



**SISTEM INFORMASI PERAMALAN KEBUTUHAN REAGEN PADA
LABORATORIUM KLINIK MENGGUNAKAN METODE *EXPONENTIAL*
*SMOOTHING HOLT-WINTERS***

SKRIPSI

Oleh
Yohanis Permadi Polowiwi
NIM 122410101038

FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2019



**SISTEM INFORMASI PERAMALAN KEBUTUHAN REAGEN PADA
LABORATORIUM KLINIK MENGGUNAKAN METODE *EXPONENTIAL*
*SMOOTHING HOLT-WINTERS***

SKRIPSI

Oleh
Yohanis Permadi Polowiwi
NIM 122410101038

FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2019

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

1. Ayahanda Budi Siswanto dan Ibunda Endah Pirenaningdyah;
2. Saudara saya Dodiek Dwi Cahyono dan Deva Dhaharoe Dwi Kharisma;
3. Sahabatku bersama dukungan dan doanya;
4. Teman-temanku Formation angkatan 2012;
5. Saudara-saudara UKM Kesenian ETALASE;
6. Spartan Community yang membimbing saya;
7. Bujus Team yang selalu mendukung dan menghibur;
8. Almamater Program Studi Sistem Informasi Universitas Jember.

MOTO

“Hidup tidak selalu tentang reputasi, tetapi juga tentang edukasi dan dedikasi”

“Luluslah pada waktu yang tepat, bukan tepat waktu” kalimat itu bukan kalimat motivasi, tetapi kalimat penghibur dan pembelaan.

(Yohanis Permadi Polowiwi)

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Yohanis Permadi Polowiwi

NIM : 122410101038

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul “Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters*”, adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika di kemudian hari pernyataan ini tidak benar.

Jember, 15 Mei 2018

Yang menyatakan,

Yohanis Permadi Polowiwi

NIM 122410101038

SKRIPSI

**SISTEM INFORMASI PERAMALAN KEBUTUHAN REAGEN PADA
LABORATORIUM KLINIK MENGGUNAKAN METODE *EXPONENTIAL
SMOOTHING HOLT-WINTERS***

Oleh

Yohanis Permadi Polowiwi

NIM 122410101038

Pembimbing :

Dosen Pembimbing Utama : Drs. Antonius Cahya P, M.App., Sc., Ph.D

Dosen Pembimbing Pendamping : Yanuar Nurdiansyah, S.T., M.CS.

PENGESAHAN PEMBIMBING

Skripsi berjudul “**Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters***” telah diuji dan disahkan pada:

hari, tanggal : Rabu, 15 Mei 2019

tempat : Fakultas Ilmu Komputer Program Studi Sistem Informasi Universitas Jember.

Disetujui oleh:

Pembimbing I,

Pembimbing II,

Drs. Antonius Cahya P, M.App., Sc., Ph.D

NIP. 196909281993021001

Yanuar Nurdiansyah, S.T., M.CS.

NIP. 198201012010121004

PENGESAHAN PENGUJI

Skripsi berjudul “**Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters***” telah diuji dan disahkan pada:

hari, tanggal : Jumat, 24 Mei 2019

tempat : Fakultas Ilmu Komputer Program Studi Sistem Informasi Universitas Jember.

Tim Penguji,

Penguji I,

Penguji II,

Prof. Dr. Saiful Bukhori, ST., M.Kom

NIP. 196811131994121001

Diksy Media Firmansyah, S.Kom., M.Kom.

NIP. 760016853

Mengesahkan

Ketua Program Studi,

Prof. Dr. Saiful Bukhori, ST., M.Kom

NIP. 196811131994121001

RINGKASAN

SISTEM INFORMASI PERAMALAN KEBUTUHAN REAGEN PADA LABORATORIUM KLINIK MENGGUNAKAN METODE *EXPONENTIAL SMOOTHING HOLT-WINTERS*; Yohanis Permadi Polowiwi, 122410101038; 2019; 141 Halaman; Program Studi Sistem Informasi Universitas Jember.

