada tabel berikut:

Tabel 3.13 Narasi Use-Case Menghapus Data Obat

Author : Olivia Dian Kusumawati **Date** : 21 November 2008
Version : 1.0

Use-case Name:	Menghapus Data Obat	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-012	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Penghapusan data dilakukan apabila ada obat yang sudah tidak diperlukan lagi karena tidak distok lagi oleh distributor.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada obat yang tidak di stok lagi.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin memilih menu setup obat. Step 3: Admin memilih kategori dan memasukan kata	Step 2: Sistem menampilkan form untuk menghapus data obat. Step 4: Sistem mencari kata kunci tersebut ke dalam

	<p>kunci yang dicari.</p> <p>Step 6: Admin mengklik data yang ditemukan.</p> <p>Step 8: Admin mengklik tombol HAPUS.</p>	<p>database.</p> <p>Step 5: Sistem menampilkan data yang ditemukan.</p> <p>Step 7: Data yang diklik muncul pada setiap textfield yang tersedia.</p> <p>Step 9: Sistem menghapus data.</p> <p>Step 10: Sistem menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 8: Jika admin mengklik tombol BATAL, maka sistem akan kembali ke form untuk menghapus data.</p> <p>Alt-Step 10: Sistem akan menampilkan peringatan apabila sistem tidak dapat menghapus.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data obat yang harus dihapus.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dari database. 	
Business Rules:	Admin harus memastikan bahwa data yang ingin dihapus adalah data yang tidak diperlukan lagi oleh sistem.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Pencarian data yang lama dan tidak terlalu akurat membuat pemilik atau pegawai harus berulang-ulang kali melihat buku-buku yang mencatat data obat untuk memastikan data obat tersebut memang data yang ingin dihapus.	

Narasi Use-Case Mengisi Data Distributor dapat dilihat pada tabel berikut:

Tabel 3.14 Narasi Use-Case Mengisi Data Distributor

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengisi Data Distributor	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-013	
Priority :	High	

Source:	Kartu Nama	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penginputan data distributor yang dilakukan oleh pemilik atau admin.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada distributor baru.	
Typical Course of event:	Actor Action	System Response
	<p>Step 1: Pemilik / admin memilih menu setup distributor.</p> <p>Step 3: Pemilik / admin mengklik tombol TAMBAH.</p> <p>Step 4: Pemilik / admin mengisikan data distributor yang ingin diinputkan.</p> <p>Step 5: Pemilik / admin mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk menambah data distributor.</p> <p>Step 6: Sistem menambahkan data distributor ke dalam database.</p> <p>Step 7: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 5: Pemilik / admin mengklik tombol BATAL, sehingga sistem tidak jadi menambahkan data distributor ke dalam database dan akan kembali ke form untuk menambah data distributor.</p> <p>Alt-Step 7: Jika pemilik / admin tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data distributor yang harus diinputkan.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik harus memasukan data dengan tipe yang sesuai.	

Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pemilik / admin untuk menginputkan data.
Assumptions:	-
Open issues:	Satu macam obat dapat berasal dari beberapa distributor yang berbeda. Jika pemilik ingin meretur obat yang hampir kadaluarsa maka pemilik membutuhkan data distributor obat tersebut.

Narasi Use-Case Mengubah Data Distributor dapat dilihat pada tabel berikut:

Tabel 3.15 Narasi Use-Case Mengubah Data Distributor

Author : Olivia Dian Kusumawati **Date : 21 November 2008**

Version : 1.0


Use-case Name:	Mengubah Data Distributor	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-014	
Priority :	Medium	
Source:	Kartu Nama.	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pengubahan data distributor yang dilakukan oleh pemilik atau admin apabila terjadi perubahan alamat, no telepon, dan lain sebagainya.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada distributor yang berubah identitsnya.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin memilih menu setup distributor.	Step 2: Sistem menampilkan form untuk merubah data distributor.
	Step 3: Pemilik / admin mengganti data lama dengan data yang baru.	
	Step 4: Pemilik / admin	Step 5: Sistem menyimpan

	mengklik tombol SIMPAN.	data yang baru tersebut ke dalam database. Step 6: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 4: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk merubah data distributor. Alt-Step 6: Sistem akan menampilkan peringatan apabila sistem tidak dapat merubah data lama dengan data yang baru.	
Conclusion:	Use case ini berhenti apabila tidak ada data distributor yang harus dirubah.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil diubah dan disimpan ke database. • Data tidak berhasil diubah. 	
Business Rules:	Pemilik / admin harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Distributor yang menyetok obat terkadang senang berpindah-pindah tempat tinggal.	

Narasi Use-Case Menghapus Data Distributor dapat dilihat pada tabel berikut:

Tabel 3.16 Narasi Use-Case Menghapus Data Distributor

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Data Distributor	Jenis Use-Case Business Requirements: 
Use-case ID:	UC-015	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	

Decription:	Use case ini menggambarkan proses penghapusan data. Penghapusan data dilakukan apabila data yang sudah tidak diperlukan lagi.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila distributor tidak menyetok obat lagi.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin memilih menu setup distributor.	Step 2: Sistem menampilkan form untuk menghapus data distributor.
	Step 3: Pemilik / admin mengklik data yang ingin dihapus.	Step 4: Data yang diklik muncul pada setiap textfield yang tersedia.
	Step 5: Pemilik / admin mengklik tombol HAPUS.	Step 6: Sistem menghapus data. Step 7: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 5: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk menghapus data distributor. Alt-Step 7: Sistem akan menampilkan peringatan apabila sistem tidak dapat menghapus.	
Conclusion:	Use case ini berhenti apabila tidak ada data distributor yang harus dihapus.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dari database. 	
Business Rules:	Pemilik / admin harus memastikan bahwa data yang ingin dihapus adalah data yang tidak diperlukan lagi oleh sistem.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Distributor obat tersebut sudah tidak menjadi distributor untuk Apotek Joint Farma atau sebaliknya.	

Narasi Use-Case Cetak Laporan dapat dilihat pada tabel berikut:

Tabel 3.17 Narasi Use-Case Cetak Laporan

Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0

Use-case Name:	Cetak Laporan	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-016	
Priority :	Medium	
Source:	-	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pencetakan laporan. Laporan yang di cetak adalah pembelian, penjualan obat psikotropika, penjualan obat narkotika, dan penjualan obat wajib apotek.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila sudah akhir bulan dan adanya obat yang hampir kadaluarsa.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin memilih menu Laporan.	Step 3: Sistem akan menampilkan preview laporan. Step 5: Sistem mengeprint laporan.
	Step 2: Admin memilih submenu berdasarkan jenis laporan yang ingin dicetak.	
	Step 4: Admin mengklik simbol print.	
Alternate Course:	Alt-Step 4: Admin tidak mengklik simbol print, sehingga sistem akan menampilkan preview laporan.	
Conclusion:	Use case ini berhenti apabila tidak ada laporan yang harus dicetak.	
Postcondition:	<ul style="list-style-type: none">• Laporan berhasil dicetak.• Laporan tidak berhasil diceetak.	
Business Rules:	Admin harus memastikan bahwa submenu yang dipilih sesuai dengan laporan yang ingin dicetak.	
Implementation Constrains and	Tampilan sistem berupa GUI dan preview laporan dalam format pdf.	

Specifications:	
Assumptions:	-
Open issues:	Karena laporan yang dicetak setiap akhir bulannya dalam jumlah yang banyak, sehingga keakuratannya menjadi tidak terjamin.

Narasi Use-Case Mengisi Data Retur dapat dilihat pada tabel berikut:

Tabel 3.18 Narasi Use-Case Mengisi Data Retur

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Version : 1.0		
Use-case Name:	Mengisi Data Retur	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-017	
Priority :	High	
Source:	-	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	Distributor	
Decription:	Use case ini menggambarkan proses penginputan data obat yang hampir kadaluarsa, hilang, ataupun rusak. Obat-obat yang hapir kadaluarsa ini diretur biasanya enam bulan sebelum tanggal kadaluarsa.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada obat yang hampir kadaluarsa, hilang, ataupun rusak.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin mengklik menu retur.	Step 2: Sistem menampilkan form untuk mengisi obat-obat yang harus diretur.
	Step 3: Admin mengklik tombol TAMBAH.	
	Step 4: Admin mengisi data retur dan distributor.	
	Step 5: Admin mengklik tombol SIMPAN.	Step 6: Sistem menyimpan data obat tersebut ke


	<p>database.</p> <p>Step 7: Sistem menampilkan pesan keberhasilan.</p> <p>Step 8: Admin mengklik data retur dan distributor.</p> <p>Step 9: Admin mengklik tombol DETAIL.</p> <p>Step 10: Sistem menampilkan form untuk mengisi data obat yang ingin diretur.</p> <p>Step 11: Admin mengisi data obat yang akan diretur.</p> <p>Step 12: Admin mengklik tombol SIMPAN.</p> <p>Step 13: Sistem menyimpan data obat tersebut ke database.</p> <p>Step 14: Sistem menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 5: Jika admin tidak jadi memasukan data retur dan distributor tersebut, maka admin akan menekan tombol BATAL dan akan kembali ke form untuk mengisi data retur dan distributor.</p> <p>Alt-Step 7: Sistem akan menampilkan peringatan apabila sistem tidak dapat data retur yang baru.</p> <p>Alt-Step 12: Jika admin tidak jadi memasukan data obat tersebut, maka admin akan menekan tombol BATAL dan akan kembali ke form untuk mengisi data obat.</p> <p>Alt-Step 14: Sistem akan menampilkan peringatan apabila sistem tidak dapat data retur yang baru.</p>
Conclusion:	Use case ini berhenti apabila tidak ada lagi obat yang harus diretur.
Postcondition:	<ul style="list-style-type: none"> • Obat berhasil diretur. • Obat tidak berhasil diretur.
Business Rules:	Admin harus memasukan data retur dan obat dengan benar.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	Setiap obat memiliki tanggal kadaluarsa, oleh karena itu enam bulan sebelum tanggal tersebut, admin harus meneliti

	obat yang hampir kadaluarsa dan membuat faktur pembelian. Selain itu setiap obat yang datang dari distributor tidak selalu dalam kondisi yang bagus. Oleh karena itu obat tersebut perlu diretur.
--	---

Narasi Use-Case Menghapus Data Retur dapat dilihat pada tabel berikut:

Tabel 3.19 Narasi Use-Case Menghapus Data Retur

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Data Retur	Jenis Use-Case Business Requirements: 
Use-case ID:	UC-018	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Penghapusan data dilakukan apabila data yang sudah tidak diperlukan lagi.	
Precondition:	Admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data retur yang tidak diperlukan lagi.	
Typical Course of event:	Actor Action	System Response
	Step 1: Admin memilih menu retur.	Step 2: Sistem menampilkan form untuk menghapus data retur.
	Step 3: Admin mengklik data yang ingin dihapus.	
	Step 4: Admin mengklik tombol HAPUS.	Step 5: Sistem menghapus data.
		Step 6: Sistem menampilkan pesan keberhasilan.
Alternate	Alt-Step 4: Jika admin mengklik tombol BATAL, sehingga	

Course:	sistem akan kembali ke form untuk menghapus data retur. Alt-Step 6: Sistem akan menampilkan peringatan apabila sistem tidak dapat menghapus.
Conclusion:	Use case ini berhenti apabila tidak ada data retur yang harus dihapus.
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dari database.
Business Rules:	Admin harus memastikan bahwa data yang ingin dihapus adalah data yang tidak diperlukan lagi oleh sistem.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	-

Narasi Use-Case Mengisi Data Embalase dapat dilihat pada tabel berikut:

Tabel 3.20 Narasi Use-Case Mengisi Data Embalase

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengisi Data Embalase	<div>Jenis Use-Case Business Requirements:<div><div></div></div></div>	
Use-case ID:	UC-019		
Priority :	High		
Source:	-		
Primary Business Actor:	Pemilik dan admin.		
Other Participating Actor:	-		
Other Interested Stakeholder:	-		
Decription:	Use case ini menggambarkan proses penginputan data embalase yang dilakukan oleh pemilik atau admin.		
Precondition:	Pemilik atau admin telah melakukan proses login.		
Trigger:	Use case ini digunakan apabila ada data embalase baru.		
Typical Course	Actor Action	System Response	

of event:	<p>Step 1: Pemilik / admin memilih menu setup embalase.</p> <p>Step 3: Pemilik / admin mengklik tombol TAMBAH.</p> <p>Step 4: Pemilik / admin mengisikan data embalase yang ingin diinputkan.</p> <p>Step 5: Pemilik / admin mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk menambah data embalase.</p> <p>Step 6: Sistem menambahkan data embalase ke dalam database.</p> <p>Step 7: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 5: Pemilik / admin mengklik tombol BATAL, sehingga sistem tidak jadi menambahkan data embalase ke dalam database dan akan kembali ke form untuk menambah data embalase.</p> <p>Alt-Step 7: Jika pemilik / admin tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data embalase yang harus diinputkan.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pemilik / admin untuk menginputkan data.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Mengubah Data Embalase dapat dilihat pada tabel berikut:

Tabel 3.21 Narasi Use-Case Mengubah Data Embalase

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengubah Data Embalase	Jenis Use-Case
-----------------------	------------------------	-----------------------

Use-case ID:	UC-020	Business Requirements: <input checked="" type="checkbox"/>
Priority :	Medium	
Source:	-	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pengubahan data embalase yang dilakukan oleh pemilik atau admin apabila terjadi perubahan harga embalase atau nama dari embalase itu sendiri.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data embalase yang ingin diubah.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin memilih menu setup embalase.	Step 2: Sistem menampilkan form untuk merubah data embalase.
	Step 3: Pemilik / admin mengganti data lama dengan data yang baru.	
	Step 4: Pemilik / admin mengklik tombol SIMPAN.	Step 5: Sistem menyimpan data yang baru tersebut ke dalam database.
		Step 6: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 4: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk mengubah data embalase. Alt-Step 6: Sistem akan menampilkan peringatan apabila sistem tidak dapat merubah data lama dengan data yang baru.	
Conclusion:	Use case ini berhenti apabila tidak ada data embalase yang harus dirubah.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil diubah dan disimpan ke database. • Data tidak berhasil diubah. 	

Business Rules:	Pemilik / admin harus memasukan data dengan tipe yang sesuai.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	-

Narasi Use-Case Menghapus Data Embalase dapat dilihat pada tabel berikut:

Tabel 3.22 Narasi Use-Case Menghapus Data Embalase

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0


Use-case Name:	Menghapus Data Embalase	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-021	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Penghapusan data dilakukan apabila data yang sudah tidak diperlukan lagi.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila embalase tidak digunakan lagi.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin memilih menu setup embalase.	Step 2: Sistem menampilkan form untuk menghapus data embalase.
	Step 3: Pemilik / admin mengklik data yang ingin dihapus.	Step 4: Data yang diklik muncul pada setiap textfield yang tersedia.
	Step 5: Pemilik / admin mengklik tombol HAPUS.	Step 6: Sistem menghapus data.

		Step 7: Sistem menampilkan pesan keberhasilan.
Alternate Course:	Alt-Step 5: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk menghapus data embalase. Alt-Step 7: Sistem akan menampilkan peringatan apabila sistem tidak dapat menghapus.	
Conclusion:	Use case ini berhenti apabila tidak ada data embalase yang harus dihapus.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dari database. 	
Business Rules:	Pemilik / admin harus memastikan bahwa data yang ingin dihapus adalah data yang tidak diperlukan lagi oleh sistem.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Mengisi Data Toeslag dapat dilihat pada tabel berikut:

Tabel 3.23 Narasi Use-Case Mengisi Data Toeslag

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengisi Data Toeslag	Jenis Use-Case Business Requirements: 	
Use-case ID:	UC-022		
Priority :	High		
Source:	-		
Primary Business Actor:	Pemilik dan admin.		
Other Participating Actor:	-		
Other Interested Stakeholder:	-		
Decription:	Use case ini menggambarkan proses penginputan data toeslag yang dilakukan oleh pemilik atau admin.		
Precondition:	Pemilik atau admin telah melakukan proses login.		
Trigger:	Use case ini digunakan apabila ada data toeslag baru.		
Typical Course	Actor Action	System Response	


of event:	<p>Step 1: Pemilik / admin memilih menu setup toeslag.</p> <p>Step 3: Pemilik / admin mengklik tombol TAMBAH.</p> <p>Step 4: Pemilik / admin mengisikan data toeslag yang ingin diinputkan.</p> <p>Step 5: Pemilik / admin mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk menambah data toeslag.</p> <p>Step 6: Sistem menambahkan data toeslag ke dalam database.</p> <p>Step 7: Sistem akan menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 5: Pemilik / admin mengklik tombol BATAL, sehingga sistem tidak jadi menambahkan data toeslag ke dalam database dan akan kembali ke form untuk menambah data toeslag.</p> <p>Alt-Step 7: Jika pemilik / admin tidak berhasil melakukan proses penambahan data, maka sistem akan menampilkan peringatan.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data toeslag yang harus diinputkan.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil ditambahkan ke database. • Data tidak berhasil ditambahkan ke database. 	
Business Rules:	Pemilik / admin harus memasukan data dengan tipe yang sesuai.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI yang akan mempermudah pemilik / admin untuk menginputkan data.	
Assumptions:	-	
Open issues:	-	

Narasi Use-Case Mengubah Data Toeslag dapat dilihat pada tabel berikut:

Tabel 3.24 Narasi Use-Case Mengubah Data Toeslag

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Mengubah Data Toeslag	Jenis Use-Case Business Requirements:
Use-case ID:	UC-023	

		
Priority :	Medium	
Source:	-	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses pengubahan data toeslag yang dilakukan oleh pemilik atau admin apabila terjadi perubahan harga toeslag atau nama dari toeslag itu sendiri.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila ada data toeslag yang ingin diubah.	
Typical Course of event:	Actor Action	System Response
	<p>Step 1: Pemilik / admin memilih menu setup toeslag.</p> <p>Step 3: Pemilik / admin mengganti data lama dengan data yang baru.</p> <p>Step 4: Pemilik / admin mengklik tombol SIMPAN.</p>	<p>Step 2: Sistem menampilkan form untuk merubah data toeslag.</p> <p>Step 5: Sistem menyimpan data yang baru tersebut ke dalam database.</p> <p>Step 6: Sistem menampilkan pesan keberhasilan.</p>
Alternate Course:	<p>Alt-Step 4: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk mengubah data toeslag.</p> <p>Alt-Step 6: Sistem akan menampilkan peringatan apabila sistem tidak dapat merubah data lama dengan data yang baru.</p>	
Conclusion:	Use case ini berhenti apabila tidak ada data toeslag yang harus dirubah.	
Postcondition:	<ul style="list-style-type: none"> • Data berhasil diubah dan disimpan ke database. • Data tidak berhasil diubah. 	
Business Rules:	Pemilik / admin harus memasukan data dengan tipe yang sesuai.	

Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	-

Narasi Use-Case Menghapus Data Toeslag dapat dilihat pada tabel berikut:

Tabel 3.25 Narasi Use-Case Menghapus Data Toeslag

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Menghapus Data Toeslag	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-024	
Priority :	Low	
Source:	Data yang sudah tersimpan.	
Primary Business Actor:	Pemilik dan admin.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses penghapusan data. Penghapusan data dilakukan apabila data yang sudah tidak diperlukan lagi.	
Precondition:	Pemilik atau admin telah melakukan proses login.	
Trigger:	Use case ini digunakan apabila toeslag tidak digunakan lagi.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin memilih menu setup toeslag.	Step 2: Sistem menampilkan form untuk menghapus data toeslag.
	Step 3: Pemilik / admin mengklik data yang ingin dihapus.	Step 4: Data yang diklik muncul pada setiap textfield yang tersedia.
	Step 5: Pemilik / admin mengklik tombol HAPUS.	Step 6: Sistem menghapus data.
		Step 7: Sistem menampilkan pesan

	keberhasilan.
Alternate Course:	<p>Alt-Step 5: Jika pemilik / admin mengklik tombol BATAL, sehingga sistem akan kembali ke form untuk menghapus data toeslag.</p> <p>Alt-Step 7: Sistem akan menampilkan peringatan apabila sistem tidak dapat menghapus.</p>
Conclusion:	Use case ini berhenti apabila tidak ada data toeslag yang harus dihapus.
Postcondition:	<ul style="list-style-type: none"> • Data berhasil dihapus dari database. • Data tidak berhasil dihapus dari database.
Business Rules:	Pemilik / admin harus memastikan bahwa data yang ingin dihapus adalah data yang tidak diperlukan lagi oleh sistem.
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.
Assumptions:	-
Open issues:	-

Narasi Use-Case Peningat Kadaluarsa dapat dilihat pada tabel berikut:

Tabel 3.26 Narasi Use-Case Peningat Kadaluarsa

Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0

Use-case Name:	Pengingat Kadaluarsa	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-025	
Priority :	High	
Source:	Data obat yang sudah tersimpan.	
Primary Business Actor:	Pemilik, admin, dan pegawai.	
Other Participating Actor:	Waktu	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses yang berjalan otomatis ketika sistem dijalankan. Proses tersebut adalah proses untuk mengingatkan tanggal obat hampir kadaluarsa.	
Precondition:	Pemilik atau admin atau pegawai telah melakukan proses login.	

Trigger:	Use case ini digunakan apabila ada obat yang hampir kadaluarsa.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin / pegawai menjalankan program.	Step 2: Sistem menampilkan pesan obat telah kadaluarsa.
Alternate Course:	Alt-Step 2: Jika tidak ada obat yang hampir kadaluarsa maka tidak ada pesan yang muncul.	
Conclusion:	Use case ini berhenti apabila tidak ada data obat yang hampir kadaluarsa.	
Postcondition:	<ul style="list-style-type: none"> • Pesan obat kadaluarsa muncul. • Pesan obat kadaluarsa tidak muncul. 	
Business Rules:	Pemilik / admin / pegawai telah melakukan login.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Data obat yang ada di Apotek Joint Farma jumlahnya banyak sekitar 500 sampai 1000 obat, sehingga apabila dicek tanggal kadaluarsanya satu per satu maka akan membutuhkan waktu yang lama.	

Narasi Use-Case Peningat Limit dapat dilihat pada tabel berikut:

Tabel 3.27 Narasi Use-Case Peningat Limit

Author : Olivia Dian Kusumawati **Date : 21 November 2008**
Version : 1.0

Use-case Name:	Pengingat Limit	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-026	
Priority :	High	
Source:	Data obat yang sudah tersimpan.	
Primary Business Actor:	Pemilik, admin, dan pegawai.	
Other Participating Actor:	Waktu	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses yang berjalan otomatis ketika sistem dijalankan. Proses tersebut adalah proses untuk mengingatkan obat yang telah mencapai limit.	
Precondition:	Pemilik atau admin atau pegawai telah melakukan proses login.	

Trigger:	Use case ini digunakan apabila ada obat yang stoknya sama dengan limit yang ditentukan oleh pemilik / admin.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik / admin / pegawai menjalankan program.	Step 2: Sistem menampilkan pesan stok obat sama dengan limit.
Alternate Course:	Alt-Step 2: Jika tidak ada obat yang hampir kadaluarsa maka tidak ada pesan yang muncul.	
Conclusion:	Use case ini berhenti apabila tidak ada data obat yang hampir kadaluarsa.	
Postcondition:	<ul style="list-style-type: none"> • Pesan obat kadaluarsa muncul. • Pesan obat kadaluarsa tidak muncul. 	
Business Rules:	Pemilik / admin / pegawai telah melakukan login.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Data obat yang ada di Apotek Joint Farma jumlahnya banyak sekitar 500 sampai 1000 obat, sehingga apabila dicek limitnya satu per satu maka akan membutuhkan waktu yang lama.	

Narasi Use-Case LOG OUT dapat dilihat pada tabel berikut:

Tabel 3.28 Narasi Use-Case LOG OUT

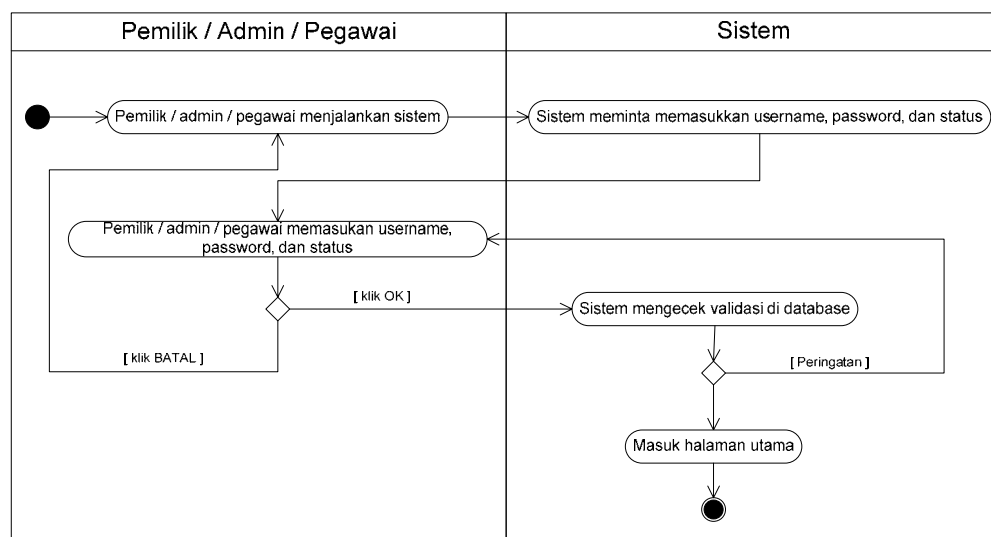
Author : Olivia Dian Kusumawati Date : 21 November 2008
Version : 1.0

Use-case Name:	LOG OUT	Jenis Use-Case Business Requirements: <input checked="" type="checkbox"/>
Use-case ID:	UC-027	
Priority :	High	
Source:	-	
Primary Business Actor:	Pemilik, admin, dan pegawai.	
Other Participating Actor:	-	
Other Interested Stakeholder:	-	
Decription:	Use case ini menggambarkan proses keluar dari sistem. Use case ini digunakan karena adanya 3 orang yang berhak mengakses sistem, namun ketiga orang tersebut mempunyai hak akses yang berbeda. Oleh karena itu dibutuhkan menu keluar dari sistem sehingga hak akses tersebut dapat terjaga.	

Precondition:	Pemilik atau admin atau pegawai telah melakukan LOG IN.	
Trigger:	Use case ini digunakan apabila pemilik atau admin atau pegawai ingin keluar dari sistem.	
Typical Course of event:	Actor Action	System Response
	Step 1: Pemilik atau admin atau pegawai mengklik menu keluar.	Step 2: Sistem akan keluar.
Alternate Course:	-	
Conclusion:	Use case ini berhenti apabila pemilik atau admin atau pegawai telah berhasil keluar dari sistem.	
Postcondition:	Pemilik atau admin atau pegawai keluar dari sistem.	
Business Rules:	Pemilik atau admin atau pegawai sudah melewati proses LOG IN.	
Implementation Constrains and Specifications:	Tampilan sistem berupa GUI.	
Assumptions:	-	
Open issues:	Karena tidak hanya satu orang saja yang mengakses data-data yang ada dikhawatirkan data yang harusnya tidak boleh dilihat oleh orang yang tidak memiliki wewenag pada akhirnya juga dapat dilihat.	

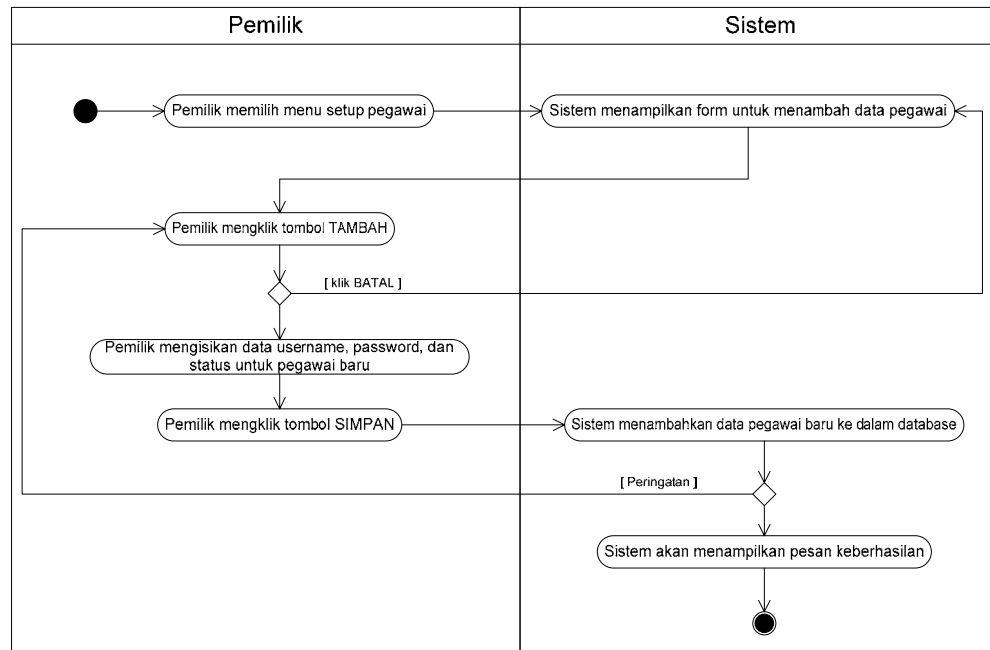
3.7. Diagram Activity

Berikut ini merupakan diagram activity untuk proses Log in.



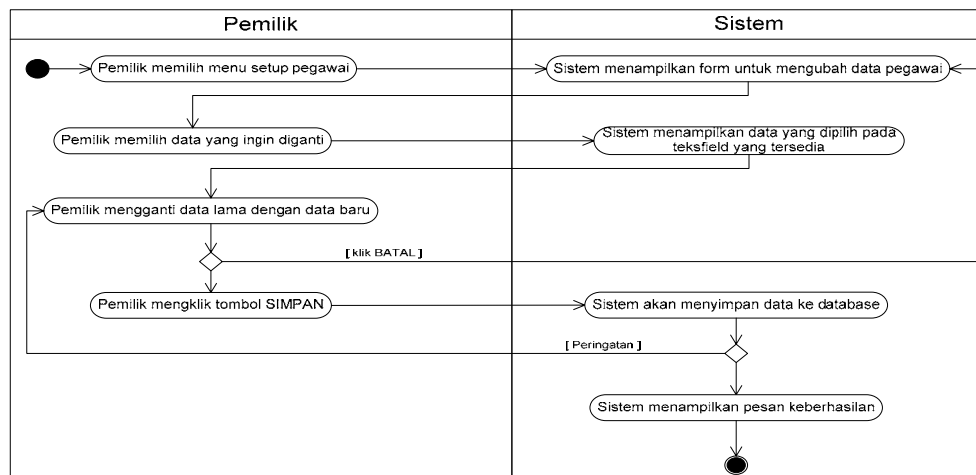
Gambar 3.3 Diagram Activity Log in

Berikut ini merupakan diagram activity untuk proses Mengisi Data Pegawai.



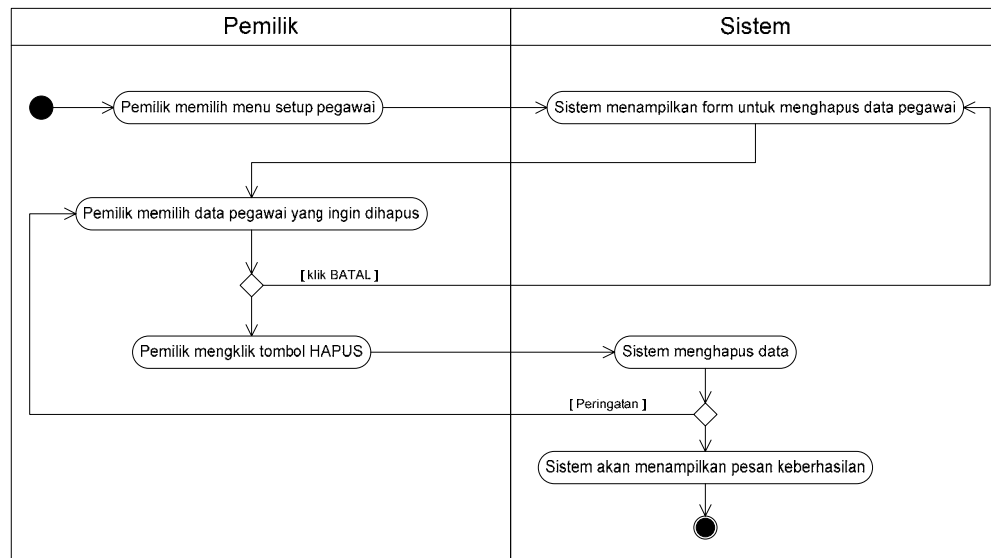
Gambar 3.4 Diagram Activity Mengisi Data Pegawai

Berikut ini merupakan diagram activity untuk proses Mengubah Data Pegawai.



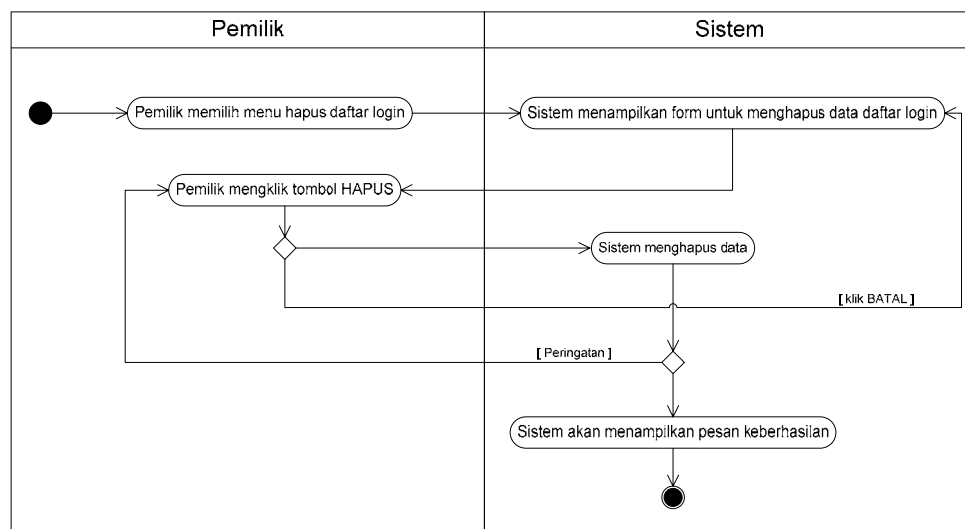
Gambar 3.5 Diagram Activity Mengubah Data Pegawai

Berikut ini merupakan diagram activity untuk proses Menghapus Data Pegawai.



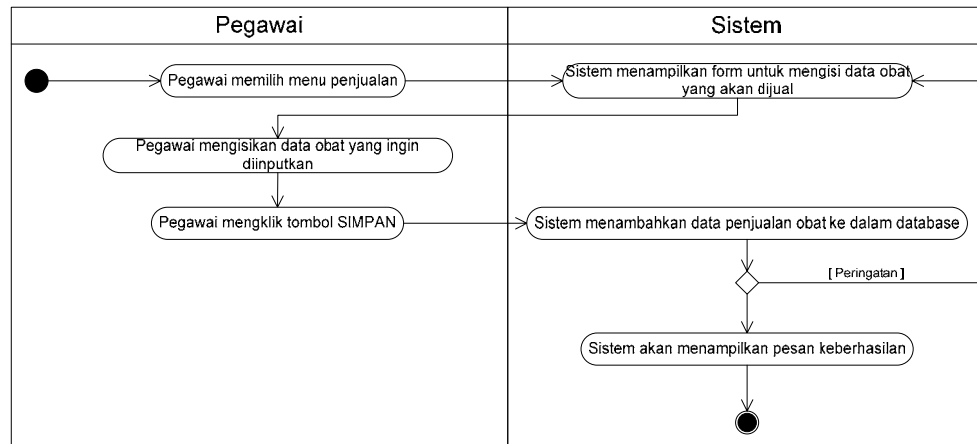
Gambar 3.6 Diagram Activity Menghapus Data Pegawai

Berikut ini merupakan diagram activity untuk proses Menghapus Daftar Login.



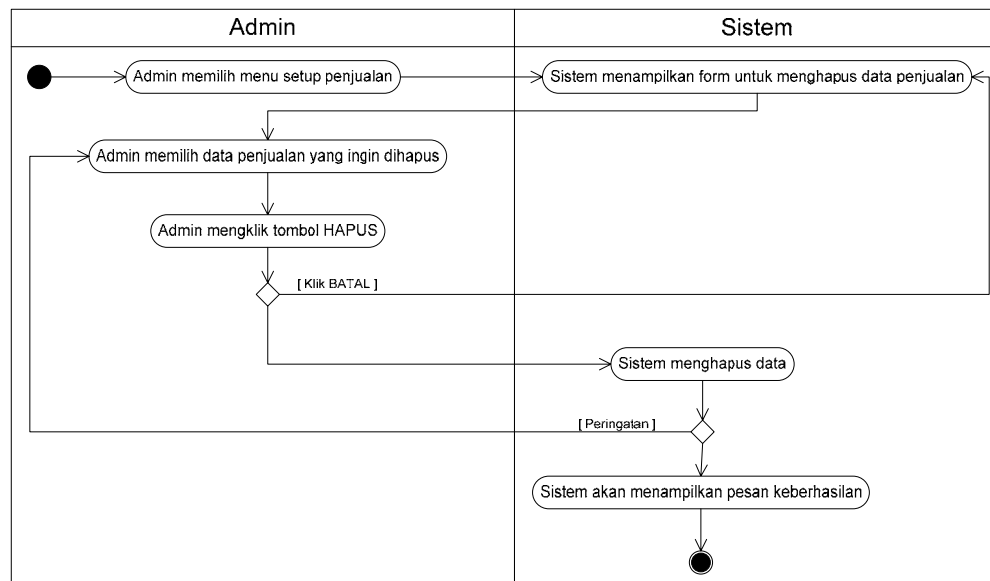
Gambar 3.7 Diagram Activity Menghapus Daftar Login

Berikut ini merupakan diagram activity untuk proses Mengisi Data Penjualan.



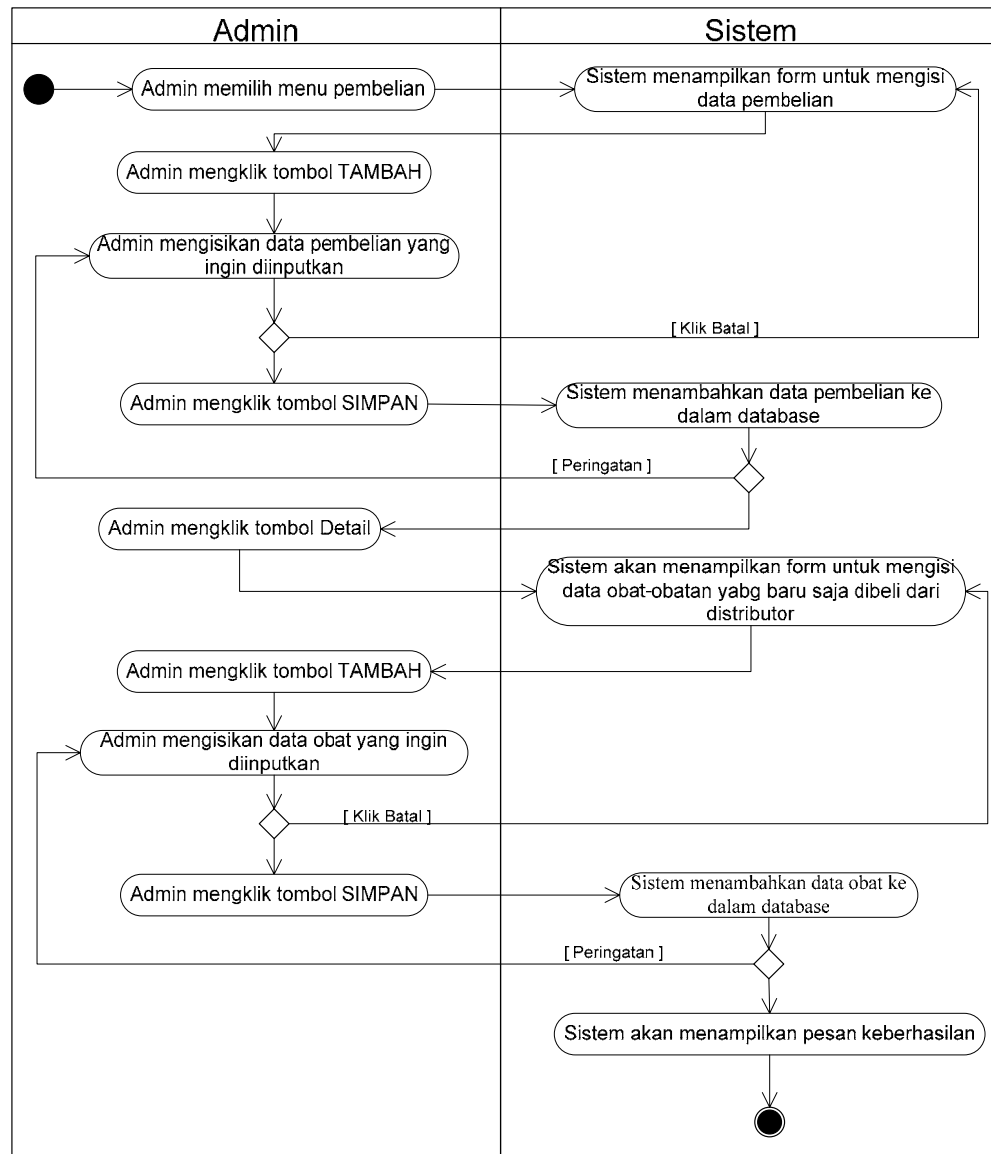
Gambar 3.8 Diagram Activity Mengisi Data Penjualan

Berikut ini merupakan diagram activity untuk proses Menghapus Data Penjualan.



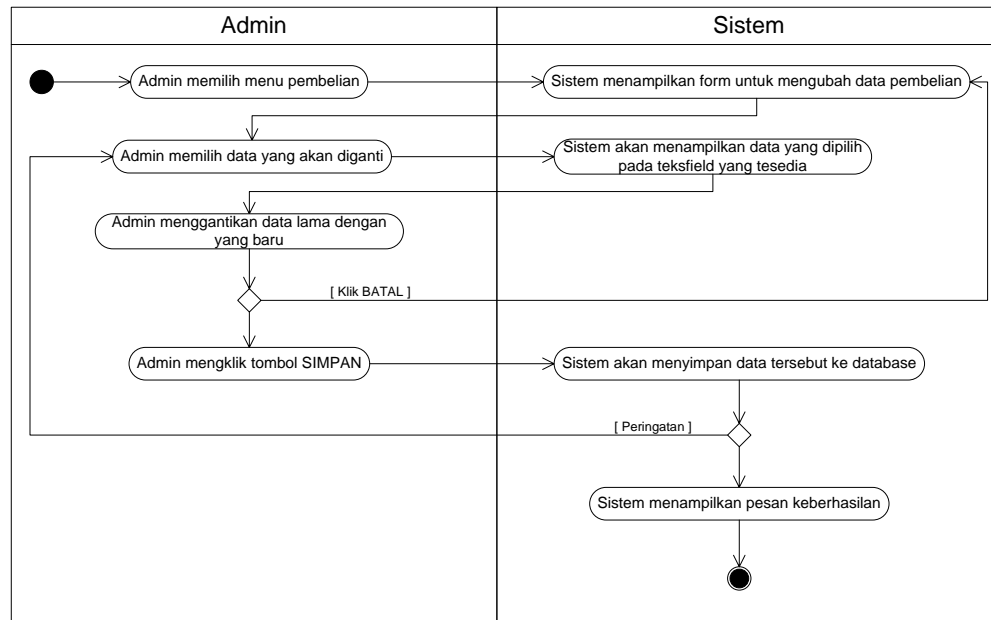
Gambar 3.9 Diagram Activity Menghapus Data Penjualan

Berikut ini merupakan diagram activity untuk proses Mengisi Data Pembelian.



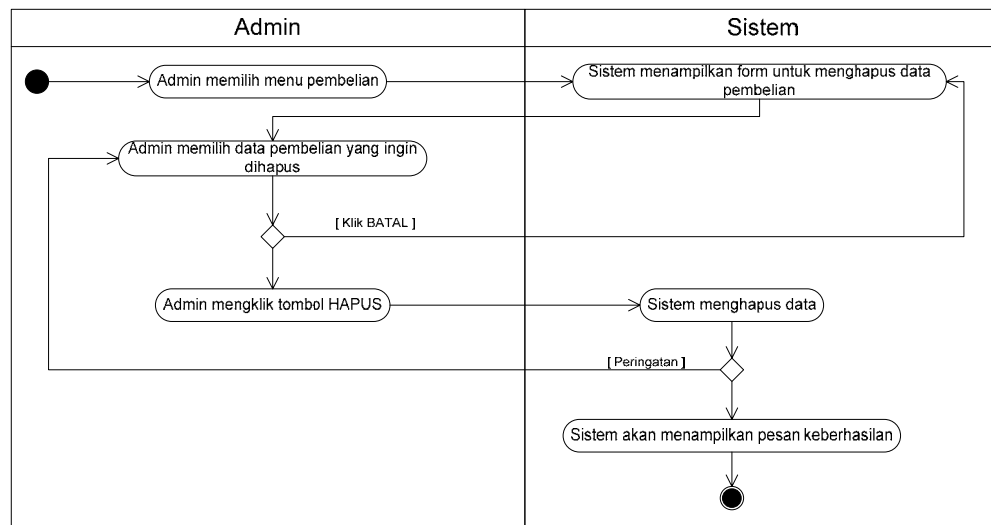
Gambar 3.10 Diagram Activity Mengisi Data Pembelian

Berikut ini merupakan diagram activity untuk proses Mengubah Data Pembelian.



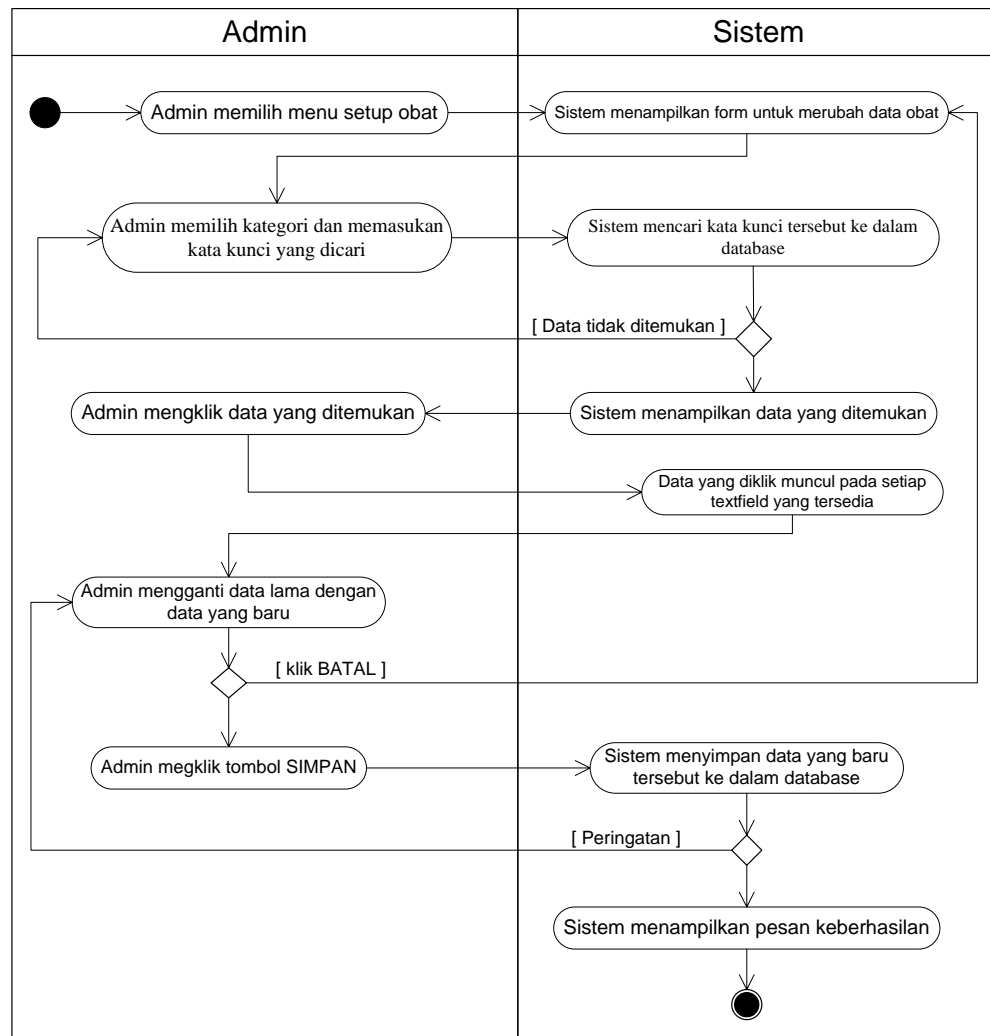
Gambar 3.11 Diagram Activity Mengubah Data Pembelian

Berikut ini merupakan diagram activity untuk proses Menghapus Data Pembelian.



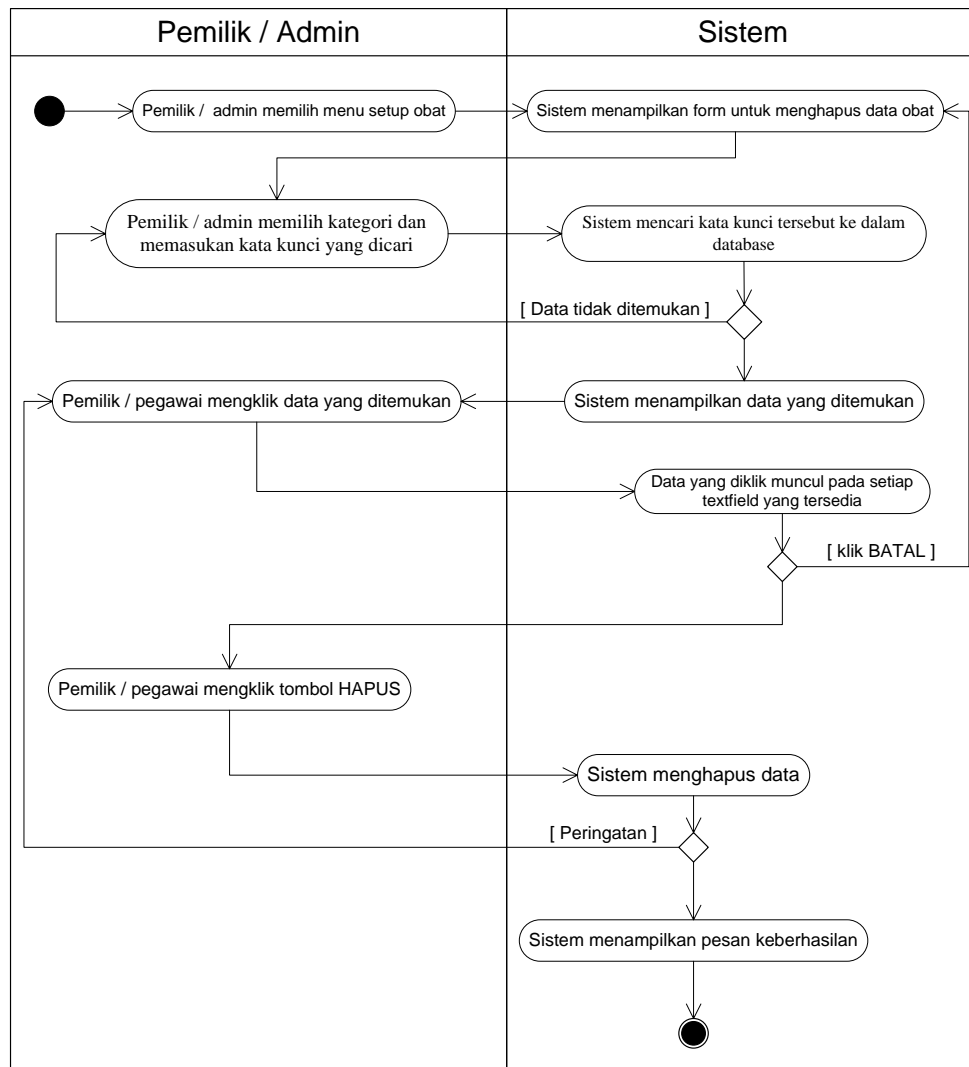
Gambar 3.12 Diagram Activity Menghapus Data Pembelian

Berikut ini merupakan diagram activity untuk proses Mengubah Data Obat.



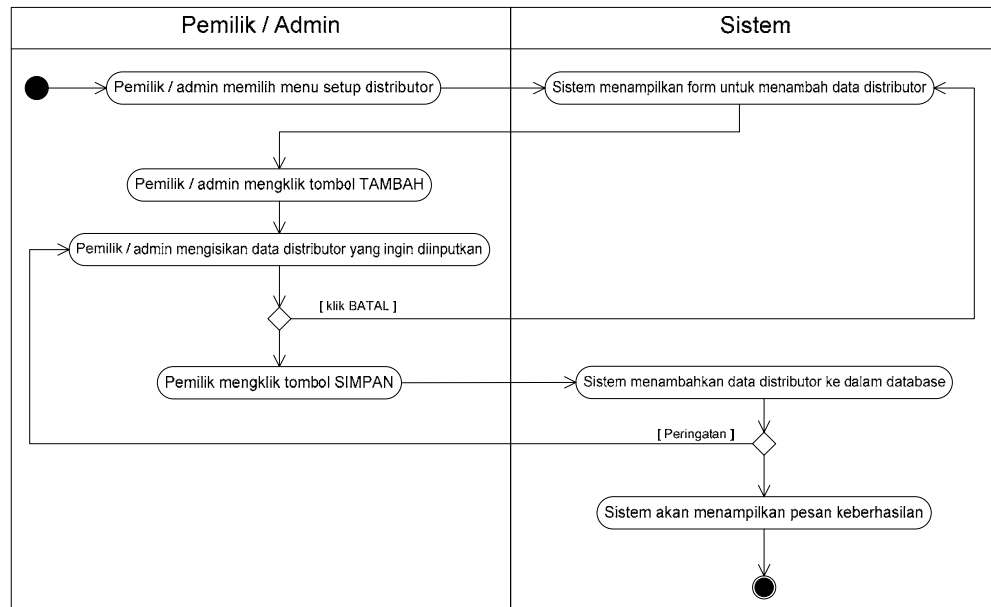
Gambar 3.13 Diagram Activity Mengubah Data Obat

Berikut ini merupakan diagram activity untuk proses Menghapus Data Obat.



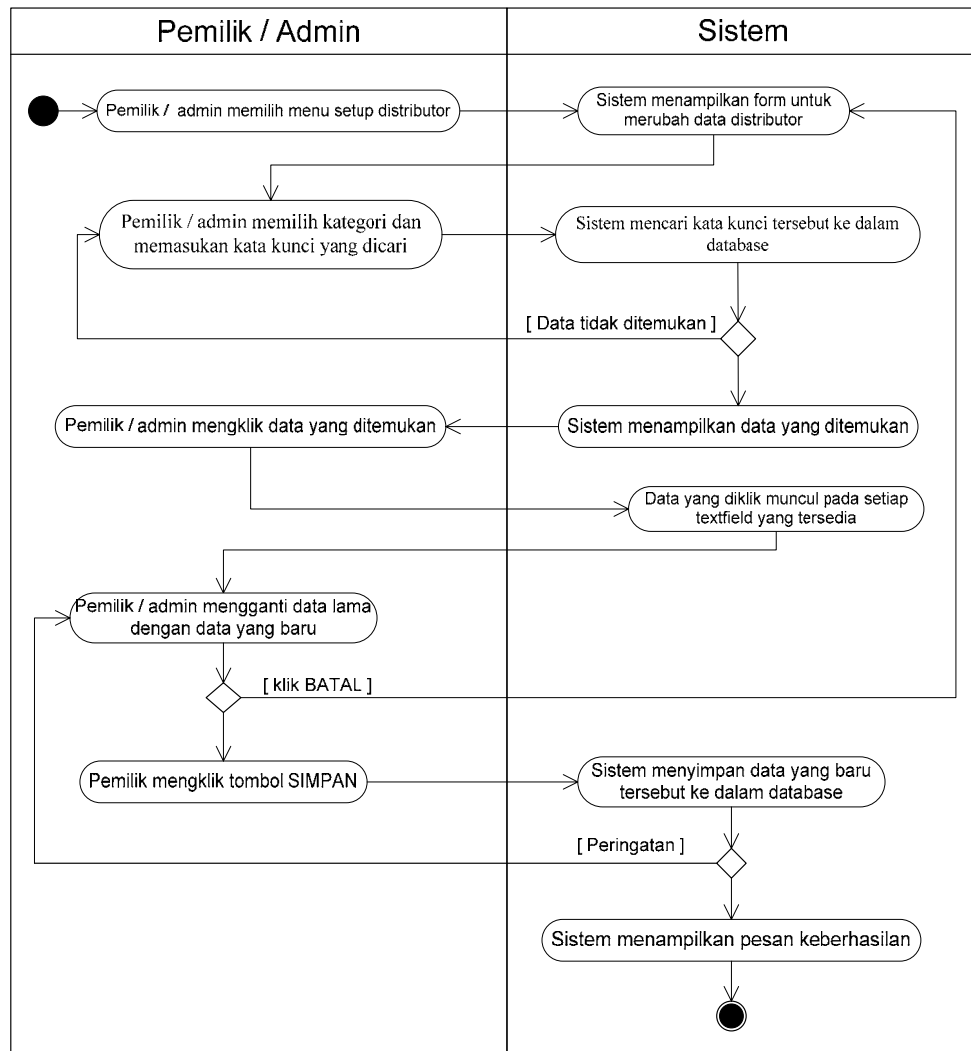
Gambar 3.14 Diagram Activity Menghapus Data Obat

Berikut ini merupakan diagram activity untuk proses Mengisi Data Distributor.



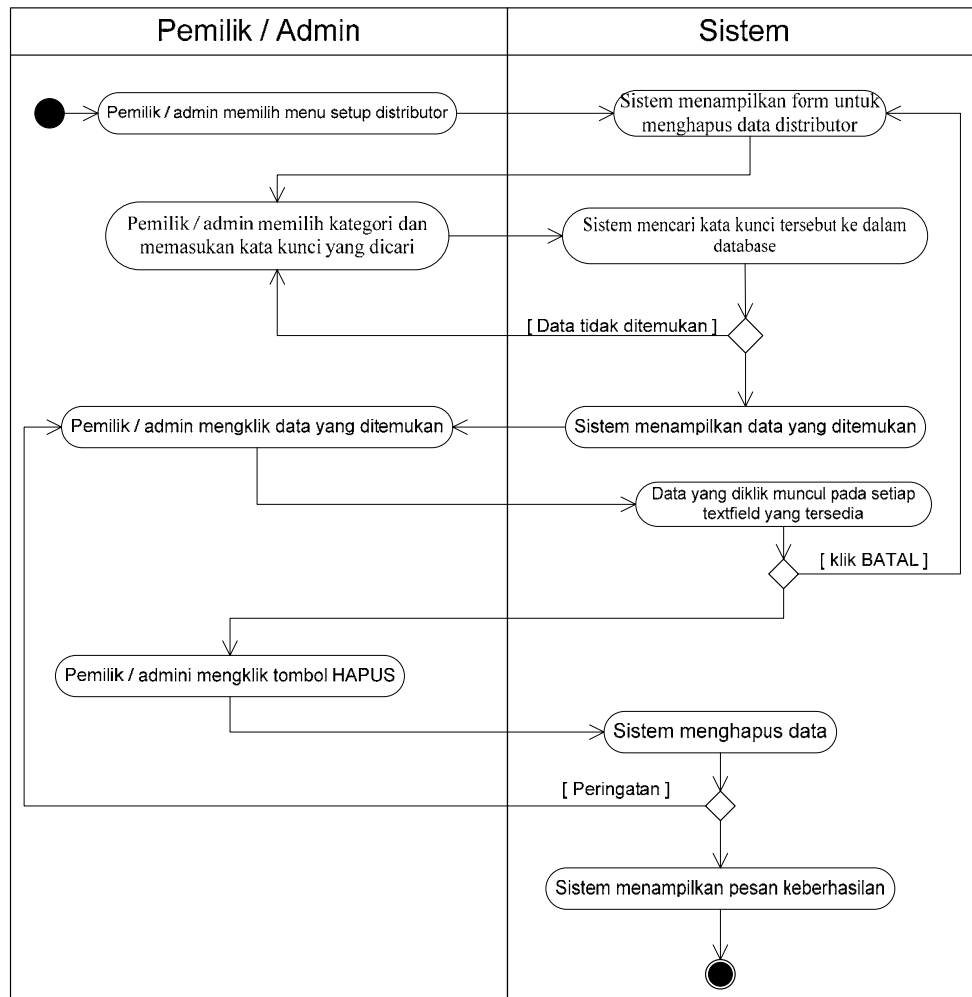
Gambar 3.15 Diagram Activity Mengisi Data Distributor

Berikut ini merupakan diagram activity untuk proses Mengubah Data Distributor.