Indonesia merupakan negara yang rawan terjadi bencana alam tsunami yang disebabkan oleh gempa, hampir setiap hari nya terjadi gempa, namun tidak semua gempa dirasakan atau berpotensi tsunami. Ketika terjadi gempa dirasakan atau berpotensi tsunami BMKG selalu menginformasikan sesuai dengan SOP yang ada, salah satunya menyebarkan informasi melalui kepala setiap daerah. Dalam merencanakan pengadaan reagen diperlukan ketepatan antara jumlah pengadaan dengan jumlah kebutuhan. Pengadaan yang berlebihan akan menyebabkan penumpukan reagen di gudang. Penyimpanan yang terlalu lama akan mempengaruhi kualitas reagen sehingga mengurangi kualitas hasil pemeriksaan sampel. Begitu juga sebaliknya pengadaan yang terlalu sedikit akan menyebabkan kekurangan stok reagen dan akan menghambat proses pemeriksaan sampel. Selama ini dalam menentukan jumlah pengadaan reagen pada bulan yang akan datang, pihak manajemen Prosenda sendiri melakukan perkiraan secara manual berdasarkan jumlah pemeriksaan bulan sebelumnya. Oleh karena itu diperlukan aplikasi prediksi kebutuhan reagen. Perhitungan prediksi kebutuhan reagen dilakukan dengan mengimplementasikan metode *Exponential Smoothing Holt-Winters*.

Penelitian ini menggunakan jenis penelitian pengembangan. Model pengembangan perangkat lunak untuk perancangan dan pembangunan dari sistem ini menggunakan model *waterfall*. Terdapat 5 tahapan penelitian yaitu : analisis kebutuhan, desain sistem, implementasi sistem, pengujian sistem, dan pemeliharaan sistem. Analisis kebutuhan merupakan tahap untuk mengumpulkan data dan informasi

yang dibutuhkan dalam membangun sebuah perangkat lunak. Analisis kebutuhan terdiri dari studi pustaka, wawancara, dan gambaran umum sistem. Desain sistem menggunakan model *Unified Modeling Language* (UML) yang dirancang menggunakan konsep *Object Oriented Programming* (OOP). Implementasi sistem menggunakan Bahasa pemrograman *Page Hyper Text Pre-Processor* (PHP) dan manajemen basis data MySQL sistem dilakukan dengan menggunakan 2 cara yaitu *White Box* dan *Black Box Testing*. Pemeliharaan sistem ini dilakukan dengan melakukan perbaikan pada *Error* yang terdapat dalam perangkat lunak.

Data penelitian yang digunakan yaitu data penggunaan reagen tahun 20017-2019 dari Laboratorium Klinik Prosenda Genteng. Perangkat lunak yang dibangun berbasis aplikasi *web* dengan dua jenis aktor untuk aplikasis berbasis *web*. Aplikasi dapat menghitung prakiraan kebutuhan reagen dengan menerapkan metode *Exponential smoothing Holt-Winters*. Dari hasil penelitian yang dilakukan, aplikasi dapat menjadi sebuah solusi sebagai media penyampaian informasi jumlah kebutuhan reagen pada masa yang akan datang.

PRAKATA

Puji syukur kepada Tuhan YME, atas segala limpahan rahmat, hidayat dan karuniaNya maka penulis dapat menyelesaikan skripsi yang berjudul “**Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters***”. Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata Satu (S1) pada Program Studi Sistem Informasi Universitas Jember.

Penyusunan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Oleh karena itu penulis menyampaikan terima kasih kepada:

1. Prof. Dr. Saiful Bukhori, ST., M.Kom selaku Ketua Fakultas Ilmu Komputer Universitas Jember;
2. Drs. Antonius Cahya P, M.App., Sc., Ph.D., selaku Dosen Pembimbing Utama dan Yanuar Nurdiansyah, S.T., M.CS., selaku Dosen Pembimbing Pendamping yang telah meluangkan waktu, pikiran dan perhatian dalam penulisan skripsi;
3. Seluruh Bapak dan Ibu dosen beserta staf karyawan di Program Studi Sistem Informasi Universitas Jember;
4. Ayahanda Budi siswanto dan Ibunda Endah Pirenaningdyah;
5. Saudara saya Dodiék Dwi Cahyono dan Deva Dhaharoe Dwi Kharisma;
6. Teman-teman seperjuangan FORMATION angkatan 2012 dan semua mahasiswa Program Studi Sistem Informasi yang telah menjadi keluarga kecil bagi penulis selama menempuh pendidikan S1;
7. Keluarga besar Unit Kegiatan Mahasiswa Kesenian (UKMK) ETALASE periode 2014-2015, Ocha, Totong, Glen, Hendri, Bubul, Dinda, Alfi, Ines, dan anggota lain yang tidak dapat disebutkan satu persatu;
8. Bujus Team yang selalu menghibur dan memimbing, Bagus, Dimas. Yoga, Rizky, Gede, Bugas, Yoko, dan Riska.
9. Spartan Community yang mendukung dan menghibur penulis.

10. Semua pihak yang tidak dapat disebutkan satu persatu;

Dengan harapan bahwa penelitian ini nantinya akan terus berlanjut dan berkembang kelak, penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Akhirnya penulis berharap, semoga skripsi ini dapat bermanfaat.

Jember, 15 Mei 2019

Penulis

DAFTAR ISI

PERSEMBAHAN.....	i
MOTO.....	ii
PERNYATAAN.....	iii
SKRIPSI.....	iv
PENGESAHAN PEMBIMBING.....	v
PENGESAHAN PENGUJI.....	vi
RINGKASAN	vii
PRAKATA.....	ix
DAFTAR ISI.....	xi
DAFTAR GAMBAR	xvii
DAFTAR TABEL.....	xix
BAB. 1 PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	3
1.3. Tujuan dan Manfaat.....	3
1.3.1. Tujuan	3
1.3.2. Manfaat	4
1.4. Batasan Masalah.....	4
1.5. Sistematika Penulisan.....	5
BAB 2. TINJAUAN PUSTAKA	6
2.1. Penelitian Terdahulu.....	6

2.2.	Laboratorium Klinik.....	7
2.3.	Peramalan (<i>Forecasting</i>)	7
BAB 3. METODOLOGI PENELITIAN.....		13
3.1	Jenis Penelitian	13
3.2	Tempat dan Waktu	13
3.3	Tahapan Penelitian	13
3.3.1	Tahapan Analisis Kebutuhan	14
3.3.2	Tahapan Desain Sistem.....	15
3.3.3	Tahapan Implementasi Sistem	17
3.3.4	Tahapan Pengujian Sistem	17
3.3.5	Tahapan Pemeliharaan Sistem	18
3.4	Gambaran Umum Sistem	18
BAB 4. PENGEMBANGAN SISTEM.....		19
4.1	Deskripsi Umum Sistem.....	19
4.1.1	SOP (<i>statement of purpose</i>)	19
4.2	Pengumpulan Data	20
4.3	Analisis Kebutuhan	20
4.3.1	Kebutuhan Fungsional	20
4.3.2	Kebutuhan Non-Fungsional	20
4.4	Desain Sistem	21
4.4.1	<i>Business Proses</i>	21
4.4.2	<i>Use Case Diagram</i>	22
4.4.3	<i>Use Case Skenario</i>	25

4.4.4	<i>Activity Diagram</i>	29
4.4.5	<i>Sequence Diagram</i>	31
4.4.6	<i>Class Diagram</i>	37
4.4.7	<i>Entity Relationship Diagram (ERD)</i>	37
4.5	Implementasi	39
4.6	Pengujian	42
4.6.1	Metode <i>White Box</i>	43
4.6.2	Metode <i>Black Box</i>	48
4.6.3	Pengujian Tingkat Kebenaran Peramalan	48
BAB 5. HASIL DAN PEMBAHASAN.....		49
5.1	Hasil Pembuatan Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode <i>Exponential Smoothing Holt-winters</i> .	49
5.1.1	Tampilan Awal Sistem.....	49
5.1.2	Fitur Menampilkan <i>Home</i>	50
5.1.3	Fitur Mengelola Data Reagen	51
5.1.4	Fitur Melihat Data Reagen	51
5.1.5	Fitur Mengelola Data Transkasi Reagen Masuk.....	52
5.1.6	Fitur Melihat Data Transaksi Reagen Masuk.....	53
5.1.7	Fitur Mengelola Data Transkasi Reagen Keluar	53
5.1.8	Fitur Melihat Data Transaksi Reagen Keluar.....	53
5.1.9	Fitur Prediksi Kebutuhan Reagen	53
5.2	Implementasi Metode <i>Exponential Smoothing Holt-winters</i>	54