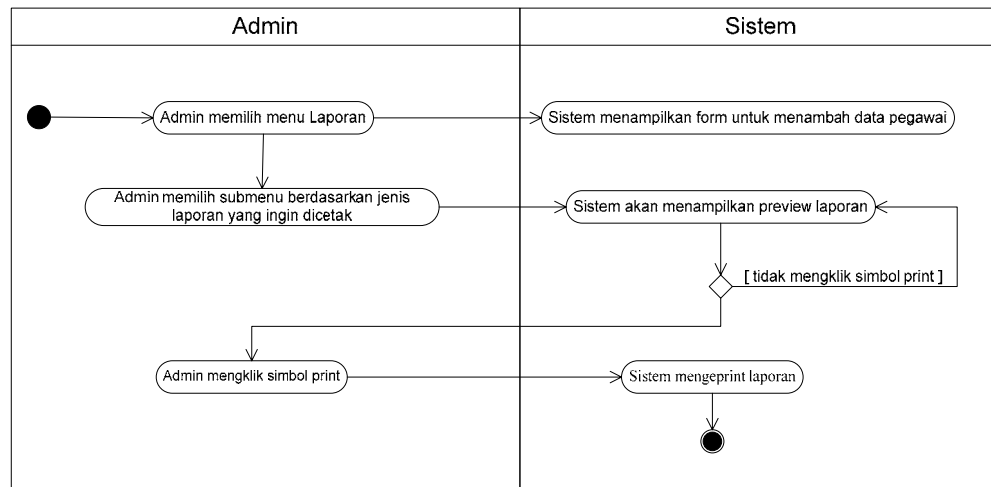
Gambar 3.16 Diagram Activity Mengubah Data Distributor

Berikut ini merupakan diagram activity untuk proses Menghapus Data Distributor.



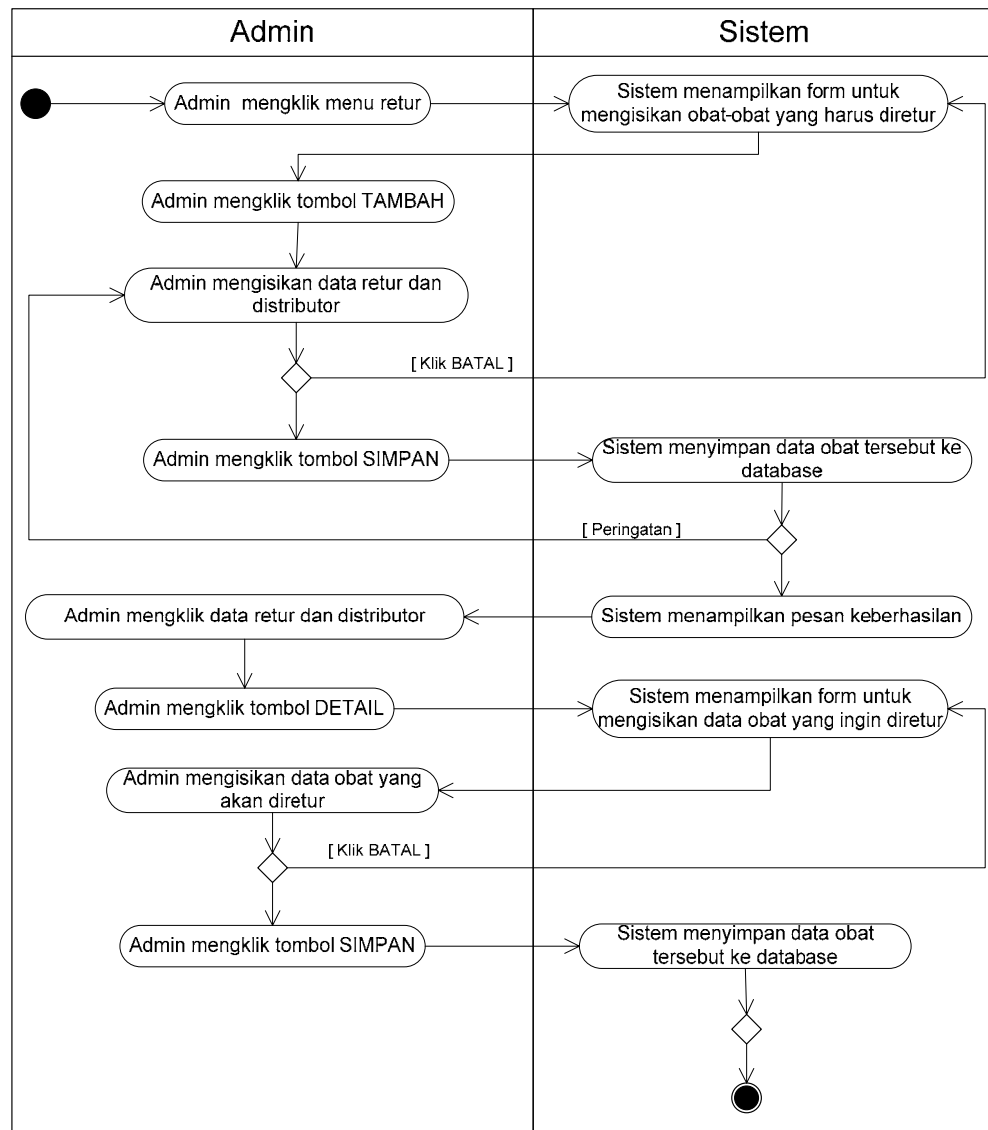
Gambar 3.17 Diagram Activity Menghapus Data Distributor

Berikut ini merupakan diagram activity untuk proses Cetak Laporan.



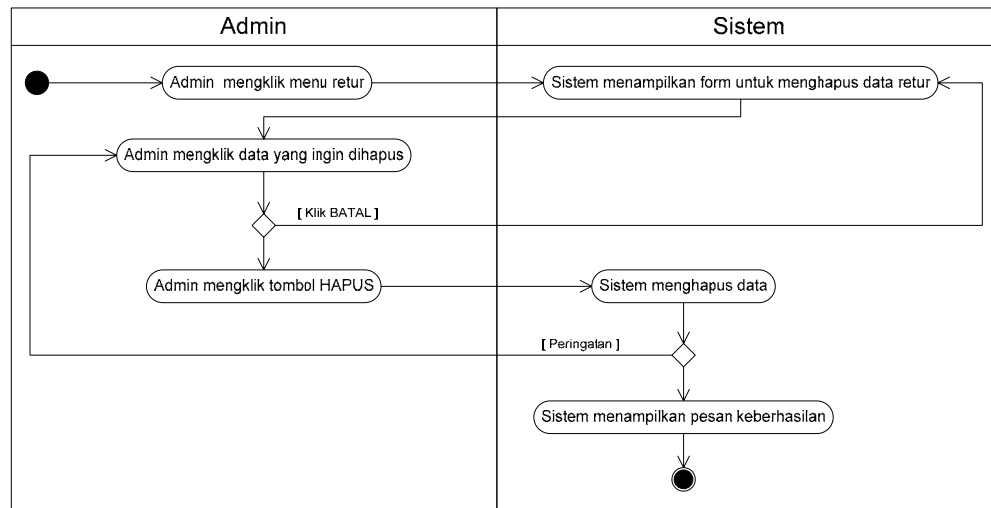
Gambar 3.18 Diagram Activity Cetak Laporan

Berikut ini merupakan diagram activity untuk proses Mengisi Data Retur.



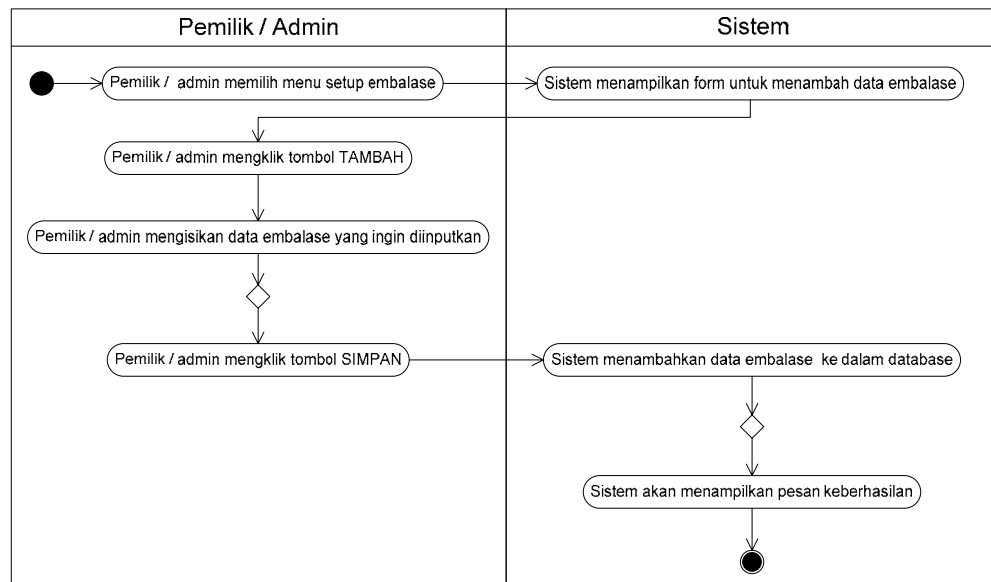
Gambar 3.19 Diagram Activity Mengisi Data Retur

Berikut ini merupakan diagram activity untuk proses Menghapus Data Retur.



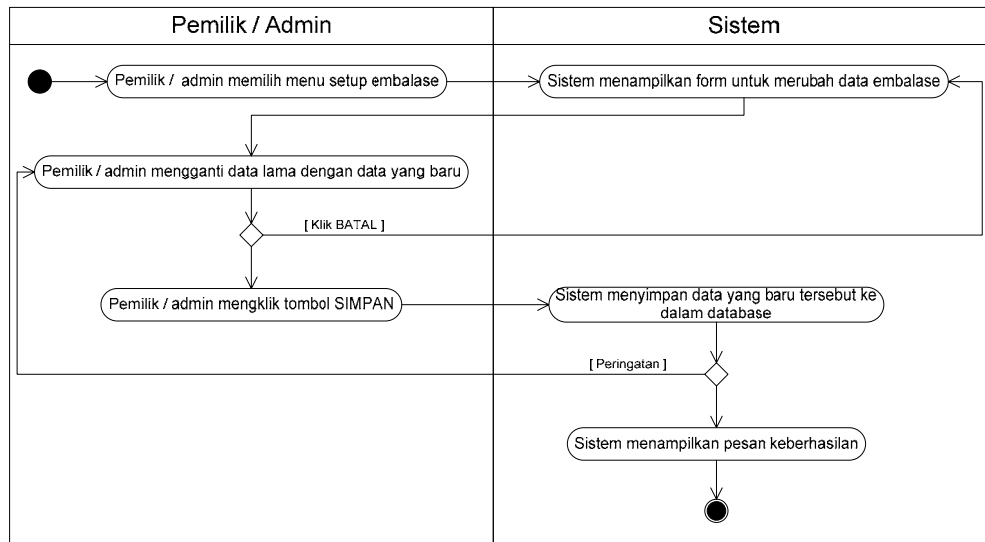
Gambar 3.20 Diagram Activity Menghapus Data Retur

Berikut ini merupakan diagram activity untuk proses Mengisi Data Embalase.



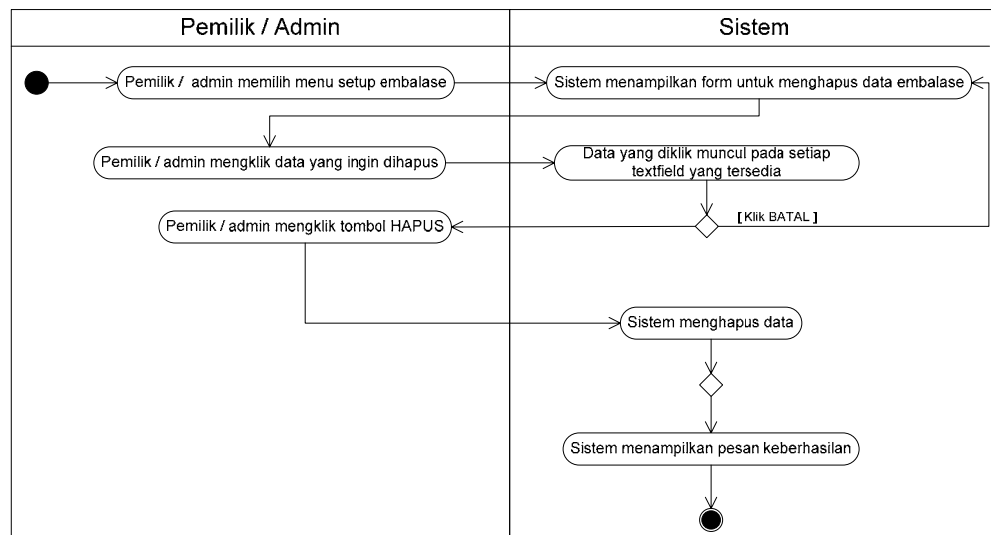
Gambar 3.21 Diagram Activity Mengisi Data Embalase

Berikut ini merupakan diagram activity untuk proses Mengubah Data Embalase.



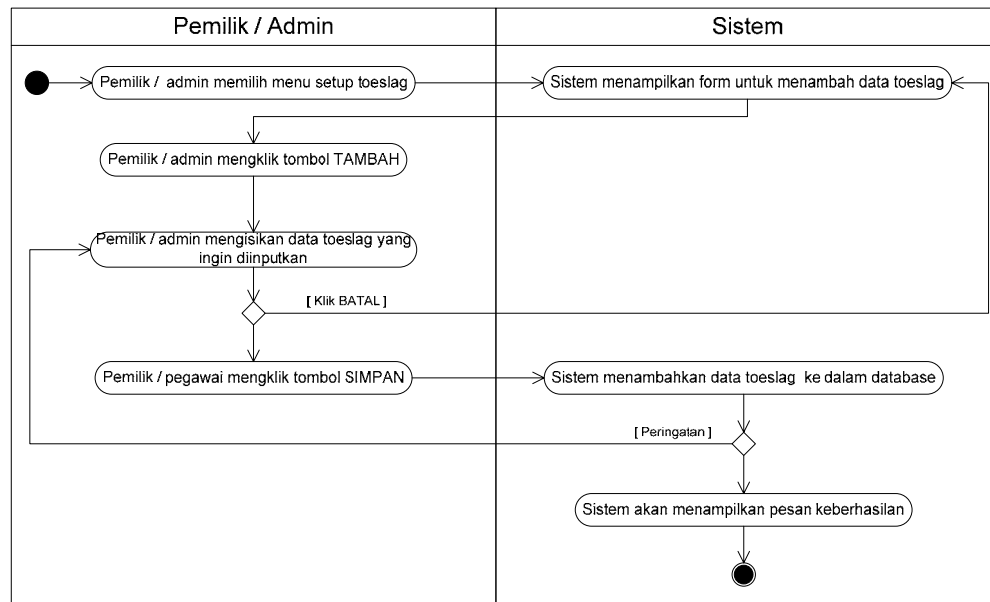
Gambar 3.22 Diagram Activity Mengubah Data Embalase

Berikut ini merupakan diagram activity untuk proses Menghapus Data Embalase.



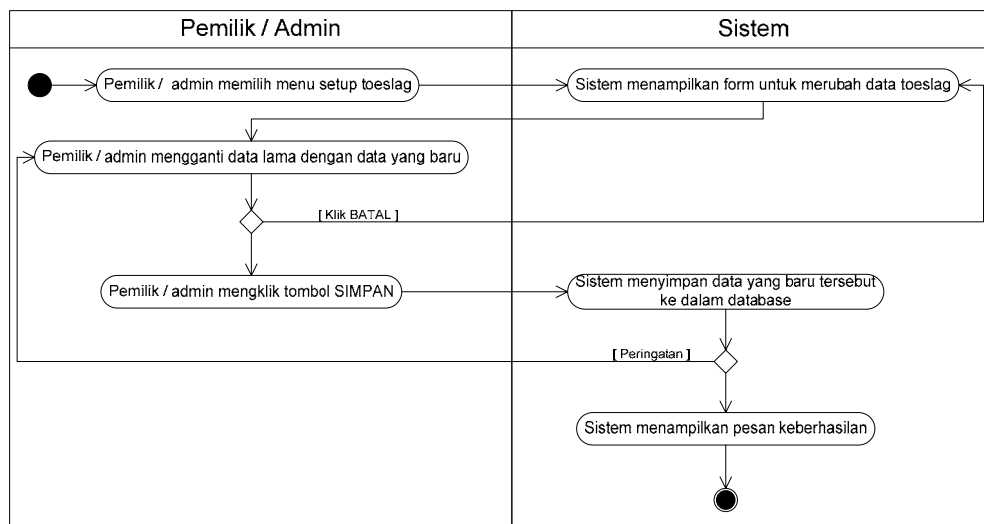
Gambar 3.23 Diagram Activity Menghapus Data Embalase

Berikut ini merupakan diagram activity untuk proses Mengisi Data Toeslag.



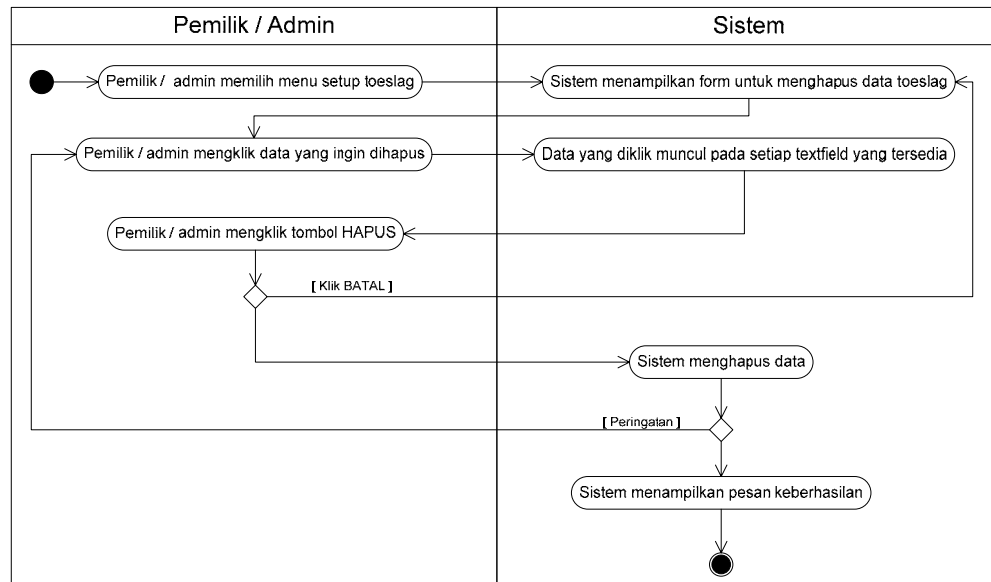
Gambar 3.24 Diagram Activity Mengisi Data Toeslag

Berikut ini merupakan diagram activity untuk proses Mengubah Data Toeslag.



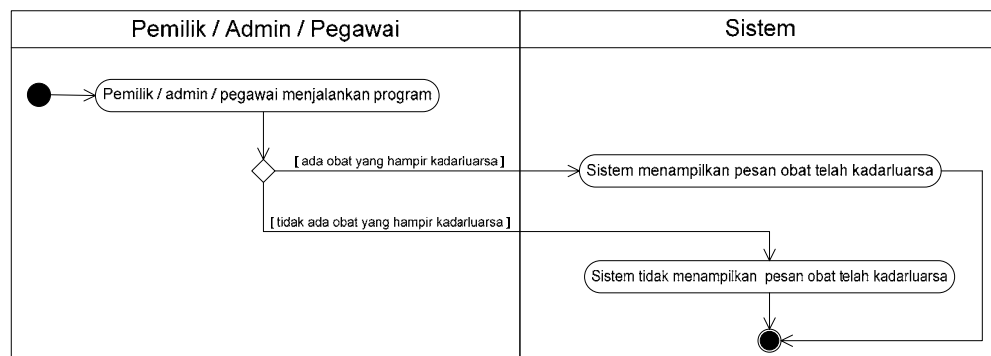
Gambar 3.25 Diagram Activity Mengubah Data Toeslag

Berikut ini merupakan diagram activity untuk proses Menghapus Data Toeslag.



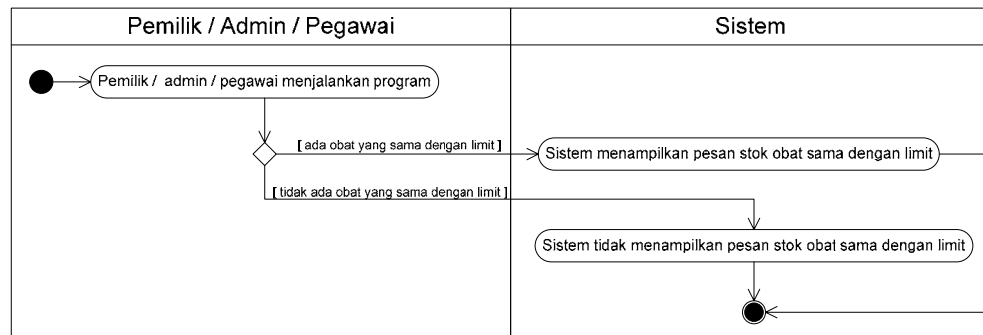
Gambar 3.26 Diagram Activity Menghapus Data Toeslag

Berikut ini merupakan diagram activity untuk proses Peningat Kadaluarsa.



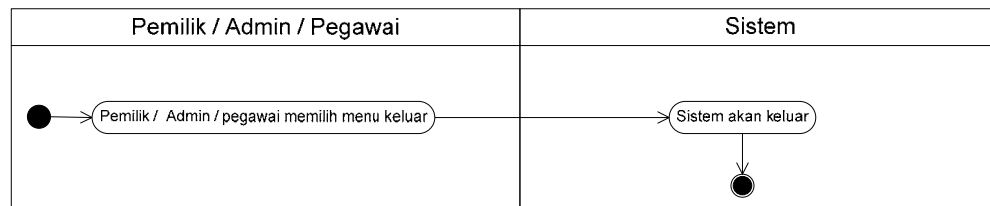
Gambar 3.27 Diagram Activity Peningat Kadaluarsa

Berikut ini merupakan diagram activity untuk proses Peningat Limit.



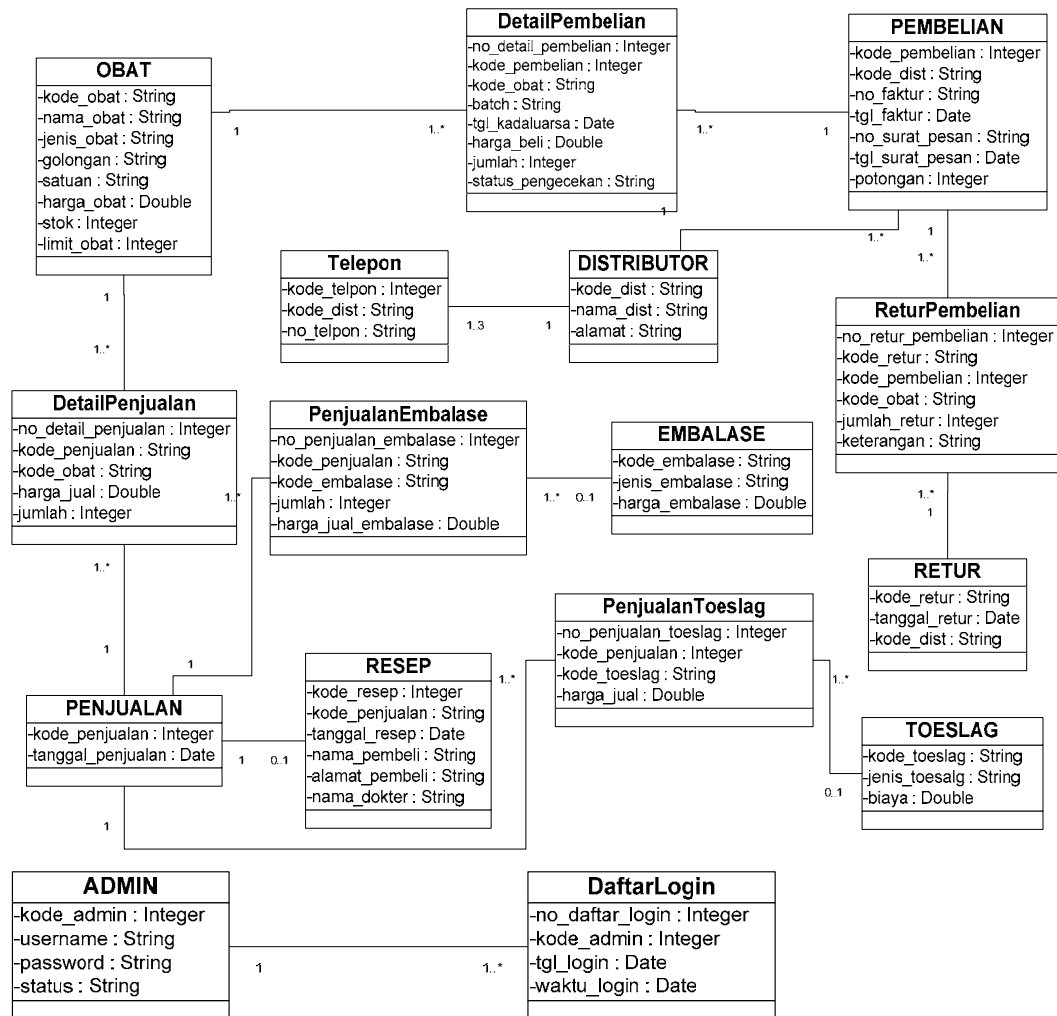
Gambar 3.28 Diagram Activity Peningat Limit

Berikut ini merupakan diagram activity untuk proses LOG OUT.



Gambar 3.29 Diagram Activity LOG OUT

3.8. Class Diagram



Gambar 3.30 Class Diagram System

3.9. Diagram Sequence

Identifikasi Kelas dalam Use-Case Design

Use case LOG IN

Interface, Controller, dan Entity Classes dari Use case LOG IN		
Interface Classes	Controller Classes	Entity Classes
W01-Form Login	Login Handler	ADMIN

Use case Mengisi Data Pegawai

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W03-Form Setup Pegawai	InsertPegawai Handler	ADMIN

Use case Mengubah Data Pegawai

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W03-Form Setup Pegawai	UpdatePegawai Handler	ADMIN

Use case Menghapus Data Pegawai

Interface, Controller, dan Entity Classes dari Use case Delete Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W03-Form Setup Pegawai	DeletePegawai Handler	ADMIN

Use case Menghapus Daftar Login

Interface, Controller, dan Entity Classes dari Use case Delete Daftar Login		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W10-Form Hapus Daftar Login	DafLogin Handler	Daftar Login

Use case Mengisi Data Penjualan

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W04-Form Penjualan	InsertPenjualan Handler	PENJUALAN OBAT EMBALASE TOESLAG

Use case Menghapus Data Penjualan

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W05-Form Setup Penjualan	deletePenjualan Handler	PENJUALAN

Use case Mengisi Data Pembelian

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W06-Form Pembelian	insertPembelian Handler	PEMBELIAN OBAT

Use case Mengubah Data Pembelian

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W06-Form Pembelian	updatePembelian Handler	PEMBELIAN

Use case Menghapus Data Pembelian

Interface, Controller, dan Entity Classes dari Use case Insert Data Pegawai		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W06-Form Pembelian	deletePembelian Handler	PEMBELIAN

Use case Mengubah Data Obat

Interface, Controller, dan Entity Classes dari Use case Update Data Obat		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W07-Form Setup Obat	UpdateObat Handler	OBAT

Use case Menghapus Data Obat

Interface, Controller, dan Entity Classes dari Use case Insert Data Obat		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W07-Form Setup Obat	DeleteObat Handler	OBAT

Use case Mengisi Data Distributor

Interface, Controller, dan Entity Classes dari Insert Data Distributor		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W08-Form Setup Distributor	InsertDistributor Handler	DISTRIBUTOR

Use case Mengubah Data Distributor

Interface, Controller, dan Entity Classes dari Use case Update Data Distributor		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W08-Form Setup Distributor	UpdateDistributor Handler	DISTRIBUTOR

Use case Menghapus Data Distributor

Interface, Controller, dan Entity Classes dari Use case Delete Data Distributor		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W08-Form Setup Distributor	DeleteDistributor Handler	DISTRIBUTOR

Use case Cetak Laporan

Interface, Controller, dan Entity Classes dari Use case Cetak Laporan		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W09-Form Cetak Laporan	CetakLaporan Handler	OBAT PENJUALAN PEMBELIAN DaftarLogin

Use case Mengisi Data Retur

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W10-Form Retur	Retur Handler	RETUR OBAT PEMBELIAN DISTRIBUTOR

Use case Menghapus Data Retur

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W10-Form Retur	Retur Handler	RETUR

Use case Mengisi Data Embalase

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W11-Form Setup Embalase	insertEmbalase Handler	EMBALASE

Use case Mengubah Data Embalase

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W11-Form Setup Embalase	updateEmbalase Handler	EMBALASE

Use case Menghapus Data Embalase

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W11-Form Setup Embalase	deleteEmbalase Handler	EMBALASE

Use case Mengisi Data Toeslag

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W12-Form Setup Toeslag	insertToeslag Handler	TOESLAG

Use case Mengubah Data Toeslag

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W12-Form Setup Toeslag	updateToeslag Handler	TOESLAG

Use case Menghapus Data Toeslag

Interface, Controller, dan Entity Classes dari Use case Retur		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama W12-Form Setup Toeslag	deleteToeslag Handler	TOESLAG

Use case Peningat Kadaluarsa

Interface, Controller, dan Entity Classes dari Use case Peningat		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama	warningED Handler	OBAT

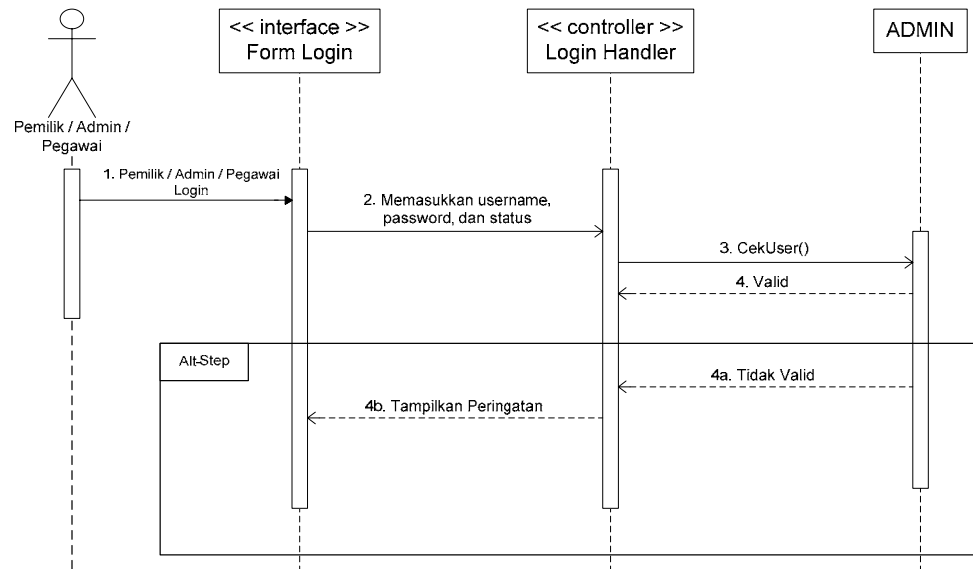
Use case Peningat Limit

Interface, Controller, dan Entity Classes dari Use case Peningat		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama	warningLimit Handler	OBAT

Use case LOG OUT

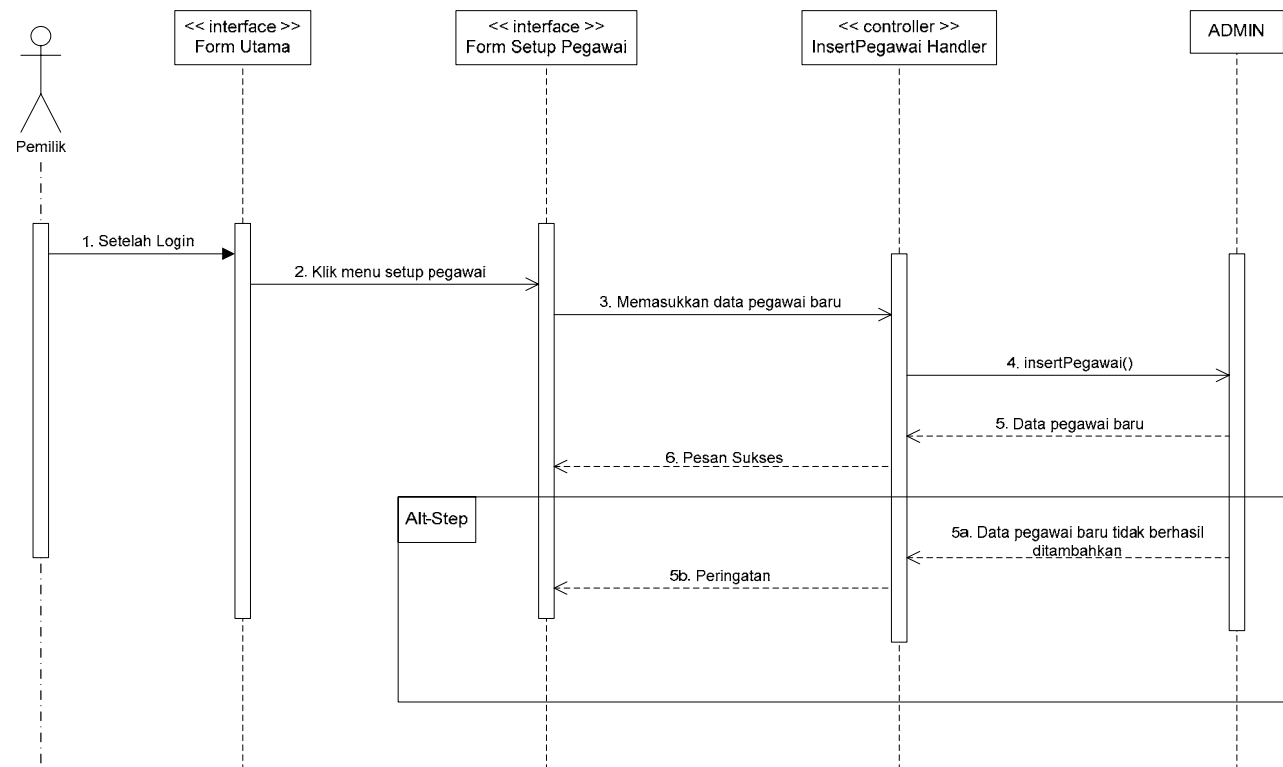
Interface, Controller, dan Entity Classes dari Use case LOG OUT		
Interface Classes	Controller Classes	Entity Classes
W02-Form Utama	LogOut Handler	ADMIN

Diagram Sequence untuk Use-Case Log in



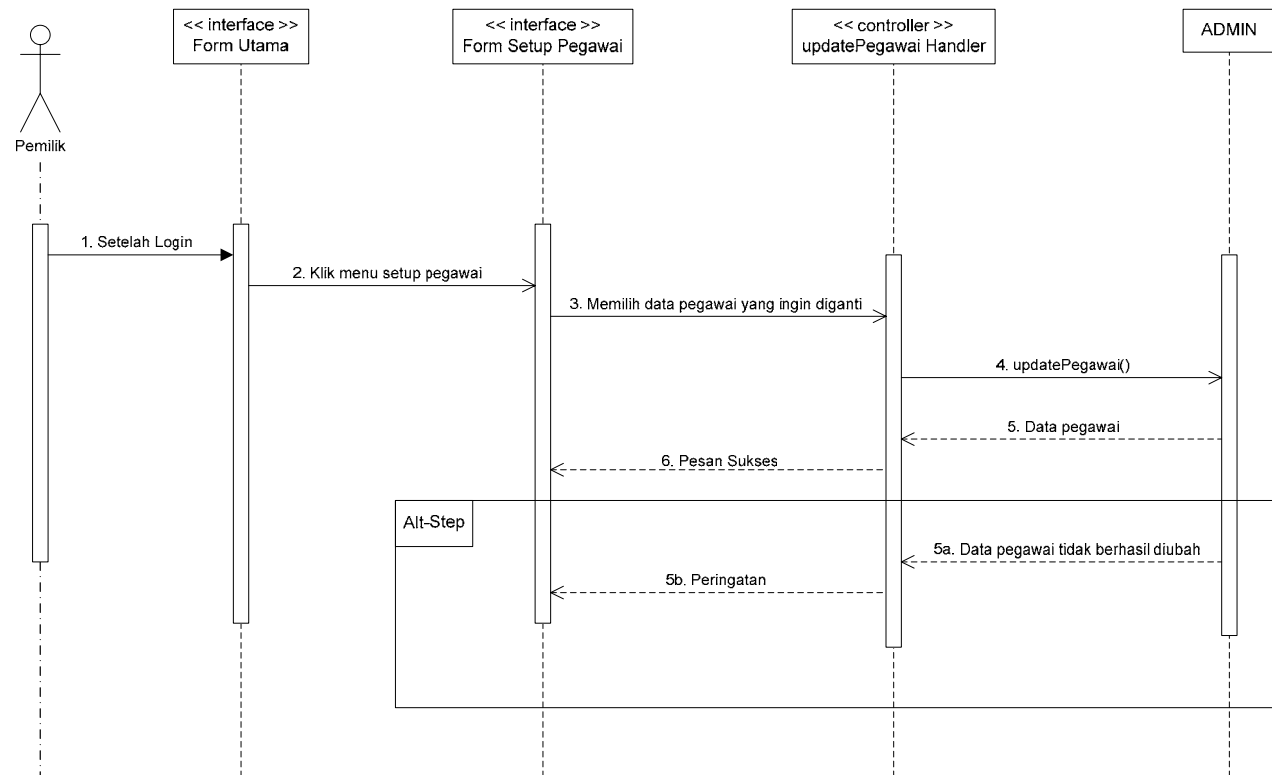
Gambar 3.31 Diagram sequence Log in

Diagram Sequence untuk Use-Case Mengisi Data Pegawai



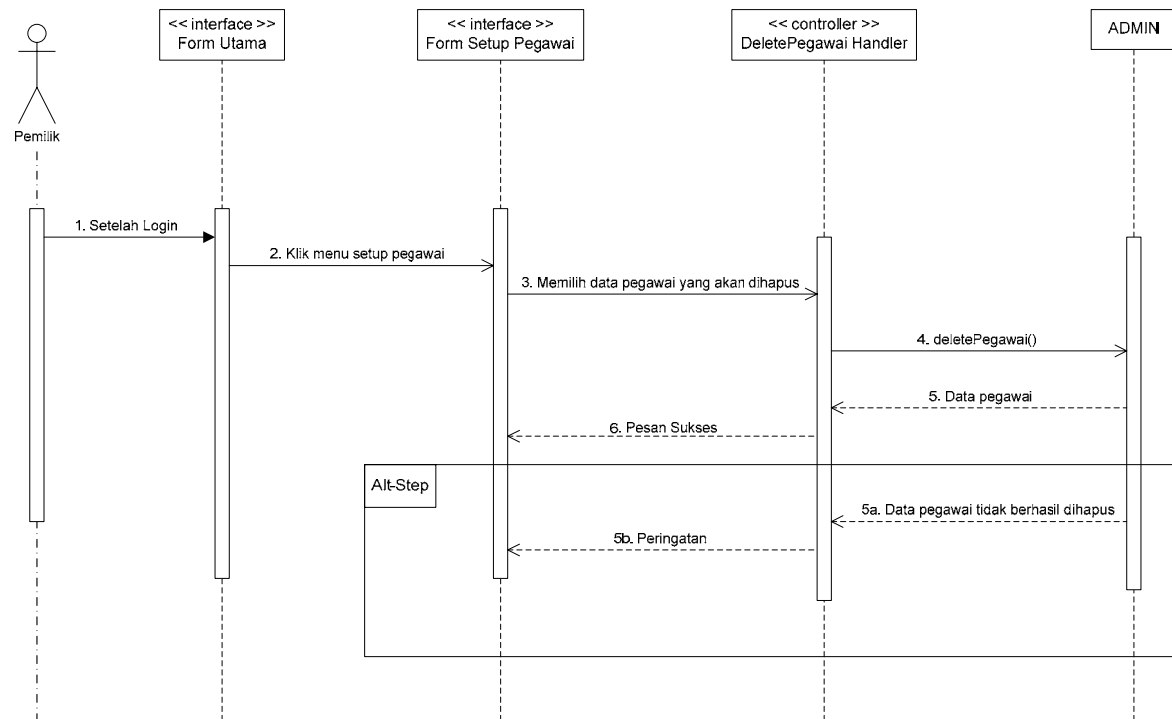
Gambar 3.32 Diagram sequence Mengisi Data Pegawai

Diagram Sequence untuk Use-Case Mengubah Data Pegawai



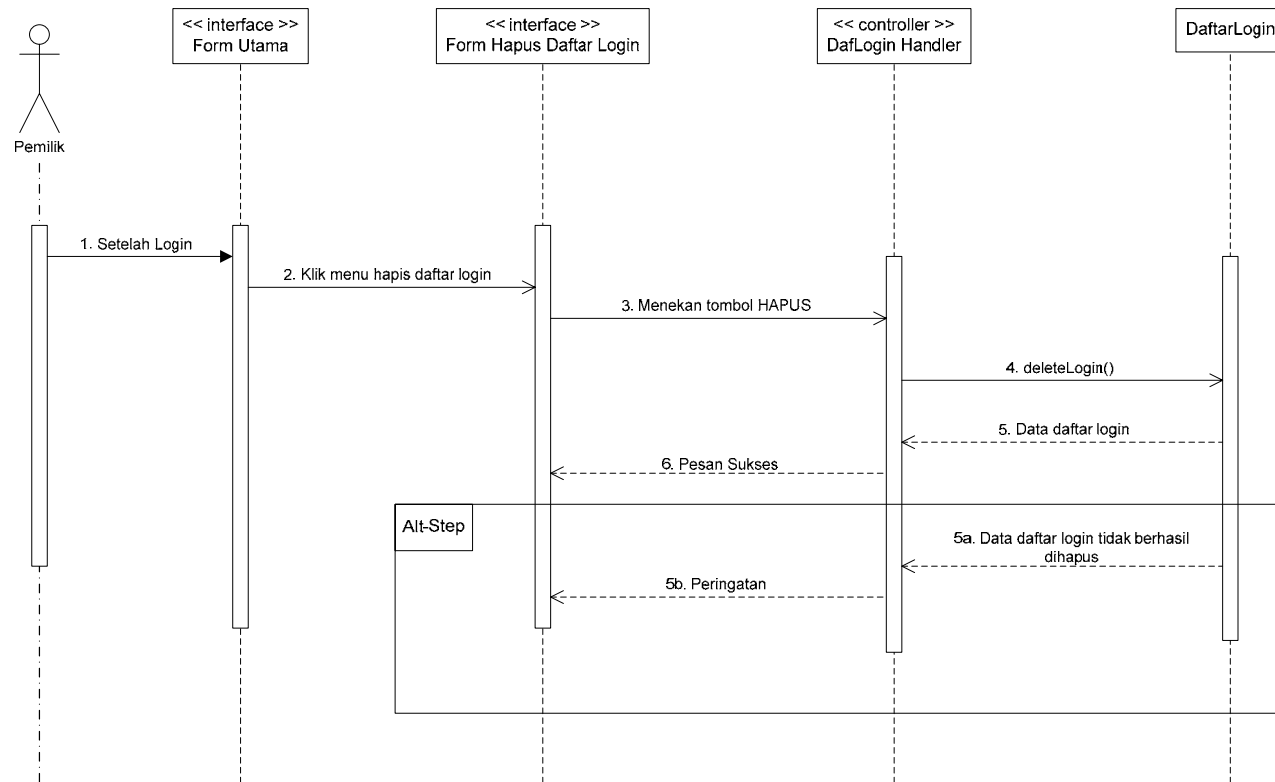
Gambar 3.33 Diagram sequence Mengubah Data Pegawai

Diagram Sequence untuk Use-Case Menghapus Data Pegawai



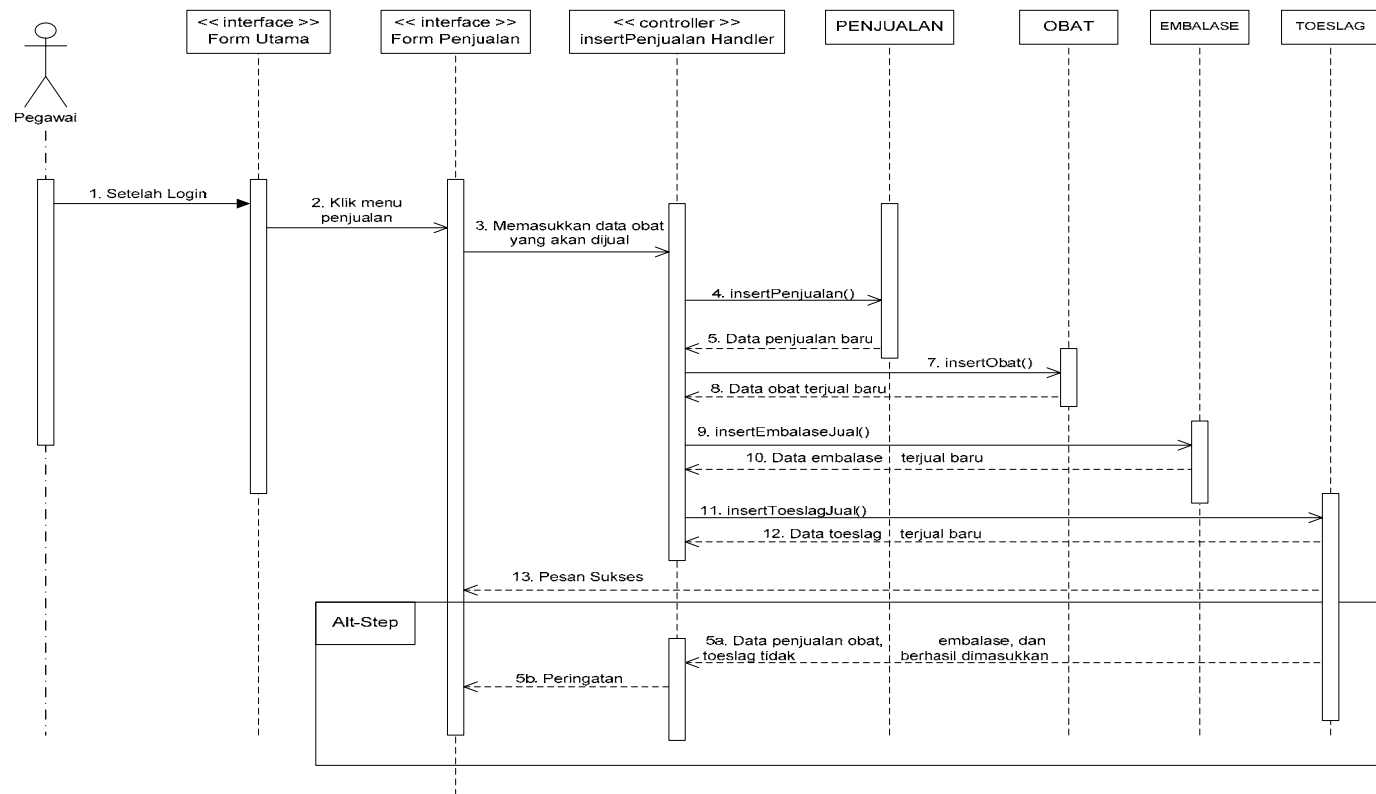
Gambar 3.34 Diagram sequence Menghapus Data Pegawai

Diagram Sequence untuk Use-Case Menghapus Daftar Login



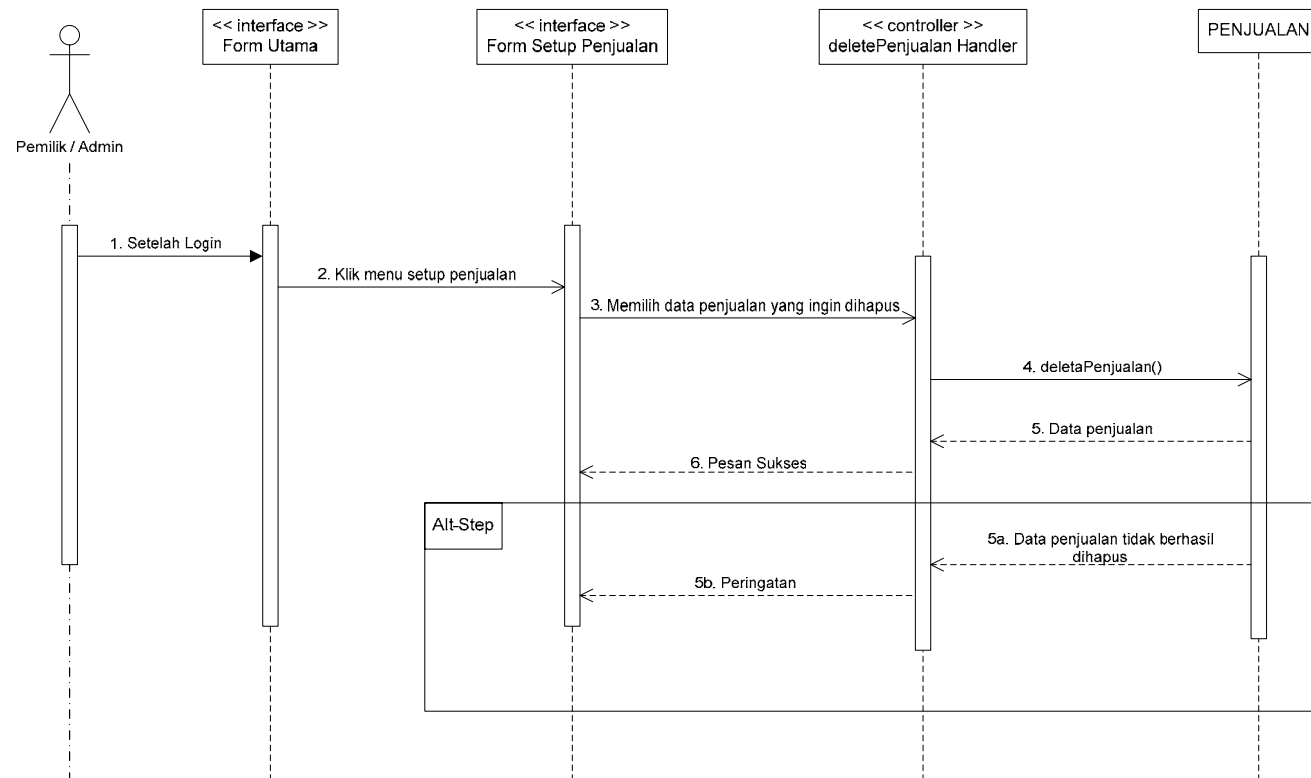
Gambar 3.35 Diagram sequence Menghapus Daftar Login

Diagram Sequence untuk Use-Case Mengisi Data Penjualan



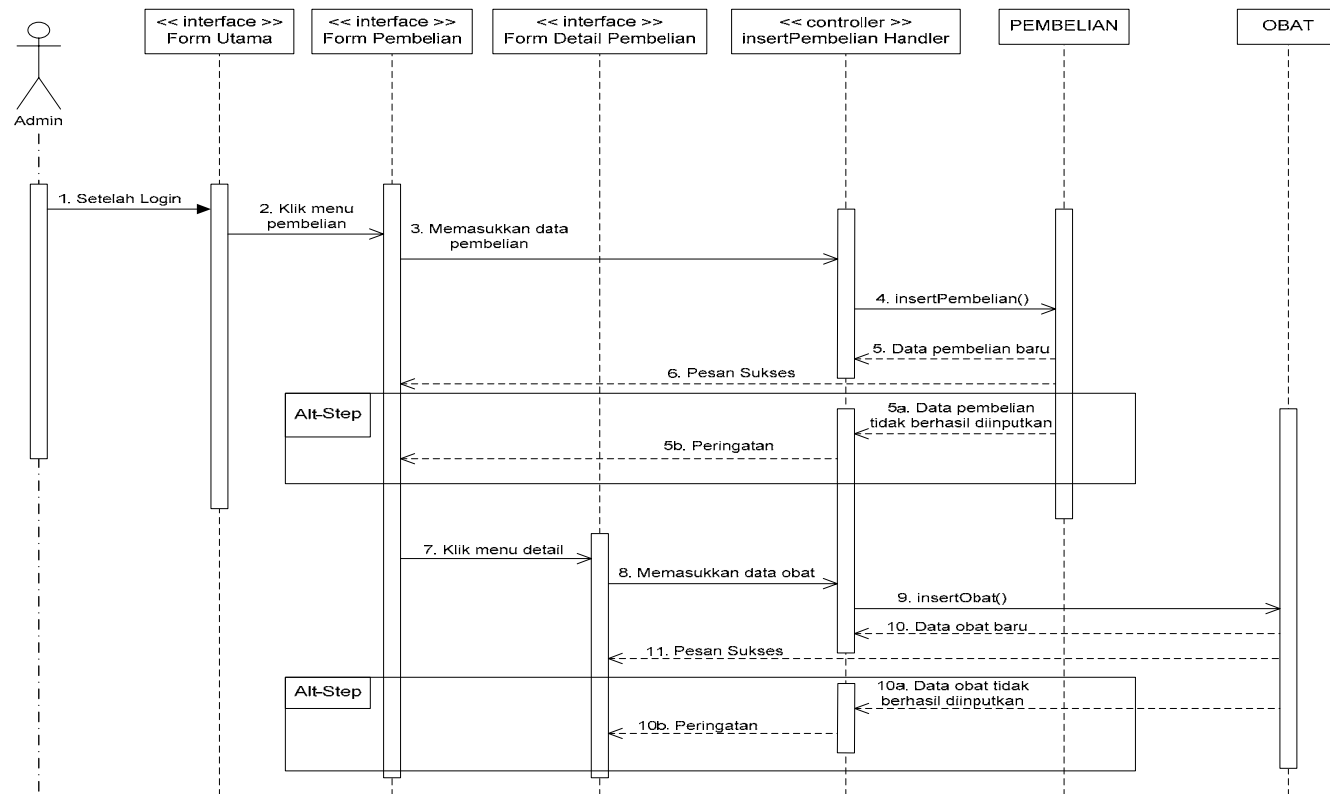
Gambar 3.36 Diagram sequence Mengisi Data Penjualan

Diagram Sequence untuk Use-Case Menghapus Data Penjualan



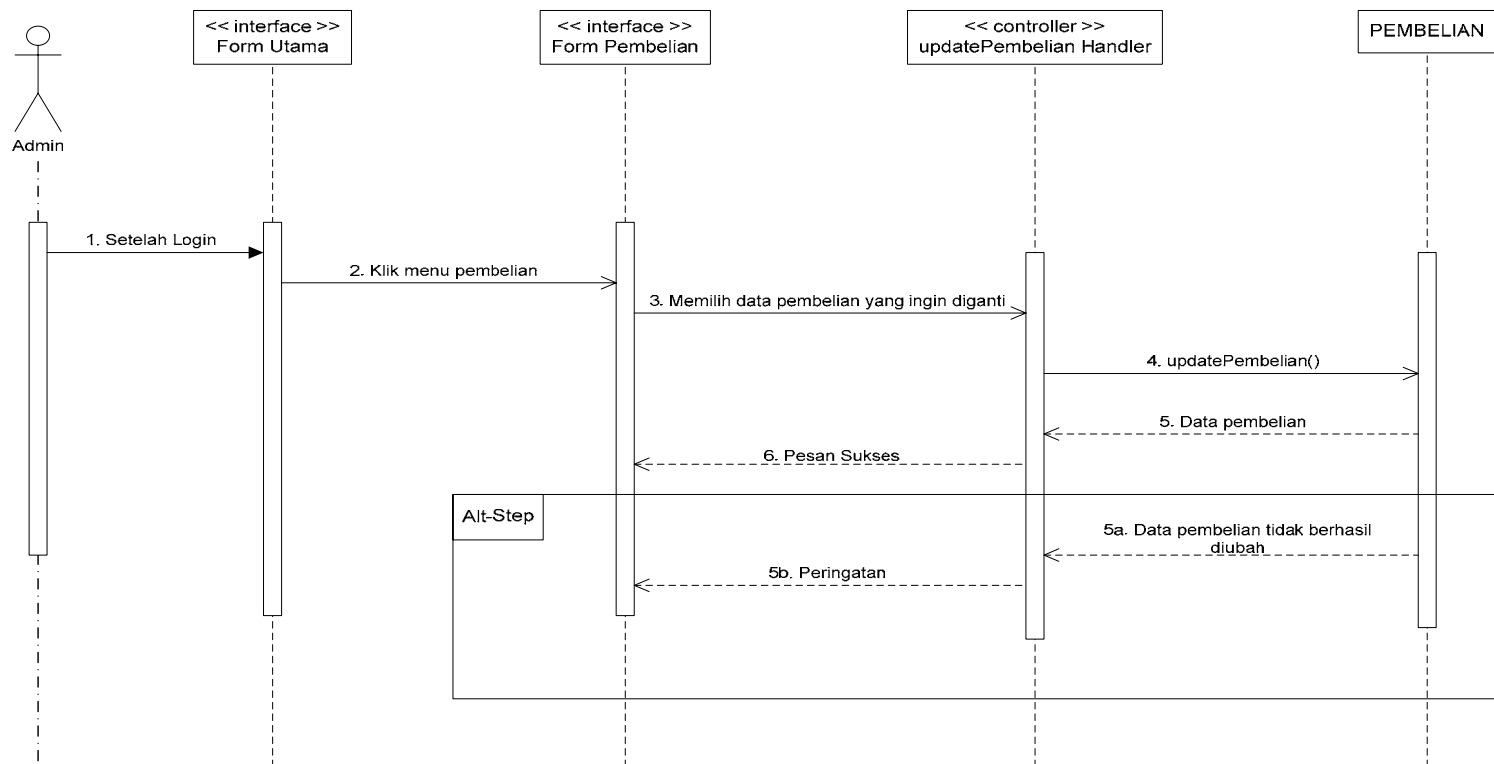
Gambar 3.37 Diagram sequence Menghapus Data Penjualan

Diagram Sequence untuk Use-Case Mengisi Data Pembelian



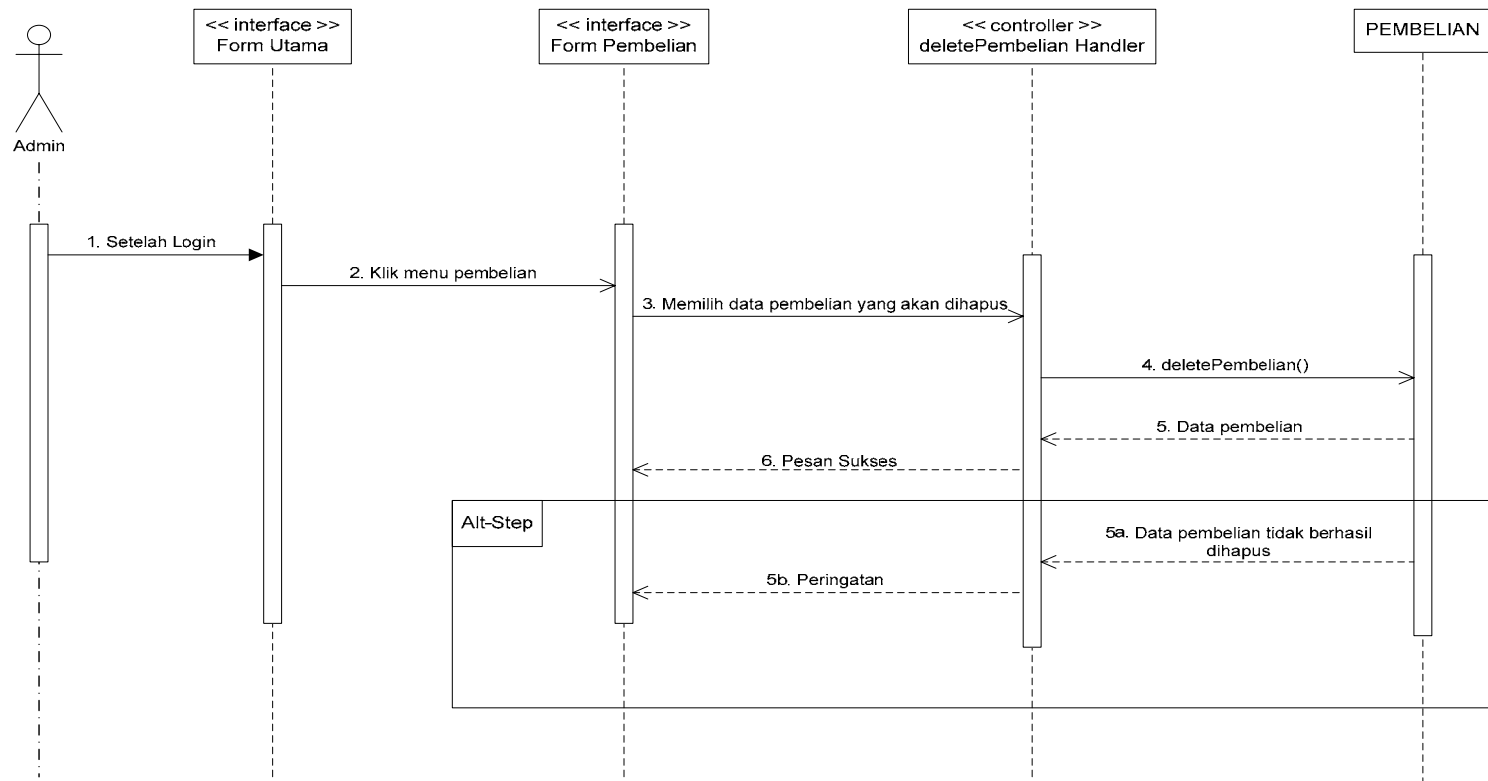
Gambar 3.38 Diagram sequence Mengisi Data Pembelian