5.2.1	Penentuan Data Penggunaan Reagen.....	55
5.2.2	Perhitungan Manual Metode <i>Exponential Smoothing Holt-winters Multiplicative</i>	57
5.3	Tingkat Kebenaran Peramalan	61
BAB 6. PENUTUP		64
6.1	Kesimpulan.....	64
6.2	Saran.....	65
DAFTAR PUSTAKA		66
LAMPIRAN.....		68
LAMPIRAN A. <i>USE CASE</i> SKENARIO		68
A.1	<i>Use Case</i> Skenario <i>Login</i>	68
A.2	<i>Use Case</i> Skenario Menampilkan Halaman <i>Home</i>	68
A.3	<i>Use Case</i> Skenario Mengelola Data Reagen	69
A.4	<i>Use Case</i> Skenario Melihat Data Reagen	70
A.5	<i>Use Case</i> Skenario Mengelola Transaksi Reagen Masuk	71
A.6	<i>Use Case</i> Skenario Melihat Transaksi Reagen Masuk	72
A.7	<i>Use Case</i> Skenario Mengelola Transaksi Reagen Keluar	73
A.8	<i>Use Case</i> Skenario Melihat Transaksi Reagen Keluar	74
LAMPIRAN B. <i>ACTIVITY DIAGRAM</i>		75
B.1	<i>Activity Diagram Login</i>	75
B.2	<i>Activity Diagram</i> Menampilkan <i>Home</i>	76
B.3	<i>Activity Diagram</i> Mengelola Data Reagen.....	77
B.4	<i>Activity Diagram</i> Melihat Data Reagen	78

B.5	<i>Activity Diagram</i> Mengelola Transaksi Reagen Masuk.....	79
B.6	<i>Activity Diagram</i> Melihat Transaksi Reagen Masuk	80
B.7	<i>Activity Diagram</i> Mengelola Transaksi Reagen Keluar.....	81
B.8	<i>Activity Diagram</i> Melihat Transaksi Reagen Keluar	82
LAMPIRAN C. <i>SEQUENCE DIAGRAM</i>		83
C.1	<i>Sequence Diagram</i> Login dan Menampilkan <i>Home</i>	83
C.2	<i>Sequence Diagram</i> Mengelola Data reagen	84
C.3	<i>Sequence Diagram</i> Melihat Data Reagen	85
C.4	<i>Sequence Diagram</i> Mengelola Transaksi Reagen Masuk.....	86
C.5	<i>Sequence Diagram</i> Melihat Transaksi Reagen Masuk	87
C.6	<i>Sequence Skenario</i> Mengelola Transaksi Reagen Keluar.....	88
C.7	<i>Sequence Skenario</i> Melihat Transaksi Reagen Keluar	89
LAMPIRAN D. KODE PROGRAM		90
D.1	Kode Program <i>Login</i>	90
D.2	Kode Program Menampilkan <i>Home</i> (<i>class controllers/Home</i>)	93
D.3	Kode Program Melihat dan Mengola Data Reagen (<i>class controllers/DataReagen</i>)	94
D.4	Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk (<i>class controllers/DataBeli</i>).....	98
D.5	Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar (<i>class controllers/DataPakai</i>).....	106
D.6	Kode Program <i>class models/M_Trans</i>	114
D.7	Kode Program <i>class models/ M_DetTrans</i>	116
D.8	Kode Program <i>class models/M_Pegawai</i>	117