Diagram Sequence untuk Use-Case Mengubah Data Pembelian



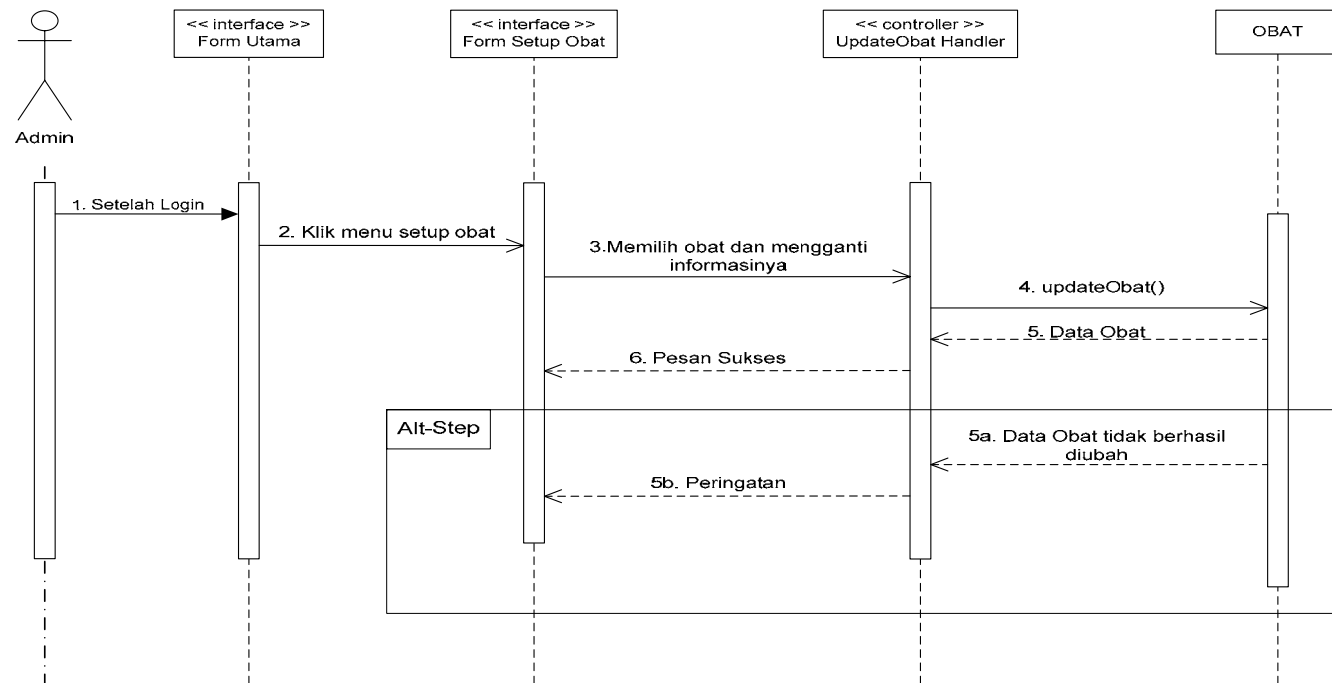
Gambar 3.39 Diagram sequence Mengubah Data Pembelian

Diagram Sequence untuk Use-Case Menghapus Data Pembelian



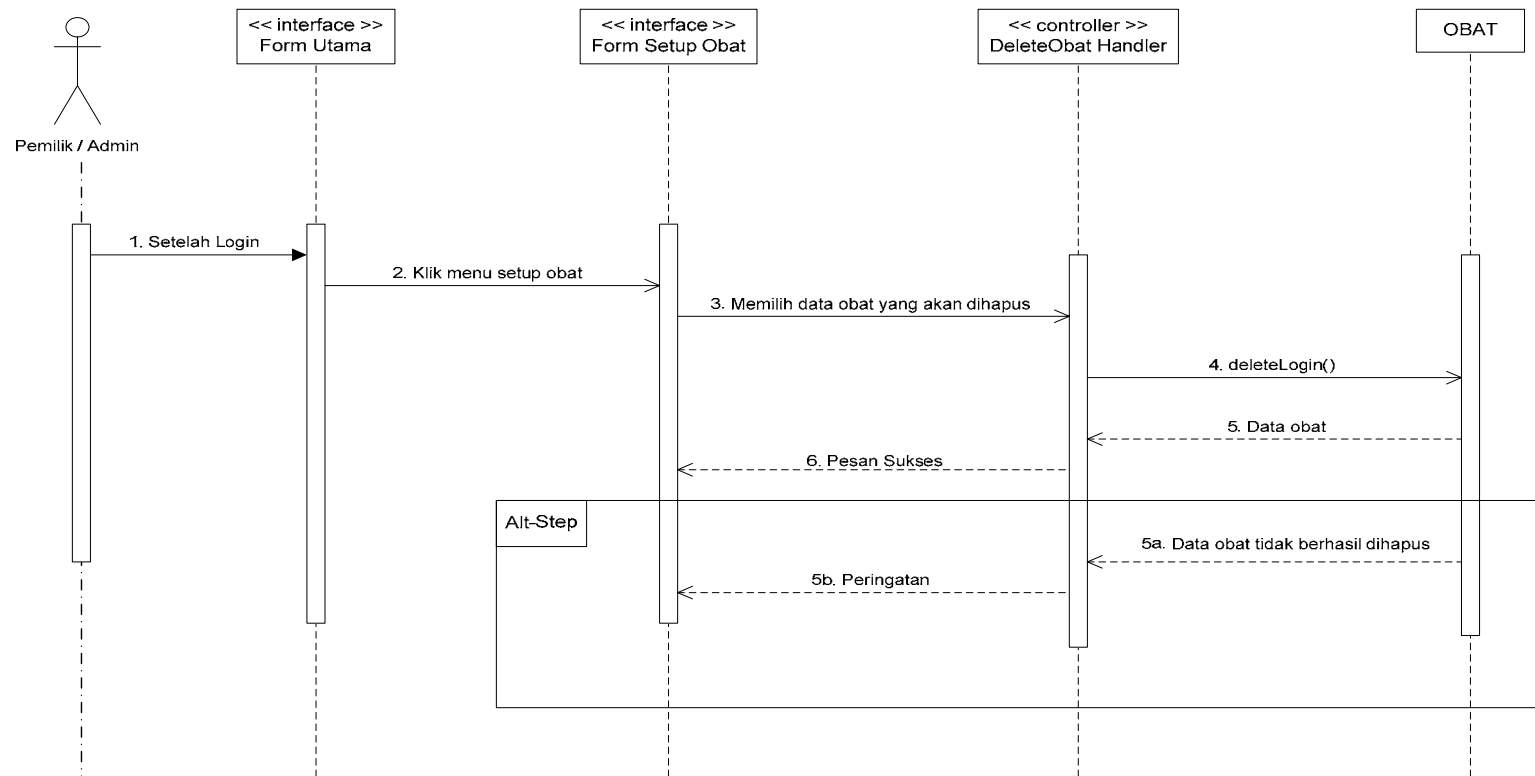
Gambar 3.40 Diagram sequence Menghapus Data Pembelian

Diagram Sequence untuk Use-Case Mengubah Data Obat



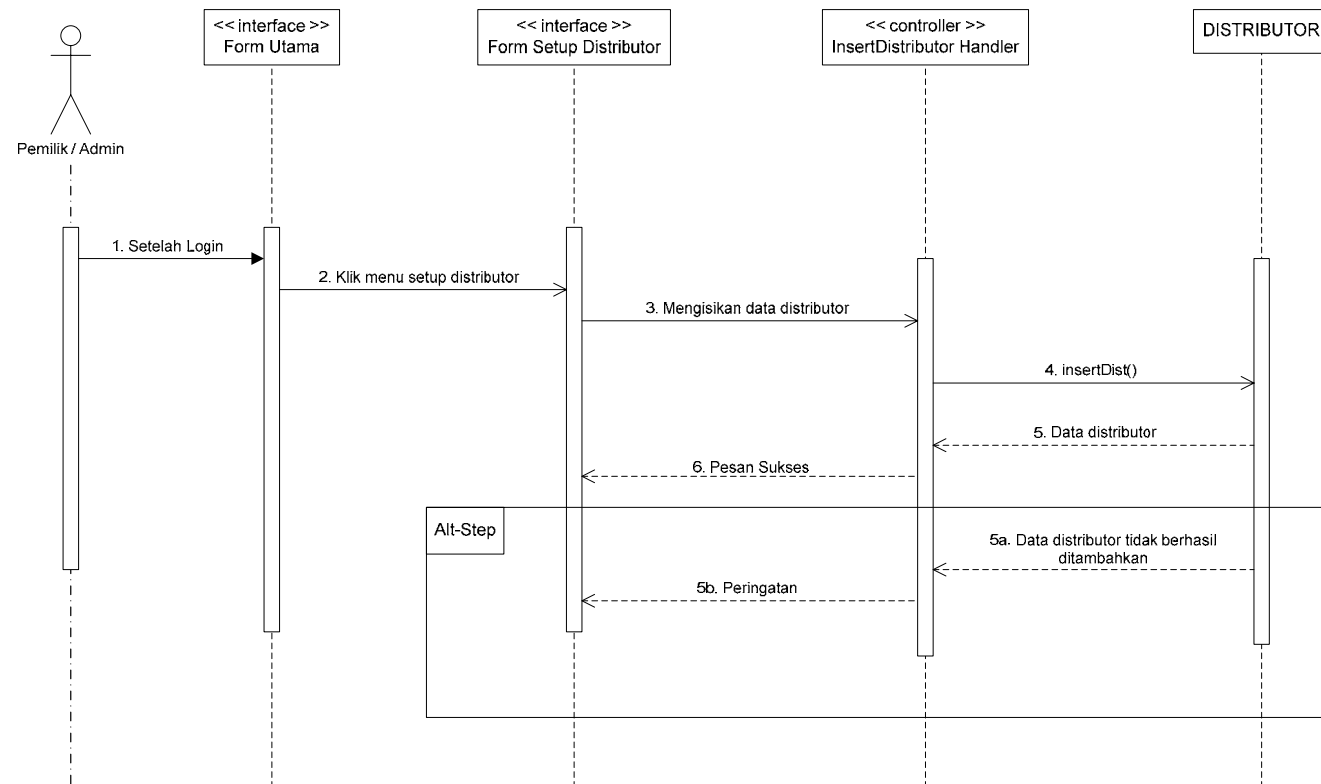
Gambar 3.41 Diagram sequence Mengubah Data Obat

Diagram Sequence untuk Use-Case Menghapus Data Obat



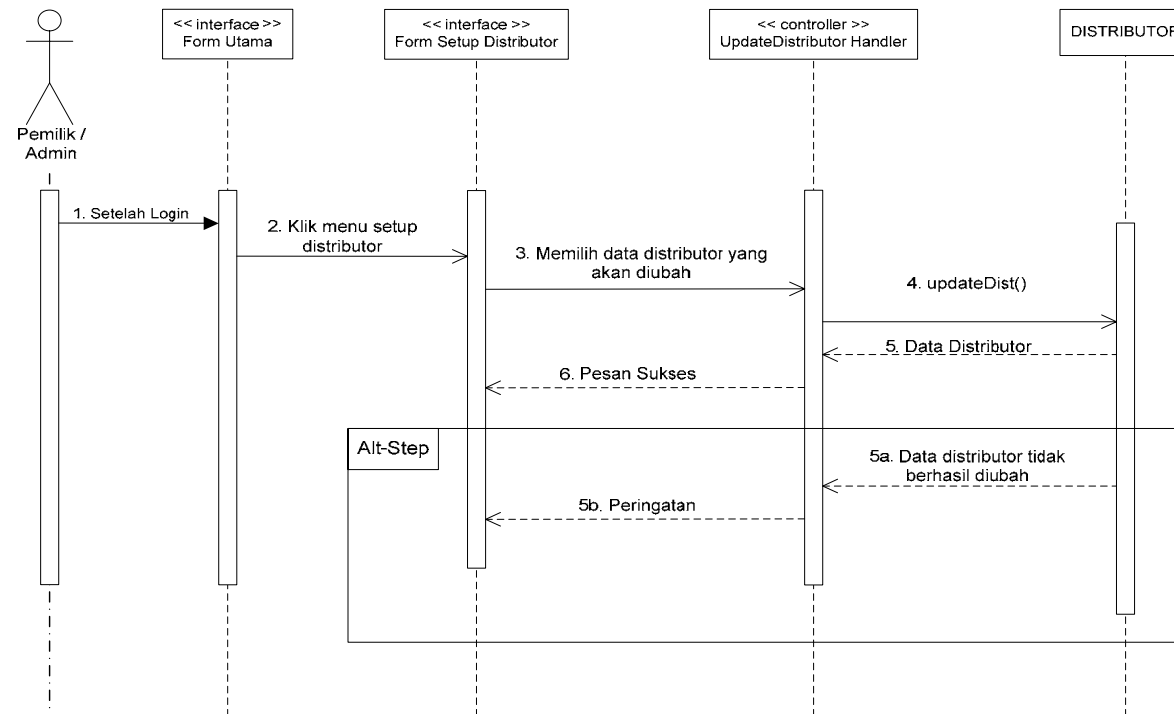
Gambar 3.42 Diagram Sequence Menghapus Data Obat

Diagram Sequence untuk Use-Case Mengisi Data Distributor



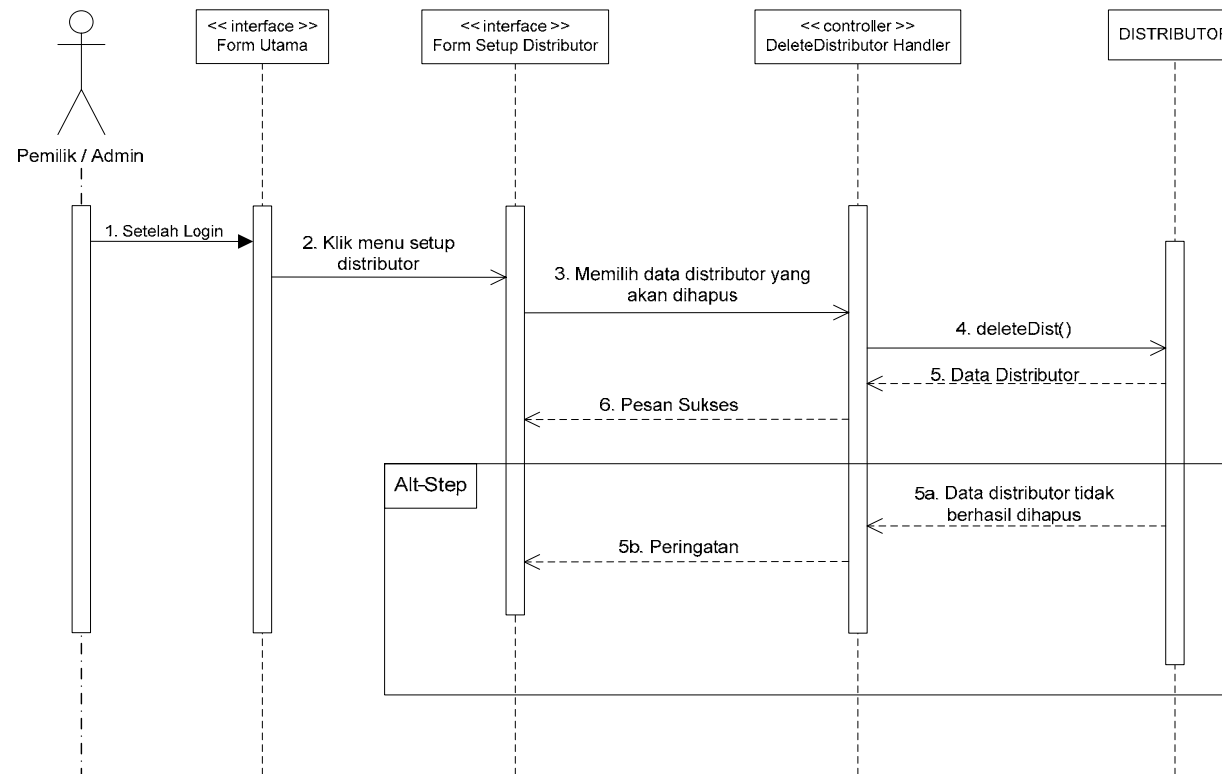
Gambar 3.43 Diagram Sequence Mengisi Data Distributor

Diagram Sequence untuk Use-Case Mengubah Data Distributor



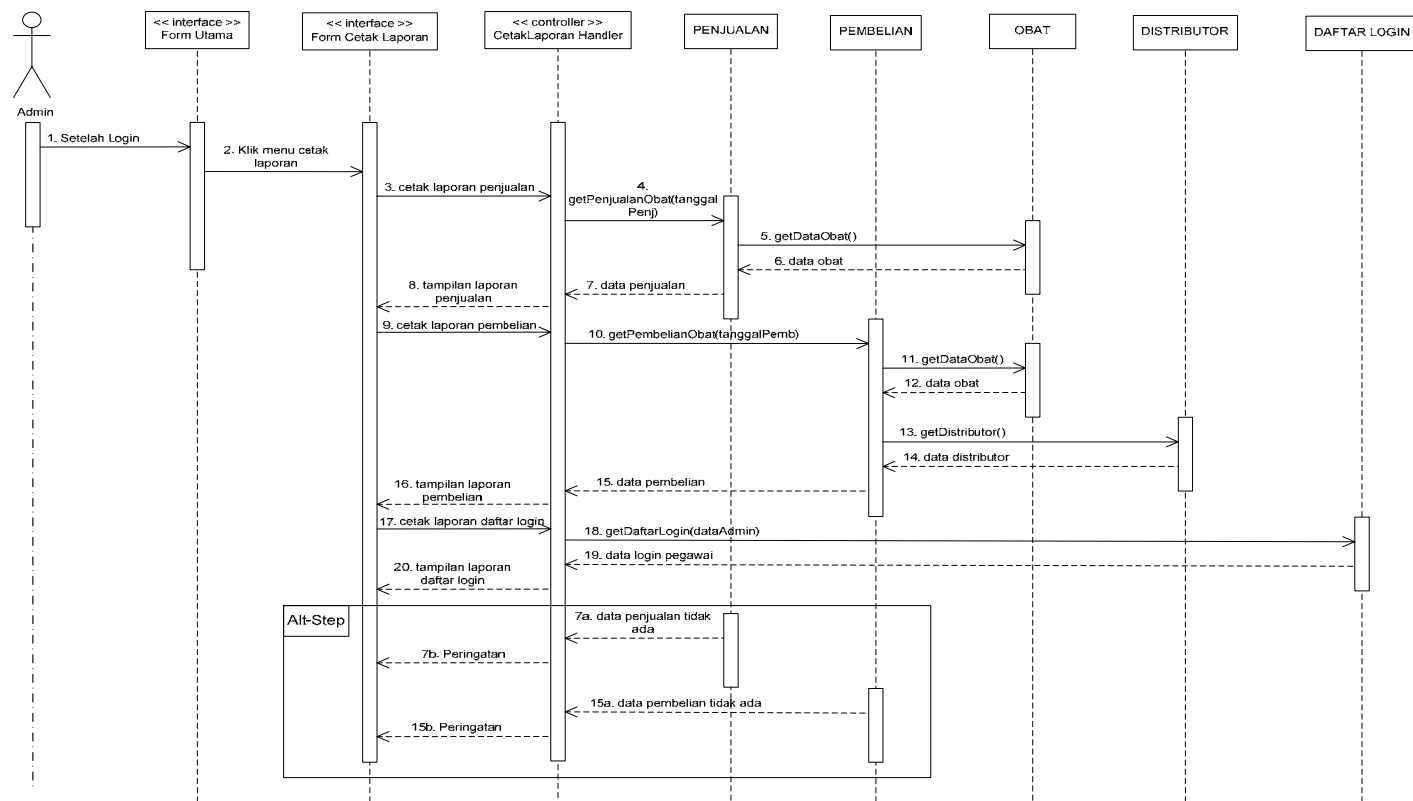
Gambar 3.44 Diagram Sequence Mengubah Data Distributor

Diagram Sequence untuk Use-Case Menghapus Data Distributor



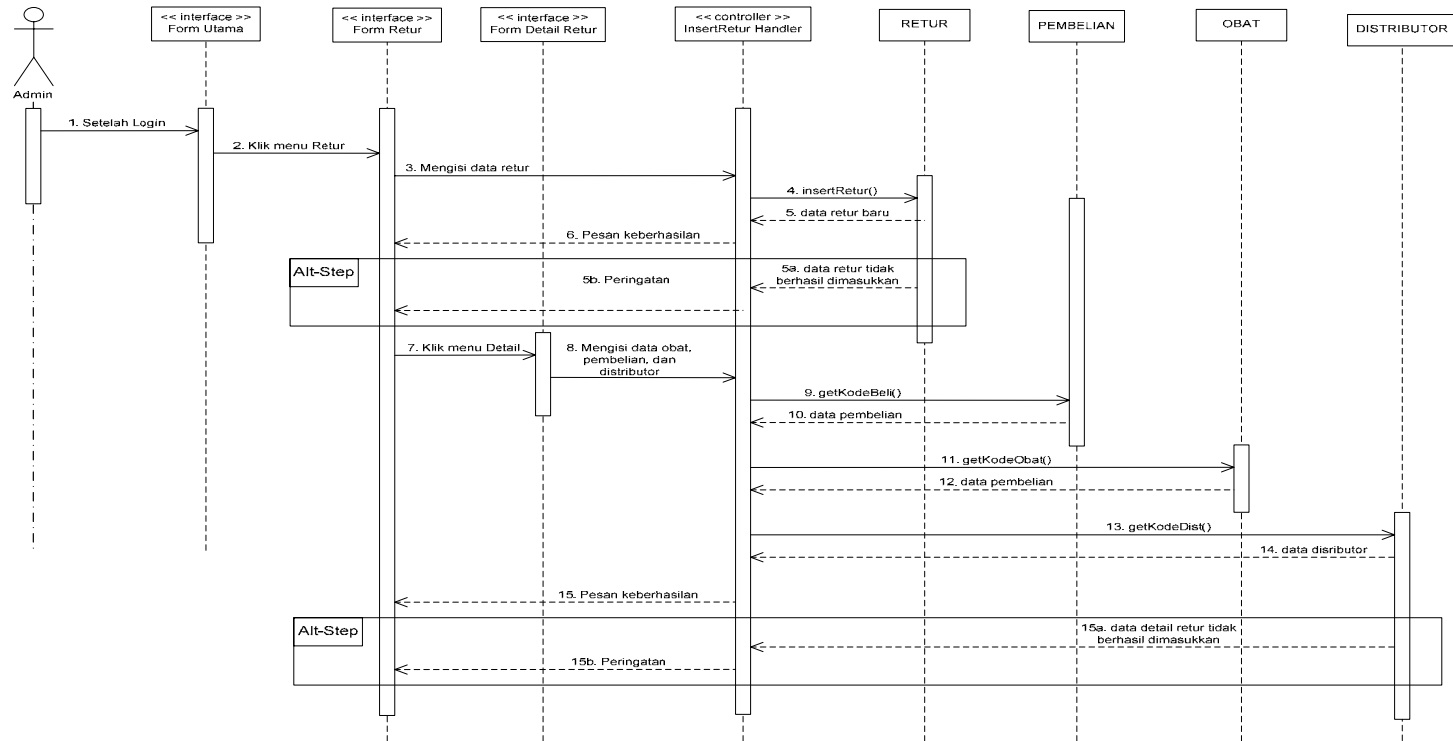
Gambar 3.45 Diagram Sequence Menghapus Data Distributor

Diagram Sequence untuk Use-Case Cetak Laporan



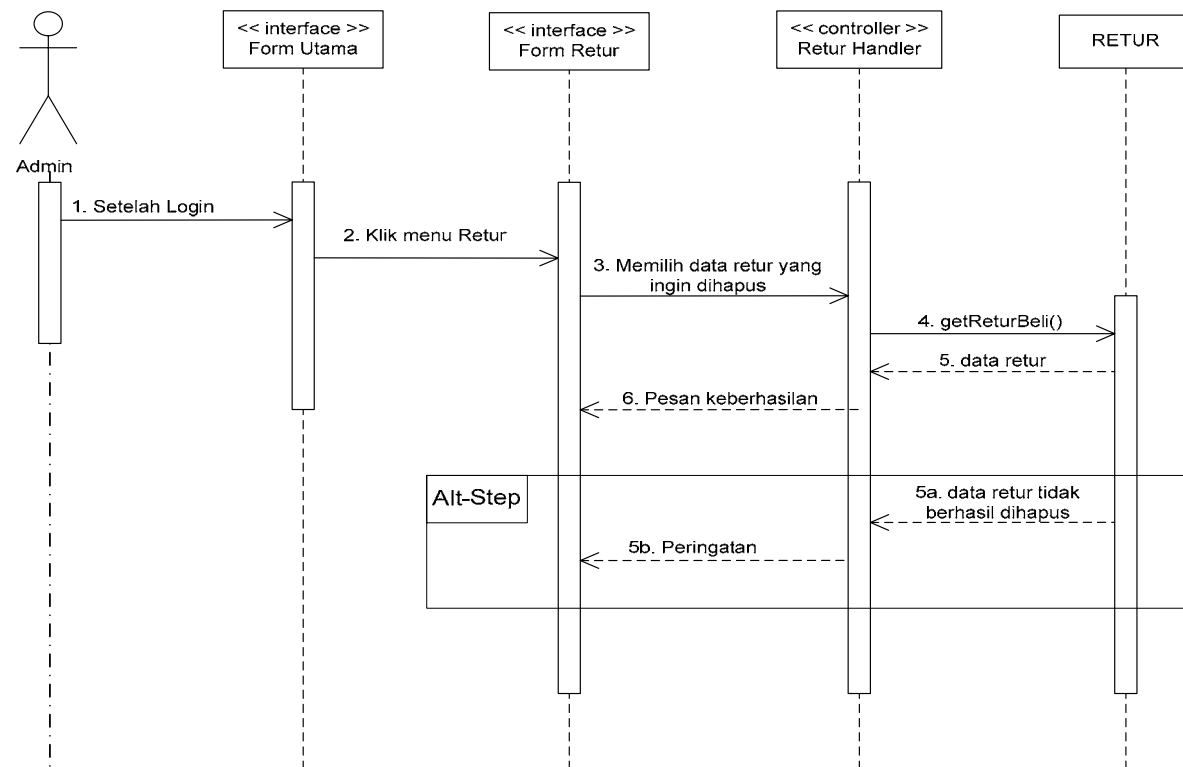
Gambar 3.46 Diagram Sequence Cetak Laporan

Diagram Sequence untuk Use-Case Mengisi Data Retur



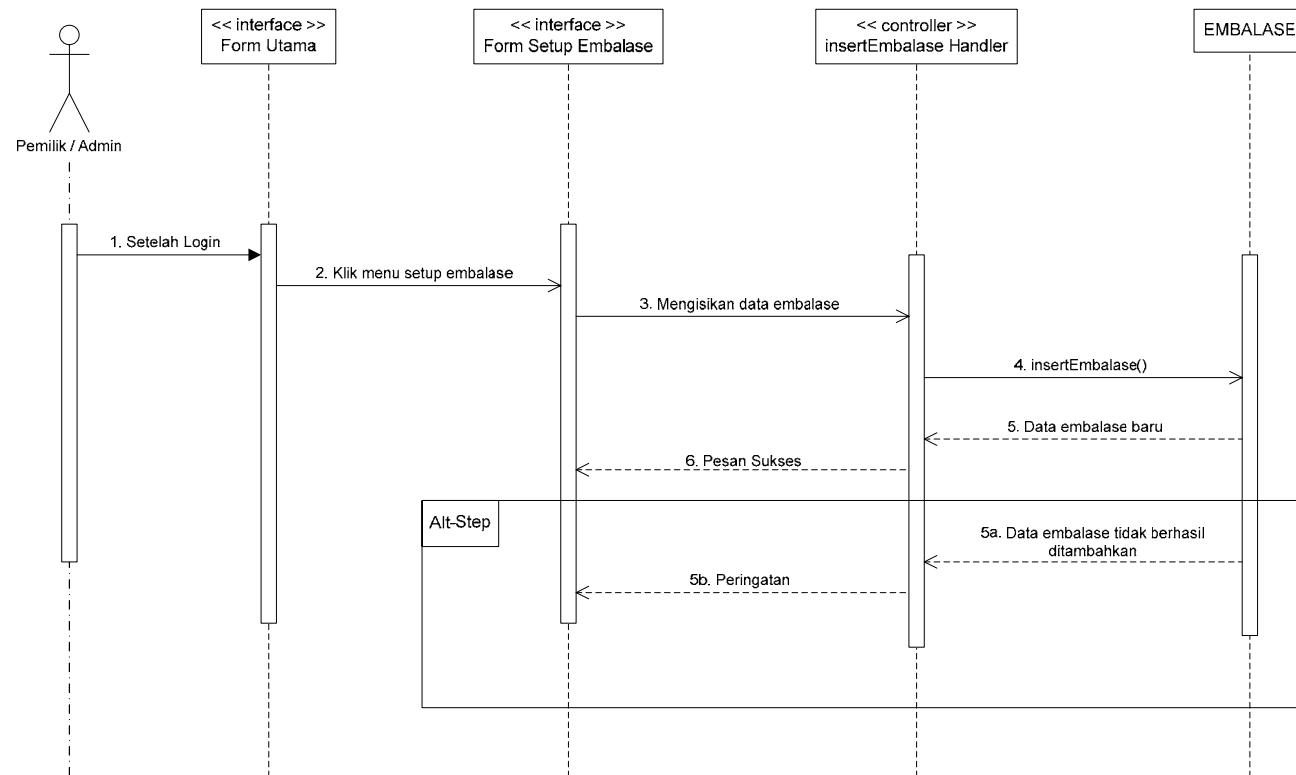
Gambar 3.47 Diagram Sequence Mengisi Data Retur

Diagram Sequence untuk Use-Case Menghapus Data Retur



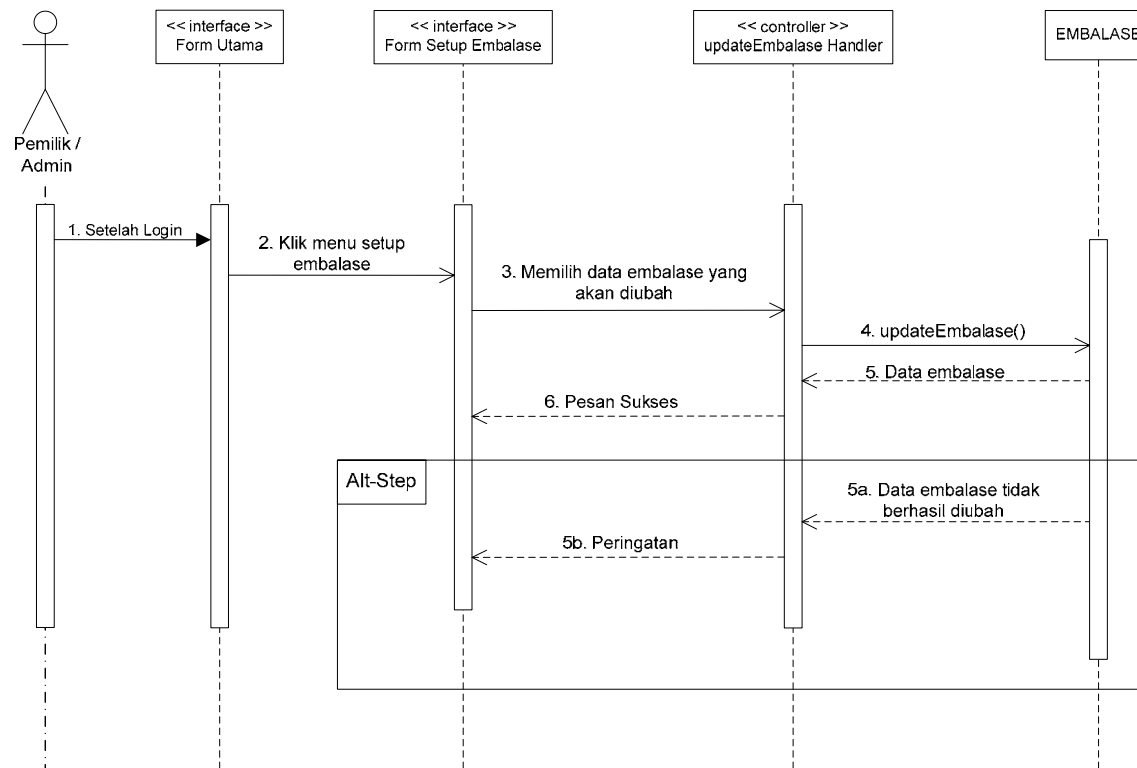
Gambar 3.48 Diagram Sequence Menghapus Data Retur

Diagram Sequence untuk Use-Case Mengisi Data Embalase



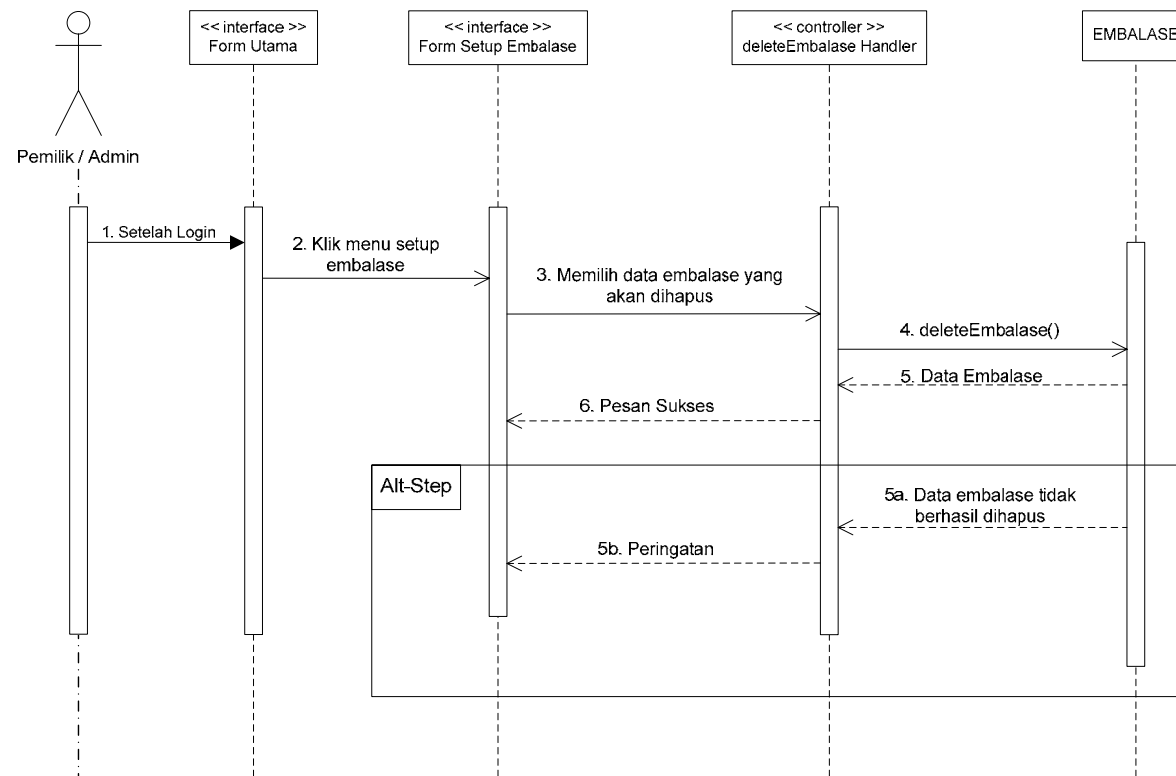
Gambar 3.49 Diagram Sequence Mengisi Data Embalase

Diagram Sequence untuk Use-Case Mengubah Data Embalase



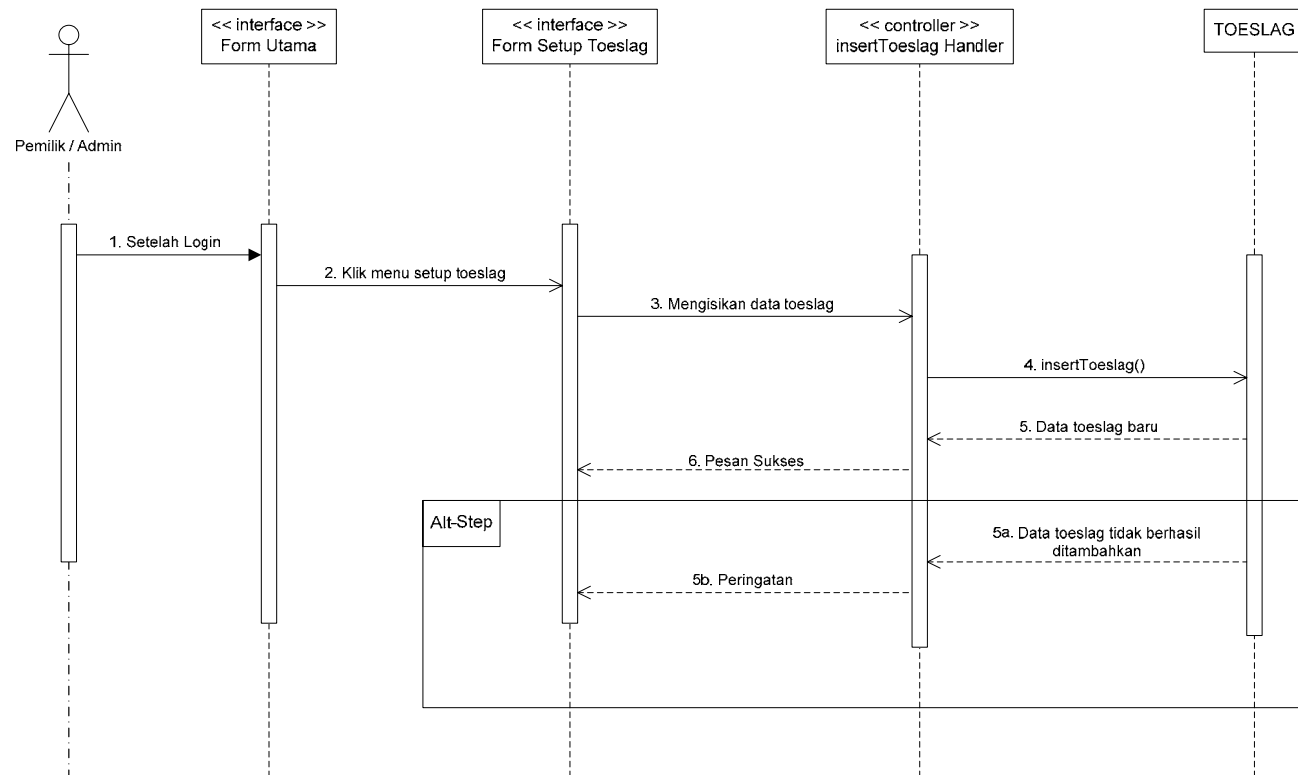
Gambar 3.50 Diagram Sequence Mengubah Data Embalase

Diagram Sequence Menghapus Data Embalase



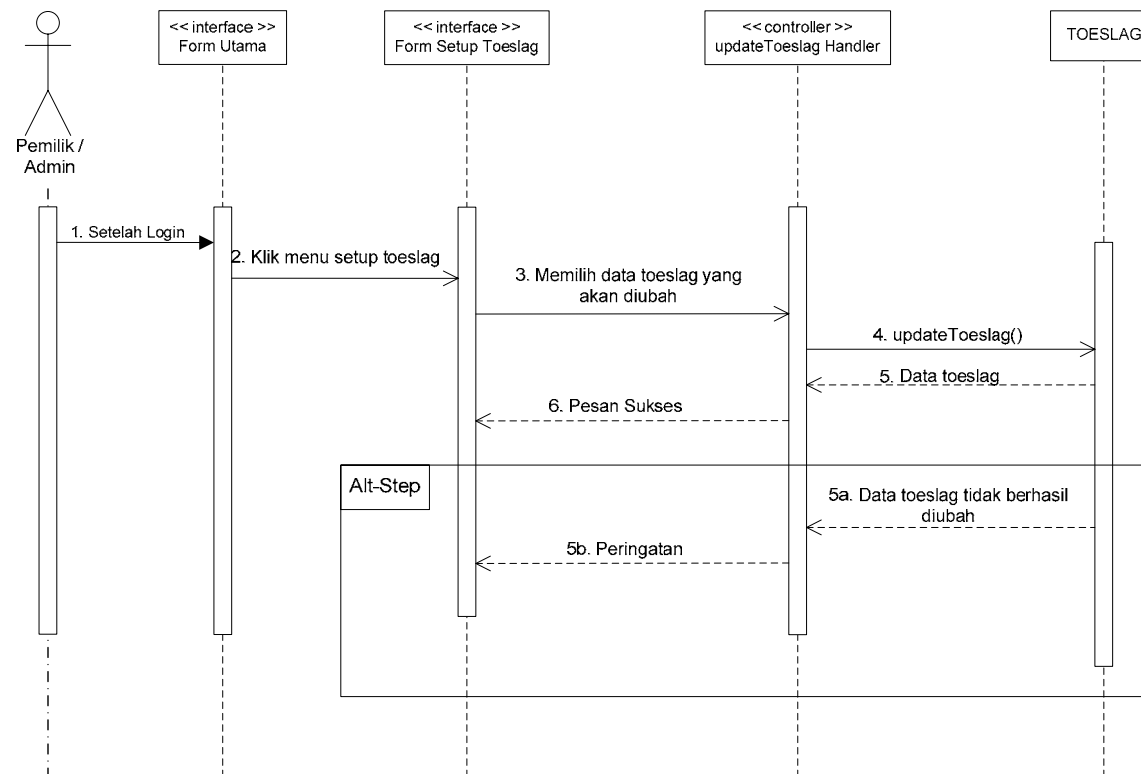
Gambar 3.51 Diagram Sequence Menghapus Data Embalase

Diagram Sequence Mengisi Data Toeslag



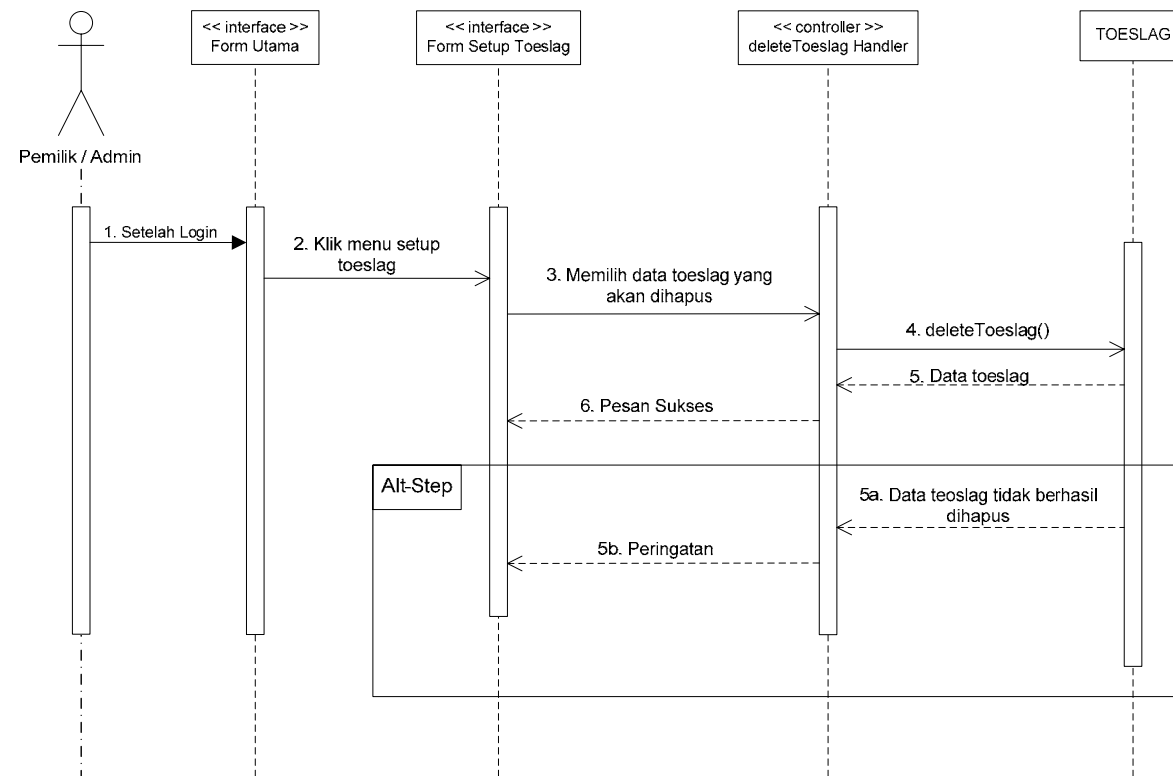
Gambar 3.52 Diagram Sequence Mengisi Data Toeslag

Diagram Sequence Mengubah Data Toeslag



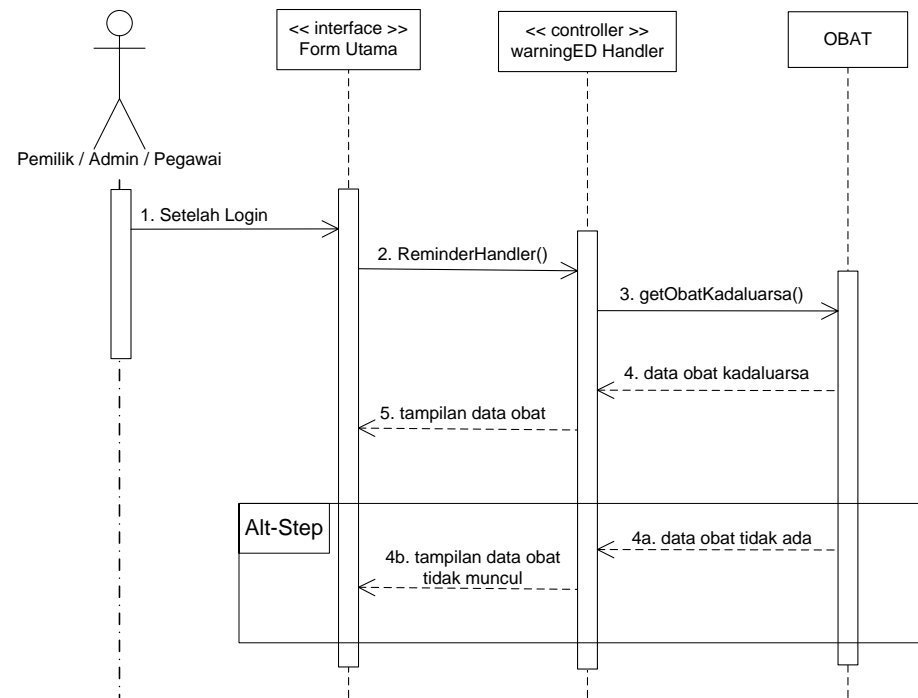
Gambar 3.53 Diagram Sequence Mengubah Data Toeslag

Diagram Sequence Menghapus Data Toeslag



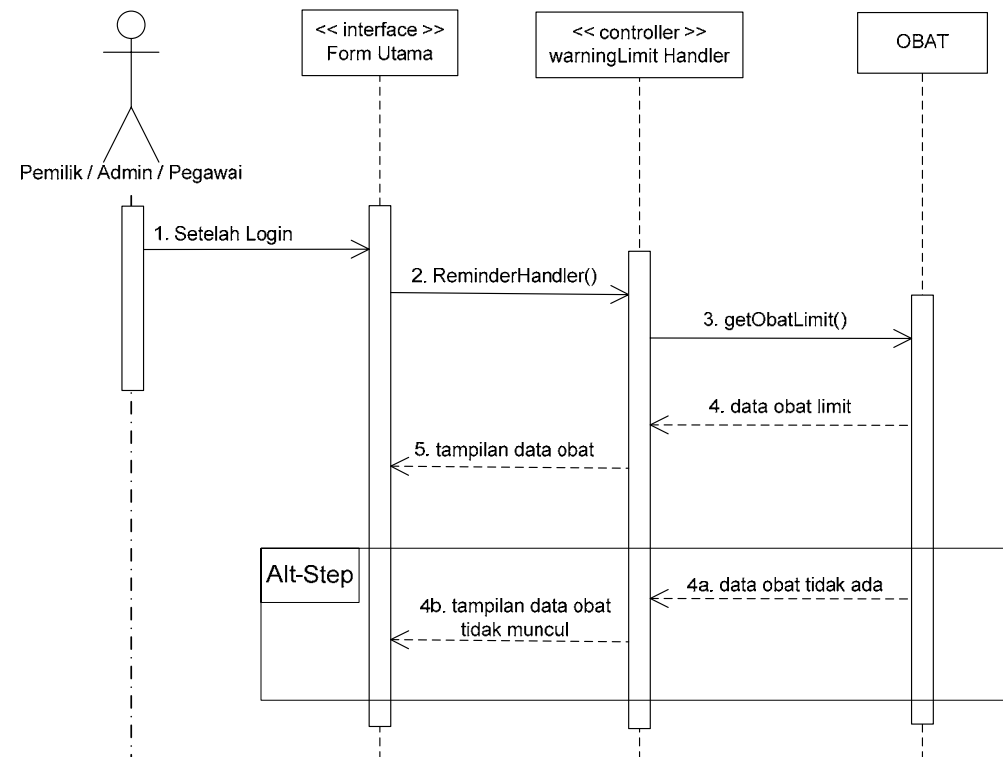
Gambar 3.54 Diagram Sequence Menghapus Data Toeslag

Diagram Sequence untuk Use-Case Peningkat Kadaluarsa



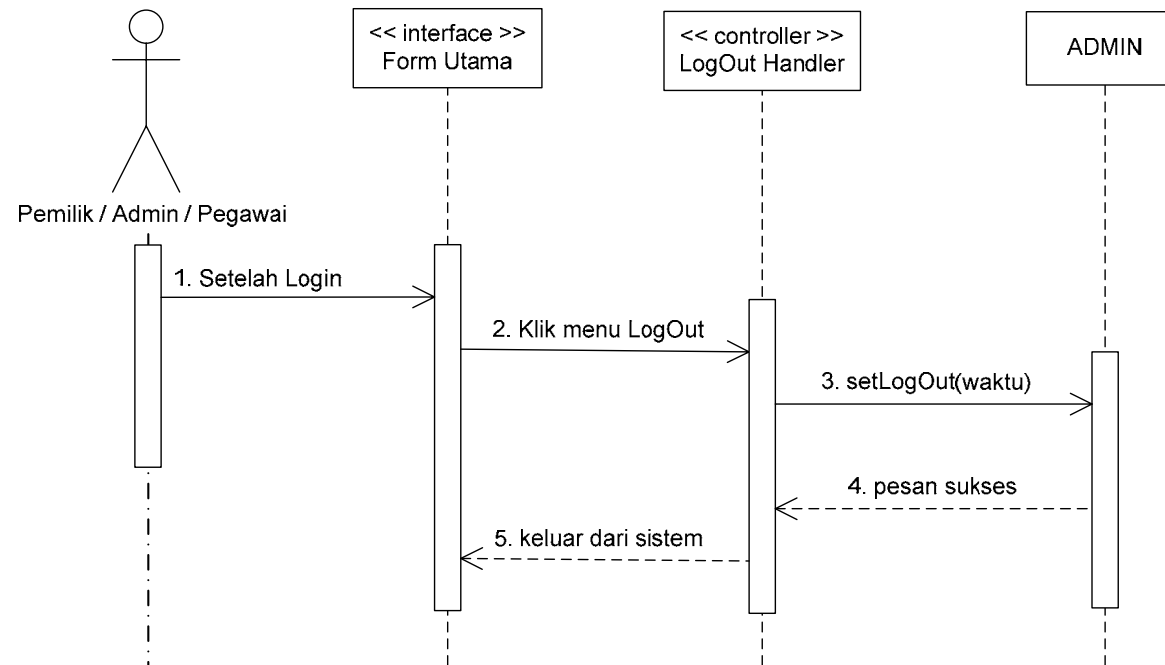
Gambar 3.55 Diagram Sequence Peningkat Kadaluarsa

Diagram Sequence untuk Use-Case Peningkat Limit



Gambar 3.56 Diagram Sequence Peningkat Limit

Diagram Sequence untuk Use-Case LOG OUT



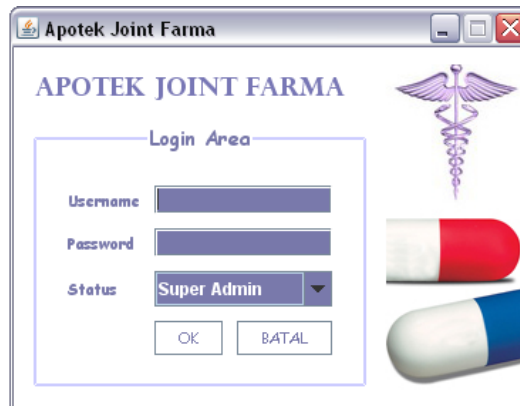
Gambar 3.57 Diagram Sequence LOG OUT

3.10. Diagram Class Desain (lampiran)

3.11. Desain User Interface

➤ Form Login

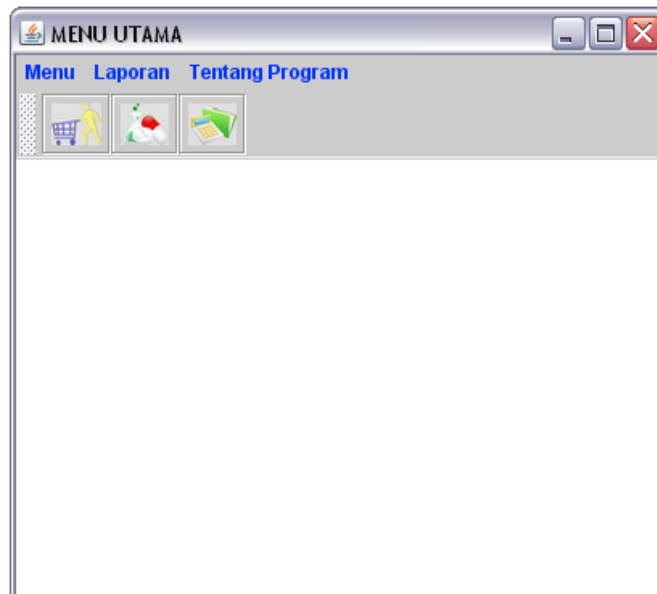
Berikut ini merupakan form yang digunakan pemilik / pegawai apotek untuk masuk ke sistem.



Gambar 3.58 Form Login

➤ Form Utama

Berikut ini merupakan form utama.



Gambar 3.59 Form Utama

➤ Form Obat-Obatan

Berikut ini merupakan form yang digunakan untuk mengubah dan menghapus data obat-obat.

OBAT-OBATAN

Tabel Data Obat

Title 1	Title 2	Title 3	Title 4

- cari berdasarkan - CARI

Detail Obat

Kode Obat : Satuan :

Nama Obat : Tanggal Pembuatan :

Jenis Obat : Psikotropika Tanggal Kadaluarsa :

Golongan : P. No.1 Harga :

SIMPAN HAPUS BATALL

Gambar 3.60 Form Obat-Obatan

➤ Form Kepegawaian

Berikut ini merupakan form yang digunakan untuk memasukkan dan menghapus data pegawai.

KEPEGAWAIAAN

Tabel Data Pegawai

Title 1	Title 2	Title 3	Title 4

Data Pegawai

Username :

Password :

Status :

SIMPAN HAPUS BATALL

Gambar 3.61 Form Kepegawaian

➤ Form Distributor

Berikut ini merupakan form yang digunakan untuk memasukkan, mengubah dan menghapus data distributor.

DISTRIBUTOR

Tabel Data Distributor

kode	nama	alamat	no telpon

- cari berdasarkan... **CARI**

Data Distributor

Nama :

Alamat :

Telpon :

SIMPAN **HAPUS** **BATAL**

Gambar 3.62 Form Disributor

➤ Form Daftar Login

Berikut ini merupakan form yang digunakan untuk menghapus data login dari pemilik / pegawai.

DAFTAR LOGIN

Tabel Daftar Login

Title 1	Title 2	Title 3	Title 4

HAPUS **BATAL**

Gambar 3.63 Form Daftar Login

➤ Form Penjualan

Berikut ini merupakan form penjualan.

PENJUALAN

Penjualan Bebas Penjualan Resep

Kata Kunci **CARI**

Data Obat

Kode Obat :

Nama Obat :

Tanggal Kadaluarsa :

Jumlah Embalase :

Jumlah Obat :

Harga Jual Obat :

Harga Jual Embalase :

OK **BATAL**

Gambar 3.64 Form Penjualan

➤ Form untuk menghitung sisa pembayaran

Berikut ini merupakan form total harga.

TOTAL

TOTAL HARGA :

DIBAYAR :

SISA :

OK

Gambar 3.65 Form Total Harga

➤ Form Penjualan dengan Resep

Berikut ini merupakan form penjualan obat menggunakan resep.

PENJUALAN

Penjualan Bebas **Penjualan Resep**

Kata Kunci **CARI**

Data Obat

Kode Obat :

Nama Obat :

Tanggal Kadaluarsa :

Nama Pasien :

Nama Dokter :

Jumlah Embalase :

Jumlah Obat :

Harga Jual Obat :

Harga Jual Embalase :

OK **BATAL**

Gambar 3.66 Form Penjualan dengan Resep

➤ Form Retur

Berikut ini merupakan form Retur.

RETUR

Data Retur

Tanggal Retur :

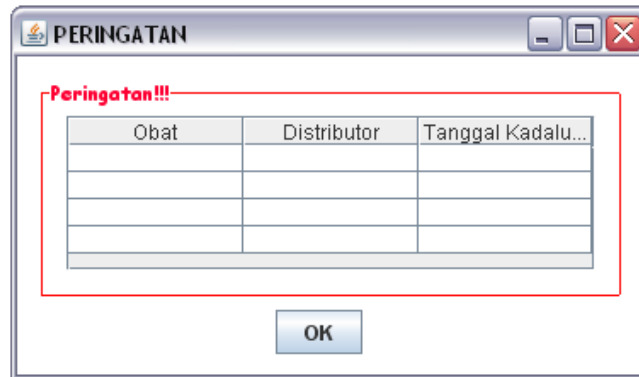
Keterangan :

CETAK FAKTUR **OK** **BATAL**

Gambar 3.67 Form Retur

➤ Form Reminder

Berikut ini merupakan form untuk mengingatkan bahwa ada obat yang hampir kadaluarsa.



PERINGATAN!!!

Obat	Distributor	Tanggal Kadalu...

OK

Gambar 3.68 Form Reminder

➤ Laporan Pembelian

Berikut ini merupakan tampilan laporan pembelian.



BUKU PEMBELIAN

NO.	NO FAKTUR	TGL FAKTUR	NAMA OBAT	STATUS OBAT	JML	BATCH	ED	HARGA SATUAN	TOTAL HARGA

Page 1 of 1

Gambar 3.69 Laporan Pembelian

➤ Laporan Register Narkotika

Berikut ini merupakan tampilan register narkotika.

BUKU REGISTER NARKOTIKA

Nama Obat :

Bulan	Persediaan Awal	PENAMBAHAN				PENGURANGAN				Sisa Akhir Bulan	Keterangan lain-lain	
		Pembelian		Resep		Resep		Nama & Alamat Pembeli				
		Tgl	Jumlah	Asal	Tgl & No Faktur	Tgl	No	Jumlah	Nama & Alamat Pembeli	Nama Dokter		

Page 1 of 1

Gambar 3.70 Laporan Pembelian

➤ Laporan Register Psikotropika

Berikut ini merupakan tampilan register psikotropika.

BUKU REGISTER PSIKOTROPIKA

Nama Obat :

Bulan	Persediaan Awal	PENAMBAHAN				PENGURANGAN				Sisa Akhir Bulan	Keterangan lain-lain	
		Pembelian		Resep		Resep		Nama & Alamat Pembeli				
		Tgl	Jumlah	Asal	Tgl & No Faktur	Tgl	No	Jumlah	Nama & Alamat Pembeli	Nama Dokter		

Page 1 of 1

Gambar 3.71 Laporan Register Psikotropika

➤ Laporan Obat Wajib Apotek (OWA)

Berikut ini merupakan tampilan laporan obat wajib apotek (owa).

NO	Tgl	Nama Pasien	Alamat Pasien	Keluhan	Nama Obat	Jmlh	Harga

Gambar 3.72 Laporan Obat Wajib Apotek (OWA)

➤ Faktur Pembelian

Berikut ini merupakan tampilan faktur pembelian.

Yogyakarta, _____

Kepada Yth _____

Banyaknya	Nama Barang	Harga Satuan	Jumlah	Keterangan

Mohon Perhatian

1. Barang yang dibeli tidak dapat dikembalikan

2. Pembayaran tidak lebih dari 1 bulan

Jumlah _____

Pengirim _____ Penerima _____

Gambar 3.73 Laporan Obat Wajib Apotek (OWA)

➤ Nota Penjualan

Berikut ini merupakan tampilan nota penjualan.

NOTA PENJUALAN

NO. NOTA: _____ Tanggal: _____

Nama Pasien : _____ Nama Dokter : _____

Alamat : _____

Kode Obat	Nama Obat	Jumlah	Harga

Total Harga _____

Dibayar _____

Sisa _____

Terima Kasih

Page 1 of 1

Gambar 3.74 Nota Penjualan

BAB IV

IMPLEMENTASI SISTEM

Implementasi merupakan tahap membangun aplikasi dari perancangan yang telah dijelaskan pada bab sebelumnya. Pada bab ini akan dijelaskan implementasi pengelolaan stok obat pada Apotek Joint Farma. Pengelolaan stok tersebut dipengaruhi oleh proses pembelian dan penjualan.

4.1. Implementasi Form Login

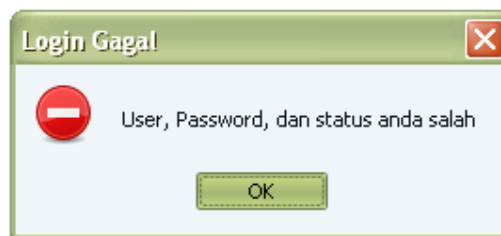
Form login merupakan form pertama yang muncul apabila program dijalankan. Form ini berisi inputan *username*, *password*, dan status. Gambar 4.1 merupakan form login.



The image shows a Windows-style login window titled "LOGIN". The window has a light green background. At the top left is a red pharmacy logo (a caduceus inside a flower-like shape). To the right of the logo, the text "APOTEK JOINT FARMA" is written in a large, green, stylized font. Below this header, there are three input fields arranged vertically. The first is labeled "Username :" and contains the text "livi". The second is labeled "Password :" and contains five black dots. The third is labeled "Status :" and is a dropdown menu with "Pemilik" selected. At the bottom of the form are two buttons: "OK" and "BATAL".

Gambar 4.1 Form Login

Pada form login diatas, agar pengguna dapat masuk ke menu utama, maka pengguna harus menginputkan *username*, *password*, dan status. Status di atas ada 3 macam yaitu pemilik, pegawai, dan admin. Seorang pemilik memiliki hak akses penuh terhadap sistem. Seorang pegawai hanya dapat mengakses menu penjualan. Dan seorang admin dapat mengakses semua menu kecuali menu setup pegawai dan menu hapus daftar login. Apabila *username*, *password*, dan status yang diinputkan tidak sesuai maka akan muncul peringatan. Gambar 4.2 merupakan peringatan gagal login.



Gambar 4.2 Peringatan Gagal Login

4.2. Implementasi Form Utama

Setelah melakukan login maka pengguna sistem akan masuk ke menu utama. Pada menu utama ini terdapat 4 pilihan yaitu menu, setup, cetak, dan bantuan. Pada pilihan menu terdapat 7 submenu, yaitu penjualan, pembelian, retur, hapus daftar login, daftar obat kadaluarsa, daftar obat limit, dan keluar. Pada pilihan setup terdapat 6 submenu, yaitu setup data distributor, setup data pegawai, setup embalase, setup toeslag, setup obat, setup penjualan. Pada pilihan cetak terdapat 7 submenu, yaitu laporan penjualan, laporan pembelian, laporan obat, laporan obat wajib apotek, narkotika, psikotropika, dan etiket. Pada pilihan bantuan terdapat 2 submenu, yaitu

bantuan program dan tentang program. Beberapa menu tersebut di atas dapat dilihat pada gambar-gambar berikut:



Gambar 4.3 form utama



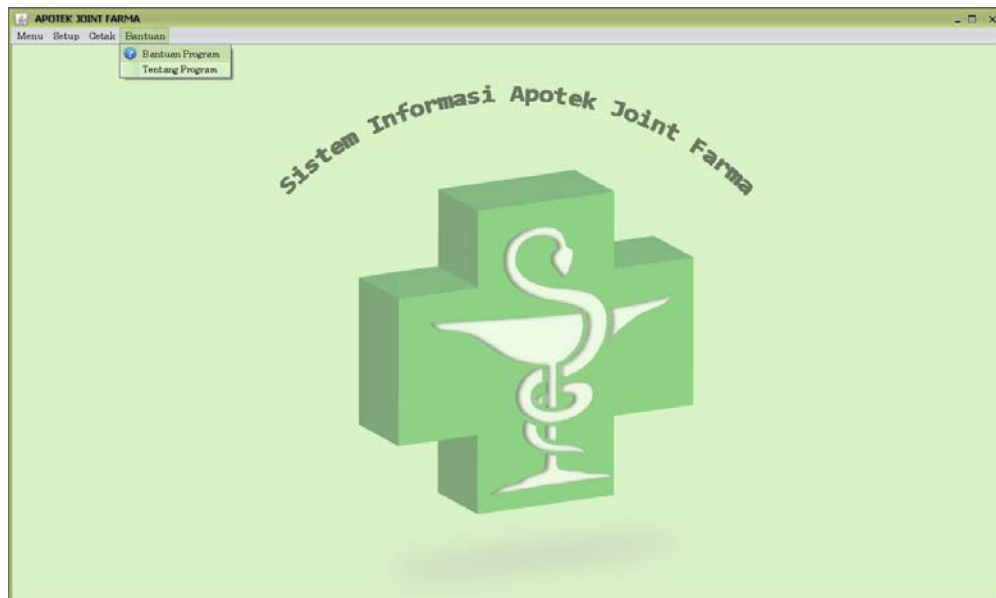
Gambar 4.4 form utama pilihan menu



Gambar 4.5 form utama pilihan setup



Gambar 4.6 form utama pilihan cetak



Gambar 4.7 form utama pilihan Bantuan

4.3. Implementasi Form Pembelian

Form pembelian ini digunakan untuk mencatat data pembelian obat yang masuk. Pada form ini terdapat 5 tombol, yaitu: tambah, simpan, hapus, detail, dan batal. Gambar 4.8 merupakan tampilan form pembelian.

PEMBELIAN

F4-Cari Data Distributor

Tabel Data Pembelian

Kode Pembelian	Kode Distributor	No Faktur	Tanggal Faktur	No Surat Pesan	Tanggal Surat Pesan	Potongan(%)
1	dist01	fak01	21-08-2009	sp01	16-08-2009	20
2	dist02	fak02	30-01-2009	sp02	26-01-2009	25
3	dist03	fak03	22-08-2009	sp03	19-08-2009	20
4	dist03	fak04	24-08-2009	sp05	16-08-2009	50

Kode Pembelian : No. Surat Pesan :

Kode Distributor : Tanggal Surat Pesan : August 29, 2009

No. Faktur : Potongan :

Tanggal Faktur : August 29, 2009

Buttons: TAMBAH, SIMPAN, HAPUS, DETAIL, BATAL

Calendar: August 2009

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
31							1
32	2	3	4	5	6	7	8
33	9	10	11	12	13	14	15
34	16	17	18	19	20	21	22
35	23	24	25	26	27	28	29
36	30	31					

Gambar 4.8 form pembelian

Pada halaman ini terdapat 4 proses, yaitu:

1. Proses Penambahan Data Pembelian.

Dalam proses ini admin harus menekan tombol tambah kemudian mengisi data-data yang ingin dimasukkan, setelah itu admin menekan tombol simpan untuk menyimpan data-data tersebut. Untuk mengisi kode distributor, admin hanya perlu menekan tombol F4 untuk mencari data distributor yang diinginkan. Gambar 4.9 merupakan tampilan form untuk mencari data distributor.

2. Proses Pengubahan Data Pembelian.

Dalam proses ini admin harus mengarahkan kursor ke tabel yang tersedia, kemudian klik data yang ingin diubah. Data-data tersebut akan muncul pada *textfield* yang tersedia, kemudian admin dapat melakukan pengubahan data. Untuk menyimpan perubahan data tersebut admin menekan tombol simpan.

3. Proses Penghapusan Data Pembelian.

Dalam proses ini admin harus mengarahkan kursor ke tabel yang tersedia, kemudian klik data yang ingin dihapus. Data-data tersebut akan muncul pada *textfield* yang tersedia setelah itu admin menekan tombol hapus.

4. Proses Penambahan Data Obat yang Dibeli.

Dalam proses ini admin harus menekan tombol detail kemudian akan muncul form detail pembelian yang digunakan untuk mengisi data obat yang dibeli. Form detail pembelian ini dapat dilihat pada gambar 4.10.

Pada form detail pembelian ini terdapat 3 proses, yaitu:

a. Proses penambahan data obat.

Dalam proses ini admin harus menekan tombol tambah kemudian mengisi data-data yang ingin dimasukkan, setelah itu admin menekan tombol simpan untuk menyimpan

data-data tersebut. Apabila data obat yang ingin diinputkan sudah ada di database, maka admin hanya perlu menekan tombol F5 pada *textfield* kode obat, sehingga akan muncul form untuk mencari data obat. Gambar 4.11 merupakan tampilan form untuk mencari data obat.

b. Proses pengubahan data obat.

Dalam proses ini admin harus mengarahkan kursor ke tabel yang tersedia, kemudian klik data yang ingin diubah. Data-data tersebut akan muncul pada *textfield* yang tersedia, kemudian admin dapat melakukan pengubahan data. Untuk menyimpan perubahan data tersebut admin menekan tombol simpan.

c. Proses penghapusan data obat.

Dalam proses ini admin harus mengarahkan kursor ke tabel yang tersedia, kemudian klik data yang ingin dihapus. Data-data tersebut akan muncul pada *textfield* yang tersedia setelah itu admin menekan tombol hapus. Admin harus berhati-hati melakukan proses ini, karena proses ini akan mengakibatkan stok obat tidak sama. Dan untuk mengatasi hal tersebut, maka pemilik atau admin perlu melakukan penjualan counter sejumlah obat yang dihapus. Kemudian admin harus mencetak nota penjualan.

PENCARIAN DATA DISTRIBUTOR

KATEGORI : kode_dist ▼

KATA KUNCI :

—Data Distributor—

Kode Distributor	Nama Distributor	Alamat	No Telp 1	No Telp 2	No Telp 3
dist01	Obat Saya	Jl.Gejayan No.5...	0274552211	08122718516	tidak ada
dist02	Mulya Jaya	Jl.Solo No.128 Y...	0271334455	08122334455	0811224466
dist03	Murni	Jl. Parangiritis N...	0274223388	08177886655	08523456789
dist04	Taufan Corp.	jalan kanigoro 2...	tidak ada	tidak ada	tidak ada

Gambar 4.9 form untuk mencari data distributor

DETAIL PEMBELIAN

F5-Cari Data Obat

—Data Detail Pembelian—

No Detail	Kode Obat	Nama Obat	Jenis Obat	Golongan	Satuan	Harga Beli	Harga Jual	Limit	Tgl Pembu...	Tgl kadalu...	Jumlah
2	obat 02	ibuprofen	psikotro...	1	botol	1500	1700	20	30-01-2...	31-01-2...	50
1	obat01	decolgen	obat beb...	2	tablet	1200	1300	10	21-08-2...	25-02-2...	100
3	obat03	amoxilin	owa	2	tablet	2000	2500	50	21-08-2...	10-08-2...	50
4	obat04	optum	narkotika	1	botol	1300	1400	20	12-08-2...	01-08-2...	50
11	obat05	paraceta...	obat beb...	1	tablet	200	2700	10	23-08-2...	31-08-2...	10

Kode Pembelian : Harga Beli :

No Detail : Harga Jual :

Kode Obat : Limit :

Nama Obat : Tgl Pembuatan :

Jenis Obat : Tgl Kadaluarsa :

Golongan : Jumlah :

Satuan :

TAMBAH
SIMPAN
HAPUS
BATAL

Gambar 4.10 form detail pembelian

Kode Obat	Nama Obat	Jenis Obat	Golongan	Satuan	Harga Obat	Stok	Limit Obat
obat01	obat pertama	obat pilek	psikotropika	tablet	1200	99	15
obat02	obat kedua	obat batuk	owa	botol	3200	146	35
obat03	obat ketiga	obat pusing	narkotika	tablet	2500	99	30
obat04	obat keempat	obat mata	bebas	botol	3000	200	20
obat05	obat kelima	obat kaki	owa	pil	1300	200	20
obat06	obat keenam	obat tangan	bebas terba...	tablet	1250	140	20
obat07	obat ketujuh	obat kepala	bebas	tablet	2250	100	20

Gambar 4.11 form untuk mencari data obat

Script berikut merupakan kelas `DataModelPembelian.java` dan kelas `Pembelian.java` yang digunakan untuk melakukan penambahan, pengubahan dan penghapusan data pembelian. Selain itu, terdapat pula script yang digunakan untuk melakukan penambahan, pengubahan, dan penghapusan data detail pembelian. Kelas yang digunakan untuk proses penambahan, pengubahan, dan penghapusan data detail pembelian adalah kelas `DataModelDetailPembelian.java` dan kelas `DetailPembelian.java`.

```

. . .
public boolean InsertRecPembelian(java.util.Vector
data) {
    int lastIdxRec = getRowCount();
    boolean kondisi = false;

    beli.setKode_dist((String) data.get(1));
    beli.setNo_faktur((String) data.get(2));
    beli.setTgl_faktur(Date.valueOf((String)
data.get(3)));
    beli.setNo_surat_pesan((String) data.get(4));

```

```

        beli.setTgl_surat_pesan(Date.valueOf((String)
data.get(5)));
        beli.setPotongan(Integer.valueOf((String)
data.get(6)));

        if (beli.insertPembelian()) {
            //triger ke table view bahwa ada data yang
diubah
            //update vector untuk data table
            fireTableRowsInserted(lastIdxRec,
lastIdxRec);
            Object[] r = {
                (String) data.get(0),
                (String) data.get(1),
                (String) data.get(2),
                (String) data.get(3),
                (String) data.get(4),
                (String) data.get(5),
                (String) data.get(6)
            };
            baris.addElement(r);
            kondisi = true;
        } else {
            kondisi = false;
        }

        return kondisi;
    }

    public boolean UpdateRecPembelian(int rowIndex,
java.util.Vector data) {
        int lastIdxRec = getRowCount();
        boolean kondisi = false;

        beli.setKode_pembelian(Integer.valueOf((String)
data.get(0)));
        beli.setKode_dist((String) data.get(1));
        beli.setNo_faktur((String) data.get(2));
        beli.setTgl_faktur(Date.valueOf((String)
data.get(3)));
        beli.setNo_surat_pesan((String) data.get(4));
        beli.setTgl_surat_pesan(Date.valueOf((String)
data.get(5)));
        beli.setPotongan(Integer.valueOf((String)
data.get(6)));

        if (beli.updatePembelian()) {
            //triger ke table view bahwa ada data yang
diubah
            //update vector untuk data table
            fireTableRowsUpdated(lastIdxRec,
lastIdxRec);
            setValueAt((String) data.get(0), rowIndex,
0);

```



```

        setValueAt((String) data.get(1), rowIndex,
1);
        setValueAt((String) data.get(2), rowIndex,
2);
        setValueAt((String) data.get(3), rowIndex,
3);
        setValueAt((String) data.get(4), rowIndex,
4);
        setValueAt((String) data.get(5), rowIndex,
5);
        setValueAt((String) data.get(6), rowIndex,
6);

        kondisi = true;
    } else {
        kondisi = false;
    }
    return kondisi;
}

public boolean DeleteRecPembelian(int rowIndex) {

    boolean kondisi = false;

//
dp.setKode_pembelian(Integer.valueOf((String)
getValueAt(rowIndex, 0)));
//
//      if (dp.cekStatus()) {

beli.setKode_pembelian(Integer.valueOf((String)
getValueAt(rowIndex, 0)));
    if (beli.deletePembelian()) {
        //triger ke table view bahwa ada data yang
diubah
        //update vector untuk data table
        fireTableRowsDeleted(rowIndex, rowIndex);
        baris.removeElementAt(rowIndex);
        //baris.addElement(r);
        kondisi = true;
    } else {
        kondisi = false;
    }
    return kondisi;
}
}
}

```

Listing 4.1 Kelas DataModelPembelian.java

```

. . .
public boolean insertPembelian(){
    boolean kondisi_insert=false;
    String sql = "{ call
spInsertPembelian(?,?,?,?,?,?)}";
    try {
        //lakukan precompile perintah sql insert

```

```

        java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

        //ganti setiap simbol ? dengan nilai yang
diharapkan
        pstmt.setString(1, this.kode_dist);
        pstmt.setString(2, this.no_faktur);
        pstmt.setDate(3, this.tgl_faktur);
        pstmt.setString(4, this.no_surat_pesan);
        pstmt.setDate(5, this.tgl_surat_pesan);
        pstmt.setInt(6, this.potongan);

        pstmt.executeUpdate();

        //kirim sql ke database server dan
jalankan
        pstmt.close();
        kondisi_insert=true;
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi_insert;
}

public boolean updatePembelian(){
    boolean kondisi_update=false;
    String sql = "{ call
spUpdatePembelian(?,?,?,?,?,?,?)}";
    try {
        //lakukan precompile perintah sql insert
        java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

        //ganti setiap simbol ? dengan nilai yang
diharapkan
        pstmt.setInt(1, this.kode_pembelian);
        pstmt.setString(2, this.kode_dist);
        pstmt.setString(3, this.no_faktur);
        pstmt.setDate(4, this.tgl_faktur);
        pstmt.setString(5, this.no_surat_pesan);
        pstmt.setDate(6, this.tgl_surat_pesan);
        pstmt.setInt(7, this.potongan);

        pstmt.executeUpdate();

        //kirim sql ke database server dan
jalankan
        pstmt.close();
        kondisi_update=true;
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi_update;
}

public boolean deletePembelian(){

```

```

        boolean kondisi_delete=false;
        String sql = "{ call spDeleteBeli(?,?)}";
        try {
            //lakukan precompile perintah sql insert
            java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

            //ganti setiap simbol ? dengan nilai yang
diharapkan
            pstmt.setInt(1, this.kode_pembelian);
            pstmt.registerOutParameter("stat",
java.sql.Types.INTEGER);
            pstmt.executeUpdate();
            int status = pstmt.getInt("stat");
            if(status==1) {
                kondisi_delete = true;
            }
            //kirim sql ke database server dan
jalankan
            pstmt.close();
        //        kondisi_delete=true;
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
        return kondisi_delete;
    }
}

```

Listing 4.2 Kelas Pembelian.java

```

...
public boolean
InsertRecDetailPembelian(java.util.Vector data) {
    int lastIdxRec = getRowCount();
    boolean kondisi = false;

    Obat obt = new Obat();

    dp.setKode_pembelian(Integer.valueOf((String)
data.get(0)));
    dp.setKode_obat((String) data.get(2));
    obt.setNama_obat((String) data.get(3));
    obt.setJenis_obat((String) data.get(4));
    obt.setGolongan((String) data.get(5));
    obt.setSatuan((String) data.get(6));
    dp.setHarga_beli(Integer.valueOf((String)
data.get(7)));
    obt.setHarga_obat(Double.valueOf((String)
data.get(8)));
    obt.setLimit_obat(Integer.valueOf((String)
data.get(9)));
    dp.setTgl_pembuatan(Date.valueOf((String)
data.get(10)));
    dp.setTgl_kadaluarsa(Date.valueOf((String)
data.get(11)));
    dp.setJumlah(Integer.valueOf((String)
data.get(12)));
}

```

```

        if (dp.insertDetailPembelian(obt)) {
            //triger ke table view bahwa ada data yang
diubah
            //update vector untuk data table
            fireTableRowsInserted(lastIdxRec,
lastIdxRec);
            Object[] r = {
                (String) data.get(1),
                (String) data.get(2),
                (String) data.get(3),
                (String) data.get(4),
                (String) data.get(5),
                (String) data.get(6),
                (String) data.get(7),
                (String) data.get(8),
                (String) data.get(9),
                (String) data.get(10),
                (String) data.get(11),
                (String) data.get(12)
            };
            baris.addElement(r);
            kondisi = true;
        } else {
            kondisi = false;
        }

        return kondisi;
    }

    public boolean UpdateRecDetailPembelian(int
rowIndex, java.util.Vector data) {
        int lastIdxRec = getRowCount();
        boolean kondisi = false;

        Obat obt = new Obat();

        dp.setKode_pembelian(Integer.valueOf((String)
data.get(0)));

        dp.setNo_detail_pembelian(Integer.valueOf((String)
data.get(1)));
        dp.setKode_obat((String) data.get(2));
        obt.setNama_obat((String) data.get(3));
        obt.setJenis_obat((String) data.get(4));
        obt.setGolongan((String) data.get(5));
        obt.setSatuan((String) data.get(6));
        dp.setHarga_beli(Integer.valueOf((String)
data.get(7)));
        obt.setHarga_obat(Double.valueOf((String)
data.get(8)));
        obt.setLimit_obat(Integer.valueOf((String)
data.get(9)));
        dp.setTgl_pembuatan(Date.valueOf((String)
data.get(10)));

```

```

        dp.setTgl_kadaluarsa(Date.valueOf((String)
data.get(11)));
        dp.setJumlah(Integer.valueOf((String)
data.get(12)));

        if (dp.updateDetailPembelian(obt)) {
            //triger ke table view bahwa ada data yang
diubah
            //update vector untuk data table
            fireTableRowsUpdated(lastIdxRec,
lastIdxRec);
            //
            setValueAt((String) data.get(0),
rowIndex, 0);
            setValueAt((String) data.get(1), rowIndex,
0);
            setValueAt((String) data.get(2), rowIndex,
1);
            setValueAt((String) data.get(3), rowIndex,
2);
            setValueAt((String) data.get(4), rowIndex,
3);
            setValueAt((String) data.get(5), rowIndex,
4);
            setValueAt((String) data.get(6), rowIndex,
5);
            setValueAt((String) data.get(7), rowIndex,
6);
            setValueAt((String) data.get(8), rowIndex,
7);
            setValueAt((String) data.get(9), rowIndex,
8);
            setValueAt((String) data.get(10),
rowIndex, 9);
            setValueAt((String) data.get(11),
rowIndex, 10);
            setValueAt((String) data.get(12),
rowIndex, 11);

            kondisi = true;
        } else {
            kondisi = false;
        }
        return kondisi;
    }

    public boolean DeleteRecPembelian(int rowIndex) {
        boolean kondisi = false;

        dp.setNo_detail_pembelian(Integer.valueOf((String)
getValueAt(rowIndex, 0)));
        if (dp.deleteDetailPembelian()) {
            //triger ke table view bahwa ada data
yang diubah
            //update vector untuk data table

```

```

        fireTableRowsDeleted(rowIndex,
rowIndex);
        baris.removeElementAt(rowIndex);

        kondisi = true;
    } else {
        kondisi = false;
    }
    return kondisi;
}
}

```

Listing 4.3 Kelas DataModelDetailPembelian.java

```

...
public boolean insertDetailPembelian(Obat obt){
    boolean kondisi_insert=false;
    String sql = "{ call
spInsertDetailBeli(?,?,?,?,?,?,?,?,?,?,?)}";
    try {
        //lakukan precompile perintah sql insert
        java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

        //ganti setiap simbol ? dengan nilai yang
diharapkan
        pstmt.setInt(1, this.kode_pembelian);
        pstmt.setString(2, this.kode_obat);
        pstmt.setString(3, obt.getNama_obat());
        pstmt.setString(4, obt.getJenis_obat());
        pstmt.setString(5, obt.getGolongan());
        pstmt.setString(6, obt.getSatuan());
        pstmt.setDouble(7, obt.getHarga_obat());
        pstmt.setInt(8, obt.getLimit_obat());
        pstmt.setDate(9, this.tgl_pembuatan);
        pstmt.setDate(10, this.tgl_kadaluarsa);
        pstmt.setDouble(11, this.harga_beli);
        pstmt.setInt(12, this.jumlah);

        pstmt.executeUpdate();

        //kirim sql ke database server dan
jalankan
        pstmt.close();
        kondisi_insert=true;
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi_insert;
}

public boolean updateDetailPembelian(Obat obt){
    boolean kondisi_update=false;
    String sql = "{ call
spUpdateDetailPembelian(?,?,?,?,?,?,?,?,?,?,?)}";
    try {
        //lakukan precompile perintah sql insert

```

```

        java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

        //ganti setiap simbol ? dengan nilai yang
diharapkan
        pstmt.setInt(1, this.kode_pembelian);
        pstmt.setInt(2, this.no_detail_pembelian);
        pstmt.setString(3, this.kode_obat);
        pstmt.setString(4, obt.getNama_obat());
        pstmt.setString(5, obt.getJenis_obat());
        pstmt.setString(6, obt.getGolongan());
        pstmt.setString(7, obt.getSatuan());
        pstmt.setDouble(8, obt.getHarga_obat());
        pstmt.setInt(9, obt.getLimit_obat());
        pstmt.setDate(10, this.tgl_pembuatan);
        pstmt.setDate(11, this.tgl_kadaluarsa);
        pstmt.setDouble(12, this.harga_beli);
        pstmt.setInt(13, this.jumlah);

        pstmt.executeUpdate();

        //kirim sql ke database server dan
jalankan
        pstmt.close();
        kondisi_update=true;
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi_update;
}

public boolean deleteDetailPembelian(){
    boolean kondisi_delete=false;
    String sql = "{ call spDeleteDetailBeli(?)}";
    try {
        //lakukan precompile perintah sql insert
        java.sql.CallableStatement pstmt=
conn.prepareStatement(sql);

        //ganti setiap simbol ? dengan nilai yang
diharapkan
        pstmt.setInt(1, this.no_detail_pembelian);

        pstmt.executeUpdate();

        //kirim sql ke database server dan
jalankan
        pstmt.close();
        kondisi_delete=true;
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi_delete;
}
}

```

Listing 4.4 Kelas DetailPembelian.java

Pada *Listing 4.1 Kelas DataModelPembelian.java* dan *Listing 4.2 Kelas Pembelian.java* terdapat method untuk melakukan :

1. Mengisi Data Pembelian

Pada kelas *DataModelPembelian.java* membuat obyek untuk menyimpan data pembelian yang akan diinputkan. Obyek tersebut adalah beli, yang dibentuk dari kelas pembelian. Selanjutnya dilakukan pemanggilan method *insertPembelian* pada kelas pembelian. Pada method ini dilakukan pemanggilan Store Procedure yang bernama *spInsertPembelian* yang ditunjukkan dengan listing berikut :

```
String sql = "{ call  
spInsertPembelian(?,?,?,?,?,?)}";
```

Kemudian simbol ? dalam parameter *spInsertPembelian* diganti dengan nilai dari atribut pada kelas ini. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setString(1, this.kode_dist);  
pstmt.setString(2, this.no_faktur);  
pstmt.setDate(3, this.tgl_faktur);  
pstmt.setString(4, this.no_surat_pesan);  
pstmt.setDate(5, this.tgl_surat_pesan);  
pstmt.setInt(6, this.potongan);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

2. Mengubah Data Pembelian

Pada proses mengubah data pembelian ini dilakukan pembentukan obyek dari kelas pembelian, kemudian obyek

tersebut akan memanggil method `updatePembelian`. Pada method `updatePembelian` ini dilakukan pemanggilan Store Procedure yang bernama `spUpdatePembelian` yang ditunjukkan dengan listing berikut :

```
String sql = "{ call  
spUpdatePembelian(?,?,?,?,?,?,?)}";
```

Kemudian simbol ? dalam parameter `spUpdatePembelian` diganti dengan nilai dari atribut pada kelas ini. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setInt(1, this.kode_pembelian);  
pstmt.setString(2, this.kode_dist);  
pstmt.setString(3, this.no_faktur);  
pstmt.setDate(4, this.tgl_faktur);  
pstmt.setString(5, this.no_surat_pesan);  
pstmt.setDate(6, this.tgl_surat_pesan);  
pstmt.setInt(7, this.potongan);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

3. Menghapus Data Pembelian

Pada proses menghapus data pembelian ini dilakukan pemanggilan Store Procedure yang bernama `spDeleteBeli` yang ditunjukkan dengan listing berikut:

```
String sql = "{ call spDeleteBeli(?)}";
```

Kemudian simbol ? dalam parameter `spDeleteBeli` diganti dengan nilai dari atribut pada kelas ini. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setInt(1, this.kode_pembelian);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

Dan pada *Listing 4.3 Kelas DataModelDetailPembelian.java* dan *Listing 4.4 Kelas DetailPembelian.java* terdapat method untuk melakukan:

1. Mengisi Data Detail Pembelian

Pada kelas *DataModelDetailPembelian.java* membuat obyek untuk menyimpan data obat dan data detail pembelian yang akan diinputkan. Obyek tersebut adalah *obt* dan *dp*, yang dibentuk dari kelas *obat* dan kelas *detail pembelian*. Selanjutnya dilakukan pemanggilan method *insertDetailPembelian* dengan parameter *obt* yang terdapat pada kelas *detail pembelian*. Pada method ini dilakukan pemanggilan Store Procedure yang bernama *spInsertDetailBeli* yang ditunjukkan dengan listing berikut :

```
String sql = "{ call  
spInsertDetailBeli(?,?,?,?,?,?,?,?,?,?,?)}";
```

Kemudian simbol ? dalam parameter *spInsertDetailBeli* diganti dengan nilai dari atribut pada kelas ini dan atribut dari kelas *obat*. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setInt(1, this.kode_pembelian);  
pstmt.setString(2, this.kode_obat);  
pstmt.setString(3, obt.getNama_obat());  
pstmt.setString(4, obt.getJenis_obat());  
pstmt.setString(5, obt.getGolongan());  
pstmt.setString(6, obt.getSatuan());
```

```
pstmt.setDouble(7, obt.getHarga_obat());
pstmt.setInt(8, obt.getLimit_obat());
pstmt.setDate(9, this.tgl_pembuatan);
pstmt.setDate(10, this.tgl_kadaluarsa);
pstmt.setDouble(11, this.harga_beli);
pstmt.setInt(12, this.jumlah);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

2. Mengubah Data Detail Pembelian

Pada proses mengubah data detail pembelian ini dilakukan pembentukan obyek dari kelas detail pembelian dan kelas obat, kemudian obyek-obyek tersebut akan memanggil method updateDetailPembelian. Pada method updateDetailPembelian ini dilakukan pemanggilan Store Procedure yang bernama spUpdateDetailPembelian yang ditunjukkan dengan listing berikut :

```
String sql = "{ call
spUpdateDetailPembelian(?,?,?,?,?,?,?,?,?,?,?,?)
";
```

Kemudian simbol ? dalam parameter

spUpdateDetailPembelian diganti dengan nilai dari atribut pada kelas ini dan atribut dari kelas obat. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setInt(1, this.kode_pembelian);
pstmt.setInt(2, this.no_detail_pembelian);
pstmt.setString(3, this.kode_obat);
pstmt.setString(4, obt.getNama_obat());
pstmt.setString(5, obt.getJenis_obat());
pstmt.setString(6, obt.getGolongan());
pstmt.setString(7, obt.getSatuan());
pstmt.setDouble(8, obt.getHarga_obat());
pstmt.setInt(9, obt.getLimit_obat());
```

```
pstmt.setDate(10, this.tgl_pembuatan);
pstmt.setDate(11, this.tgl_kadaluarsa);
pstmt.setDouble(12, this.harga_beli);
pstmt.setInt(13, this.jumlah);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

3. Menghapus Data Detail Pembelian

Pada proses menghapus data detail pembelian ini dilakukan pemanggilan Store Procedure yang bernama spDeleteDetailBeli yang ditunjukkan dengan listing berikut :

```
String sql = "{ call spDeleteDetailBeli(?)}";
```

Kemudian simbol ? dalam parameter spDeleteDetailBeli diganti dengan nilai dari atribut pada kelas ini. Proses tersebut dapat ditunjukkan dengan sintaks sebagai berikut :

```
pstmt.setInt(1, this.no_detail_pembelian);
```

Untuk menjalankan Store Procedure di atas digunakan perintah berikut :

```
pstmt.executeUpdate();
```

Store procedure spInsertDetailBeli akan mempengaruhi jumlah stok pada tabel obat. Listing 4.5 merupakan store procedure spInsertDetailBeli.

```
DELIMITER $$

DROP PROCEDURE IF EXISTS `apotek`.`spInsertDetailBeli`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE
`spInsertDetailBeli`(pKodeBeli int, pKodeObat varchar(50),
pNamaObat varchar(100), pJenis varchar(25),
                        pGolongan varchar(20), pSatuan
VArchar(10), pHrgObat double, pLimit Int,
```

```

                                pTglPembuatan date, pTglLED
date, pHrgBeli double, pJumlah int)
BEGIN
    start transaction;

    -- pengecekan obat, 1 pembelian ga bole ada obat yang
sama
        IF NOT EXISTS(select * from detailpembelian
where kode_obat like pKodeObat
                        AND kode_pembelian=pKodeBeli) THEN

        -- setelah itu insert ke tabel obat
        if exists (select * from obat
where kode_obat=pKodeObat) then
            UPDATE obat set stok
            =if(isnull(stok),0,stok)+pJumlah where
kode_obat = pKodeObat;
        ELSE
            insert into obat
            values(pKodeObat, pNamaObat, pJenis,
pGolongan, pSatuan, pHrgObat, pJumlah, pLimit);
        END IF;

        -- insert ke tabel detailpembelian
        insert into detailpembelian
(kode_pembelian, kode_obat, tgl_pembuatan, tgl_kadaluarsa,
harga_beli, jumlah)
        values(pKodeBeli, pKodeObat,
pTglPembuatan, pTglLED, pHrgBeli, pJumlah);

        COMMIT;
    ELSE
        ROLLBACK;
    END IF;

END$$
DELIMITER ;

```

Listing 4.5 Store Procedure spInsertDetailBeli

Pada listing 4.5 di atas dilakukan pengecekan apakah ada data obat sudah ada atau masih kosong pada satu pembelian. Jika sudah ada maka store procedure ini tidak jadi dijalankan dan akan ROLLBACK. Namun apabila data obat pada proses pembelian tersebut belum ada maka proses akan berlanjut dengan mengisi data obat ke tabel obat. Jika obat yang dimasukkan sudah ada di tabel obat maka obat yang dibeli tadi hanya akan menambah jumlah stok, tapi jika obat tersebut belum ada di tabel obat maka

data obat yang baru dibeli tadi dimasukkan ke dalam tabel obat. Selanjutnya proses berlanjut ke tabel detail pembelian. Pada tabel ini diisi kode_pembelian, kode_obat, tgl_pembuatan, tgl_kadaluarsa, harga_beli, dan jumlah. Apabila semua proses di atas dapat dijalankan maka akan COMMIT. Jika salah satu proses tersebut di atas tidak dapat dijalankan maka akan ROLLBACK.

4.4. Implementasi Form Penjualan

Apabila pegawai ingin melakukan proses penjualan. Maka pegawai memilih menu penjualan pada form utama. Kemudian akan muncul form penjualan. Pegawai dapat melakukan dua jenis penjualan, yaitu penjualan counter dan penjualan resep. Penjualan counter adalah penjualan obat yang tidak perlu menggunakan resep. Sedangkan penjualan resep adalah penjualan obat yang harus menggunakan resep. Gambar 4.12 merupakan tampilan form penjualan counter dan gambar 4.13 merupakan tampilan form penjualan resep.

Untuk memilih penjualan counter, pegawai menekan *radiobutton* penjualan counter, kemudian mengisi data obat yang dibeli oleh pasien atau pembeli pada tabel yang tersedia. Untuk mengisi data obat pada tabel, pegawai hanya perlu menekan tombol F5 sehingga muncul form untuk mencari data obat. Selanjutnya jika ada data embalase atau data toeslag, pegawai hanya perlu menekan tombol F7 atau F8. Tombol F7 akan menampilkan form untuk mencari data embalase dan tombol F8 akan

menampilkan form untuk mencari data toeslag. Setelah itu untuk menghitung harga dari obat yang dibeli tadi, pegawai menekan tombol enter pada tabel, selanjutnya total harga yang dibayar akan muncul pada *textfield* total harga. Kemudian pegawai mengisi harga yang dibayar oleh pasien atau pembeli, setelah itu pegawai menekan tombol enter pada *textfield* dibayar, sehingga akan muncul sisa yang harus dikembalikan. Setelah itu pegawai menekan tombol SIMPAN untuk menyimpan data penjualan obat ke *database*. Gambar 4.14 merupakan tampilan form untuk mencari data obat, gambar 4.15 merupakan tampilan form untuk mencari data embalase dan gambar 4.16 merupakan tampilan form untuk mencari data toeslag.

Untuk memilih penjualan resep, pegawai menekan *radiobutton* penjualan resep, kemudian pegawai mengisi data pasien dan data dokter pada *textfield* yang tersedia, selanjutnya untuk proses pengisian obat, penghitungan harga, dan penyimpanan data penjualan ke *database* sama dengan penjualan counter.

Proses penyimpanan data penjualan ke *database* hampir sama dengan proses pembelian obat pada point 4.3. Implementasi Form Pembelian. Perbedaannya disini adalah store procedure yang digunakan dan proses penjualan ini akan mengurangi jumlah stok pada tabel obat. Listing 4.6 merupakan store procedure *spInsertJual*.

```
DELIMITER $$

DROP PROCEDURE IF EXISTS `apotek`.`spInsertJual`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE
`spInsertJual`(pKodeJual int, pKodeObat varchar(50),
pJumlahObat INT,OUT cek varchar(5))
```

```

BEGIN

    declare vHargaObat double;
    declare vStokObat INT;
    -- ambil harga n stok berdasarkan kode_obat
    select harga_obat, stok into vHargaObat, vStokObat
    from obat
    where kode_obat like pKodeObat;
    if(pJumlahObat <= vStokObat) then
    -- insert ke tabel penjualan
    if not exists(select * from penjualan where kode_penjualan
    like pKodeJual) then
    insert into penjualan values(pKodeJual, NOW());
    end if;

    -- ngambil kode penjualan
    insert into
    detailpenjualan(kode_penjualan,kode_obat,harga_jual,jumlah)
    values(pKodeJual,pKodeObat,vHargaObat,pJumlahObat);
    update obat SET stok = vStokObat - pJumlahObat
    where kode_obat like pKodeObat;

    set cek = "benar";
    else
    set cek = "salah";
    end if;
    END$$

DELIMITER ;

```

Listing 4.6 Store Procedure spInsertJual

Pada *Listing 4.6 Store Procedure spInsertJual* dilakukan pengecekan apakah stok obat mencukupi untuk melakukan proses penjualan. Apabila mencukupi maka akan dilakukan *insert* ke tabel penjualan, kemudian dilanjutkan ke tabel detail penjualan dengan mengisi kode penjualan, kode obat, harga jual, dan jumlah. Selanjutnya akan dilakukan pengurangan stok obat pada tabel obat.

PENJUALAN

F5-Cari Data Obat | F7-Cari Data Embalase | F8-Cari Data Toeslag

☒ Penjualan Counter ☐ Penjualan Resep

Data Pasien dan Data Dokter

Nama Pasien : Tanggal Resep :

Alamat Pasien : Nama Dokter :

Keluhan :

Obat Yang Akan Di Beli

Kode	Nama Obat/Embalase/...	Jenis	Jumlah	Harga	Total Harga
obat01	obat pertama	Obat	1	1200	1200
obat02	obat kedua	Obat	1	3200	3200
embas01	plastik	Embalase	1	250	250
toes01	racikan 3 obat	Toeslag	1	3000	3000

Total Harga : Rp.

Dibayar : Rp.

Sisa : Rp.

APOTEK JOINT FARMA

Gambar 4.12 Form Penjualan Counter

PENJUALAN

F5-Cari Data Obat | F7-Cari Data Embalase | F8-Cari Data Toeslag

☐ Penjualan Counter ☒ Penjualan Resep

Data Pasien dan Data Dokter

Nama Pasien : Tanggal Resep :

Alamat Pasien : Nama Dokter :

Keluhan :

Obat Yang Akan Di Beli

Kode	Nama Obat/Embalase/...	Jenis	Jumlah	Harga	Total Harga
obat01	obat pertama	Obat	1	1200	1200
obat02	obat kedua	Obat	1	3200	3200
embas01	plastik	Embalase	1	250	250
toes01	racikan 3 obat	Toeslag	1	3000	3000

Total Harga : Rp.

Dibayar : Rp.

Sisa : Rp.

APOTEK JOINT FARMA

Gambar 4.13 Form Penjualan Resep

PENCARIAN DATA OBAT

KATEGORI : kode_obat ▼

KATA KUNCI :

Data Obat

Kode Obat	Nama Obat	Jenis Obat	Golongan	Satuan	Harga Obat	Stok	Limit Obat
obat01	obat pertama	obat pilek	psikotropika	tablet	1200	99	15
obat02	obat kedua	obat batuk	owa	botol	3200	146	35
obat03	obat ketiga	obat pusing	narkotika	tablet	2500	99	30
obat04	obat keempat	obat mata	bebas	botol	3000	200	20
obat05	obat kelima	obat kaki	owa	pil	1300	200	20
obat06	obat keenam	obat tangan	bebas terba...	tablet	1250	140	20
obat07	obat ketujuh	obat kepala	bebas	tablet	2250	100	20

Gambar 4.14 Form untuk mencari data obat

PENCARIAN DATA EMBALASE

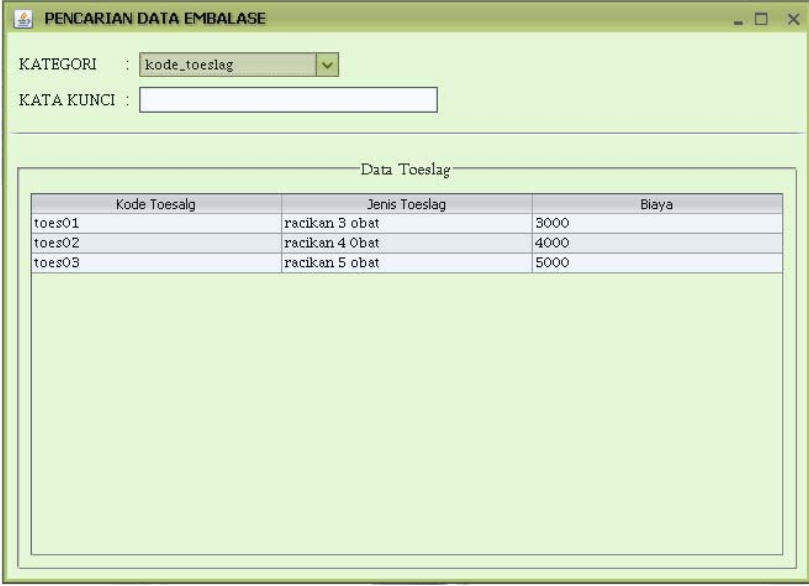
KATEGORI : kode_embalase ▼

KATA KUNCI :

Data Embalase

Kode Embalase	Jenis Embalase	Harga Embalase
embas01	plastik	250
embas02	kertas	100
embas04	kantong	250

Gambar 4.15 Form untuk mencari data embalase

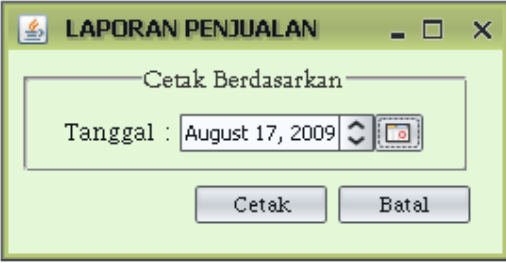


Kode Toesalg	Jenis Toeslag	Biaya
toes01	racikan 3 obat	3000
toes02	racikan 4 Obat	4000
toes03	racikan 5 obat	5000

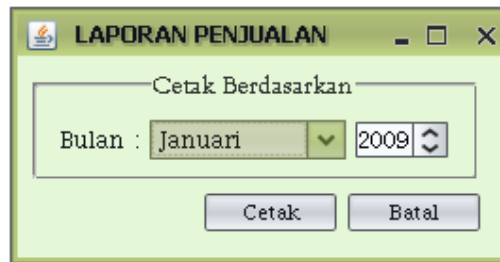
Gambar 4.16 Form untuk mencari data toeslag

4.5. Implementasi Form Cetak Laporan Penjualan

Untuk mencetak laporan penjualan, admin memilih menu cetak kemudian memilih submenu laporan penjualan. Laporan penjualan ini dapat di cetak per hari atau per bulan. Gambar 4.17 merupakan tampilan form cetak laporan penjualan harian dan gambar 4.18 merupakan tampilan form cetak laporan penjualan bulanan.

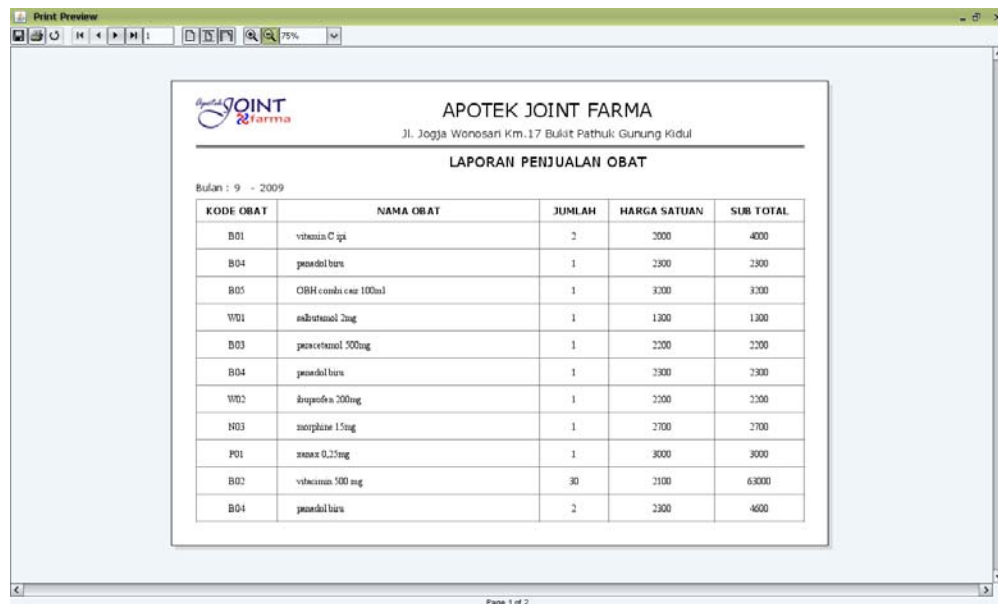


Gambar 4.17 Form cetak laporan penjualan harian



Gambar 4.18 Form cetak laporan penjualan bulanan

Setelah memilih ingin mencetak laporan penjualan berdasarkan hari atau bulan, maka admin menekan tombol Cetak. Kemudian akan muncul *preview* laporan penjualan. Gambar 4.19 merupakan *preview* laporan penjualan berdasarkan bulan.



KODE OBAT	NAMA OBAT	JUMLAH	HARGA SATUAN	SUB TOTAL
B01	vitamin C 500	2	2000	4000
B04	parasetamol	1	2300	2300
B05	OBH combi 100ml	1	3200	3200
W01	salbutamol 2mg	1	1300	1300
B03	parasetamol 500mg	1	2200	2200
B04	parasetamol	1	2300	2300
W02	ibuprofen 200mg	1	2200	2200
N03	aspirin 150mg	1	2700	2700
P01	asam 0.25mg	1	3000	3000
B02	vitamin 500 mg	30	2100	63000
B04	parasetamol	2	2300	4600

Gambar 4.19 preview laporan penjualan berdasarkan bulan

Script berikut merupakan kelas MyJasperViewer.java yang digunakan untuk menampilkan *preview* laporan.

```

. . .
public static boolean showPreview(JFrame parent, JasperPrint
jasperPrint){
    Frame fparent =
JOptionPane.getFrameForComponent((Component)parent);

    try{
        jview = new MyJasperViewer(fparent, jasperPrint);
        jview.setVisible(true);
        return true;
    }catch(JRException jrex){
        jrex.printStackTrace();
        return false;
    }
}

public void init(){
    ...
    java.awt.Dimension screenSize =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
    setSize(new java.awt.Dimension(800, 600));
    setLocation((screenSize.width-
800)/2, (screenSize.height-600)/2);
}
. . .

```

Listing 4.7 Kelas MyJasperViewe.java

Pada *Listing 4.7 Kelas MyJasperViewe.java* terdapat method *showPreview* dengan 2 parameter. Pada method ini terdapat inialisasi objek dari kelas *MyJasperViewer*. Kemudian objek ini di-*setVisible = true* agar dapat muncul di desktop. Proses ini ditunjukkan dengan listing sebagai berikut :

```

try{
    jview = new MyJasperViewer(fparent, jasperPrint);
    jview.setVisible(true);
    return true;
}

```

Setelah ditampilkan, form akan diatur posisinya dengan menggunakan *Dimension* di mana diatur ukurannya dan lokasi muncul di desktop yang ditunjukkan dengan listing sebagai berikut :

```

java.awt.Dimension screenSize =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
    setSize(new java.awt.Dimension(800, 600));
    setLocation((screenSize.width-
800)/2, (screenSize.height-600)/2);

```

4.6. Implementasi Form Warning Limit

Proses pengingat limit ini merupakan proses yang berjalan otomatis pada saat sistem dijalankan. Proses pengingat limit ini akan berjalan apabila ada stok obat yang jumlahnya sama dengan limit yang telah ditentukan sebelumnya. Gambar 4.20 merupakan tampilan form warning limit.

Kode Obat	Nama Obat	Jenis Obat	Golongan	Satuan	Harga Obat	Stok	Limit
amat03	amoxilin	owa	2	tablet	2500	50	50
amat04	opium	narkotika	1	botol	1400	20	20

Gambar 4.20 form warning limit

Script berikut merupakan kelas `DataModelWarningLimit.java` dan `Obat.java` yang digunakan untuk menampilkan data obat yang harus dipesan ke distributor.

```

. . .
public void getRecDBSetupWarning(String kategori, String
kataKunci) {
    try {
        java.sql.ResultSet resultSet =
obat.selectWarningLimit(kategori, kataKunci);

        baris.removeAllElements();
        while (resultSet.next()) {
            Object[] r = {resultSet.getString(1),
                resultSet.getString(2),
                resultSet.getString(3),
                resultSet.getString(4),
                resultSet.getString(5),
                resultSet.getString(6),
                resultSet.getString(7),
                resultSet.getString(8)
            };
            baris.addElement(r);
            r = null;
        }
    } catch (java.sql.SQLException e) {
        System.out.println("Error : " + e);
    }
}

```

```

    }
}

public boolean cekTabel() {
    boolean kondisi = false;

    try {
        java.sql.ResultSet resultSet =
obat.selectWarningLimit(null, null);
        if (resultSet.next()) {
            kondisi = true;
        }
    } catch (SQLException ex) {
//
Logger.getLogger(DataModelWarningED.class.getName()).log(Leve
l.SEVERE, null, ex);
        ex.printStackTrace();
    }
    return kondisi;
}
}

```

Listing 4.8 Kelas DataModelWarningLimit.java

```

. . .
public ResultSet selectWarningLimit(String kategori, String
kataKunci) {
    ResultSet rs = null;
    String sql;

    if (kategori == null) {
        sql = "select * from obat where stok <=
limit_obat";
    } else {
        sql = "select * from obat where " + kategori + "
like '%" + kataKunci + "%'";
    }

    try {
        java.sql.CallableStatement pstmt =
conn.prepareCall(sql);
        rs = pstmt.executeQuery();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    sql = null;
    return rs;
}
}

```

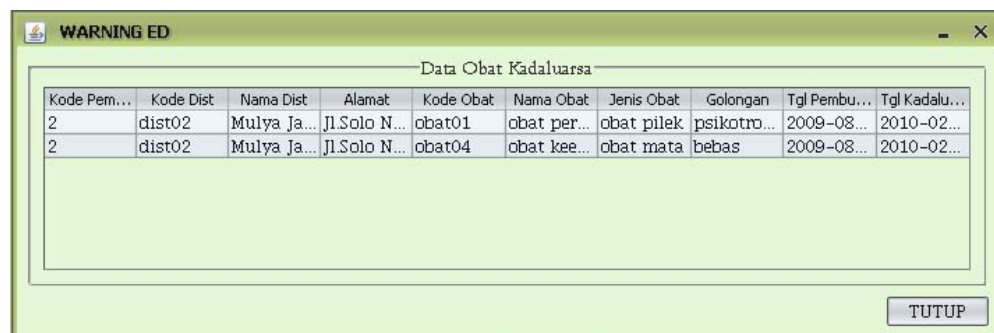
Listing 4.9 Kelas Obat.java

Pada *Listing 4.8 Kelas DataModelWarningLimit.java* dan *Listing 4.9 Kelas Obat.java* dibentuk obyek obat dari kelas obat, kemudian obyek ini memanggil method

selectWarningLimit dengan 2 parameter yaitu kategori dan katakunci. Pada method selectWarningLimit dibuat statement untuk memanggil query sql. Selanjutnya pada kelas DataModelWarningLimit.java dilakukan pengecekan tabel. Apakah tabel tersebut memiliki data untuk ditampilkan atau tidak. Apabila ada data yang akan ditampilkan maka pada sistem akan muncul form warning limit seperti pada gambar 4.20.

4.7. Implementasi Form Warning ED

Proses pengingat obat kadaluarsa ini merupakan proses yang berjalan otomatis pada saat sistem dijalankan. Proses pengingat kadaluarsa ini akan berjalan apabila ada obat yang kurang dari 6 bulan tanggal kadalursanya sama dengan tanggal pada saat sistem dijalankan. Gambar 4.21 merupakan tampilan form warning ED.



Gambar 4.21 form warning ed

Script berikut merupakan kelas DataModelWarningED.java dan Obat.java yang digunakan untuk menampilkan data obat yang hampir kadaluarsa.

```

. . .
public void getRecDBSetupWarning(String kategori, String
kataKunci) {
    try {

```



```

        java.sql.ResultSet resultSet =
obat.selectWarningED(kategori, kataKunci);

        baris.removeAllElements();
        while (resultSet.next()) {
            Object[] r = {resultSet.getString(1),
                resultSet.getString(2),
                resultSet.getString(3),
                resultSet.getString(4),
                resultSet.getString(5),
                resultSet.getString(6),
                resultSet.getString(7),
                resultSet.getString(8),
                resultSet.getString(9),
                resultSet.getString(10)
            };
            baris.addElement(r);
            r = null;
        }
    } catch (java.sql.SQLException e) {
        System.out.println("Error : " + e);
    }
}

public boolean cekTabel() {
    boolean kondisi = false;

    try {
        java.sql.ResultSet resultSet =
obat.selectWarningED(null, null);
        if (resultSet.next()) {
            kondisi = true;
        }
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    return kondisi;
}

public boolean UpdateRecStatus(int rowIndex,
java.util.Vector data) {
    int lastIdxRec = getRowCount();
    boolean kondisi = true;

    for (int i = 0; i < lastIdxRec; i++) {
        dp.setKode_pembelian(Integer.valueOf((String)
this.getValueAt(i, 0)));
        obat.setKode_obat((String) this.getValueAt(i,
4));

        if (!obat.updateStatus(dp)) {
            kondisi = false;
        }
    }
    return kondisi;}

```

Listing 4.9 Kelas DataModelWarningED.java

```

. . .
public ResultSet selectWarningED(String kategori, String
kataKunci) {
    ResultSet rs = null;
    String sql;

    if (kategori == null) {
        sql = "select kode_pembelian,kode_dist,nama_dist,
alamat," +
            "kode_obat,nama_obat, jenis_obat,
golongan,tgl_pembuatan,tgl_kadaluarsa " +
            "from obat join detailpembelian
using(kode_obat) " +
            "join pembelian using(kode_pembelian)
join distributor using(kode_dist) " +
            "where (date_sub(tgl_kadaluarsa, interval
6 month) like date(now())) " +
            "and status_pengecekan like 'Belum di
Cek'";
    } else {
        sql = "select kode_pembelian,kode_dist,nama_dist,
alamat," +
            "nama_obat, jenis_obat,
golongan,tgl_pembuatan,tgl_kadaluarsa " +
            "from obat join detailpembelian
using(kode_obat) " +
            "join pembelian using(kode_pembelian) " +
            "join distributor using(kode_dist) where
" + kategori + " like '%" + kataKunci + "%'";
    }

    try {
        java.sql.CallableStatement pstmt =
conn.prepareCall(sql);
        rs = pstmt.executeQuery();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
    sql = null;
    return rs;
}
. . .

```

Listing 4.10 Kelas Obat.java

Pada *Listing 4.9 Kelas DataModelWarningED.java* dan *Listing 4.10 Kelas Obat.java* dibentuk obyek obat dari kelas obat, kemudian obyek ini memanggil method `selectWarningED` dengan 2 parameter yaitu kategori dan katakunci. Pada method `selectWarningED` dibuat statement untuk memanggil query sql. Selanjutnya pada kelas `DataModelWarningED.java` dilakukan update status

pada tabel detail pembelian, selain dilakukan pengecekan tabel. Apakah tabel tersebut memiliki data untuk ditampilkan atau tidak. Apabila ada data yang akan ditampilkan maka pada sistem akan muncul form warning ed seperti pada gambar 4.19. Untuk mengatasi agar form warning ed ini tidak muncul setiap sistem dijalankan maka selain dilakukan pengecekan tabel juga dilakukan perubahan status pengecekan pada tabel detail pembelian sehingga form warning ed tidak muncul setiap kali program dijalankan.

BAB V

ANALISIS HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai analisa hasil dari implementasi Sistem Informasi Pengelolaan Stok Obat Pada Apotek Joint Farma. Analisa tersebut meliputi:

5.1. Pengetesan Program dari *Programmer*

Sebelum program diterapkan, maka dilakukan uji coba dengan memasukkan data-data sementara dan memprosesnya. Pengetesan dilakukan secara menyeluruh dengan mengetes semua menu yang tersedia pada sistem informasi pengelolaan stok obat ini. Hal yang diutamakan dalam pengetesan ini adalah bagaimana stok obat dapat bertambah dan berkurang secara otomatis melalui proses penjualan dan pembelian obat, serta jalannya pengingat otomatis yang berfungsi untuk mengingatkan stok limit obat dan obat kadaluarsa.

Tujuan utama dari pengetesan program ini adalah untuk memastikan bahwa program bebas dari kesalahan-kesalahan sebelum program benar-benar diterapkan di Apotek Joint Farma.

5.2. Pengetesan Program dari *User*

Pada pengetesan ini penilaian sistem dilihat dari segi tampilan, jalannya sistem, dan penanganan *error*. Pada pengetesan ini dibutuhkan ketelitian dan kesabaran dari *user* dalam mengisi data obat karena data obat yang dimasukkan tidak sedikit sekitar 500 hingga 1000 obat. Hasil dari pengetesan ini adalah sistem informasi pengelolaan stok obat dapat berjalan dengan baik.

5.3. Kelebihan Sistem

Selain kelemahan tersebut di atas, sistem informasi pengelolaan stok obat ini juga memiliki kelebihan antara lain:

1. Sistem ini mampu meningkatkan kinerja dari pengolahan data obat baik obat masuk maupun obat keluar secara efektif dan efisien.
2. Sistem ini dilengkapi dengan fasilitas pengingat obat kadaluarsa yang secara otomatis berjalan pada saat sistem dijalankan. Fasilitas ini berguna untuk pengguna sistem agar tidak mengecek tanggal kadaluarsa obat-obatan secara manual satu per satu, kemudian melakukan retur ke distributor.
3. Sistem ini juga dilengkapi dengan fasilitas pengingat limit obat yang secara otomatis berjalan pada saat sistem dijalankan, sehingga mengurangi resiko stok obat kosong.

5.4. Kelemahan Sistem

Pada saat pengujian, terdapat kelemahan sistem yaitu sistem belum dilengkapi dengan fasilitas *backup database* dalam kurun waktu tertentu, sehingga *backup* datanya hanya berupa rekap laporan yang dapat dicetak per hari atau per bulan. Selain itu pada sistem ini, penjualan dilakukan tidak berdasarkan tanggal kadaluarsa yang paling dekat, sehingga masih diperlukan pengecekan pada saat menaruh obat di etalase.

BAB VI

PENUTUP

Pada bab ini akan diberikan kesimpulan dan saran dari pembuatan “Sistem Informasi Pengelolaan Stok Obat” yang mengambil studi kasus di Apotek Joint Farma.

6.1. Kesimpulan

Berdasarkan implementasi dan analisis sistem informasi pengelolaan stok obat, dapat disimpulkan bahwa:

1. Sistem informasi pengelolaan data obat ini mampu memberikan kemudahan untuk mendapatkan informasi dengan cepat dan tepat.
2. Fasilitas pengingat obat kadaluarsa yang berjalan otomatis pada saat program dijalankan dapat memperkecil resiko dalam keterlambatan meretur obat tersebut ke distributor.
3. Fasilitas pengingat limit obat yang berjalan otomatis pada saat program dijalankan dapat memperkecil resiko dalam memesan obat dari distributor sehingga stok obat tidak menjadi kosong.

6.2. Saran

Saran yang dapat diberikan dalam mengembangkan perangkat lunak ini lebih lanjut adalah sistem informasi ini belum dilengkapi dengan fasilitas *backup* data, diharapkan sistem informasi ini dapat dikembangkan menjadi satu sistem informasi yang lebih lengkap dan sempurna.

DAFTAR PUSTAKA

- Bruegge, Bernd & Dutoit, Allen H. 2004. *Object-Oriented Software Engineering : Using UML, Patterns and Java (ed.2)*. USA : Pearson Education, Inc.
- Hartati, Yustina Sri & Sulasmono. 2006. *APOTEK*. Yogyakarta : Universitas Sanata Dharma.
- Rickyanto, Isak. 2004. *Pemrograman Database Java dengan JDBC*. Yogyakarta : Andi.
- Whitten, Jeffrey L., Bentley, Lonnie D., & with Kevin C Dittman. 2001. *Systems Analysis and Design Methods (ed.5)*. New York : McGraw-Hill.

LAMPIRAN

Diagram Class Desain