LAMPIRAN E. PENGUJIAN BLACK BOX.....	118
LAMPIRAN F. PENGUJIAN WHITE BOX.....	122

DAFTAR GAMBAR

Gambar 3. 1 Tahapan metode Waterfall (Pascapraharastyan, Supriyanto, & Sudarmaningtyas, 2014).....	14
Gambar 4. 1 Business Proses Sistem	22
Gambar 4. 2 Use Case Diagram.....	23
Gambar 4. 3 Activity Diagram Prediksi Kebutuhan Reagen	32
Gambar 4. 4 Sequence Diagram Prediksi Kebutuhan Reagen.....	37
Gambar 4. 5 Class Diagram	38
Gambar 4. 6 Entity Relationship.....	39
Gambar 4. 7 Diagram alir fuction hasil()	46
Gambar 5. 1 Tampilan halaman login.....	50
Gambar 5. 2 Tampilan halaman home	50
Gambar 5. 3 Alert Data Kosong Input Reagen	51
Gambar 5. 4 Tampilan Konfirmasi Hapus Data Reagen Baru.....	52
Gambar 5. 5 Tampilan Mengelola Data Reagen Baru	52
Gambar 5. 6 Tampilan Form Prediksi.....	54
Gambar 5. 7 Tampilan Hasil Prediksi	54
Gambar 5. 8 Grafik Data Penggunaan Reagen	57
Gambar B. 1 Activity Diagram Login	75
Gambar B. 2 Activity Diagram Menampilkan Home	76
Gambar B. 3 Activity Diagram Mengelola Data Reagen	77
Gambar B. 4 Activity Diagram Melihat Data Reagen.....	78
Gambar B. 5 Activity Diagram Mengelola Transaksi Reagen Masuk	79
Gambar B. 6 Activity Diagram Melihat Transaksi Reagen Masuk	80
Gambar B. 7 Activity Diagram Mengelola Transaksi Reagen Keluar	81
Gambar B. 8 Activity Diagram Melihat Transaksi Reagen Keluar	82
Gambar C. 1 Sequence Diagram Login dan Menampilkan Home.....	83
Gambar C. 2 Sequence Diagram Mengelola Data Reagen	84

Gambar C. 3 Sequence Diagram Melihat data Reagen.....	85
Gambar C. 4 Sequence Diagram Mengelola Transaksi Reagen Masuk	86
Gambar C. 5 Sequence Diagram Melihat Transaksi Reagen Masuk	87
Gambar C. 6 Sequence Diagram Mengelola Transaksi Reagen Keluar	88
Gambar C. 7 Sequence Diagram Melihat Transaksi Reagen Keluar	89
Gambar F. 1 Diagram Alir Function Tambah Data Reagen	122

DAFTAR TABEL

Tabel 4. 1 Deskripsi pembagian aktor sistem	23
Tabel 4. 2 Deskripsi use case sistem.....	24
Tabel 4. 3 Skenario Prediksi Kebutuhan Reagen.....	28
Tabel 4. 4 Kode Metode Exponential Smoothing Holt-winters function hasil()	40
Tabel 4. 5 Listing program function hasil()	43
Tabel 4. 6 Test case function hasil()	47
Tabel 4. 7 Pengujian Black Box Prediksi Kebutuhan Reagen	48
Tabel 5. 1 Data Penggunaan Reagen Albumin Dyasis	55
Tabel 5. 2 Peramalan Reagen Albumin Dyasis dengan Metode Holt-winters.....	57
Tabel 5. 3 menunjukkan nilai PE peramalan selama bulan Januari 2018.....	61
Tabel 5. 4 contoh perhitungan manual PE Holt-winters.....	62
Tabel 5. 5 Perhitungan MAPE tahun 2018	63
Tabel A. 1.1 Use Case Login	68
Tabel A. 2.1 Use Case Menampilkan Halaman Home	68
Tabel A. 3.1 Use Case Skenario Mengelola Data Reagen.....	69
Tabel A. 4.1 Use Case Skenario Melihat Data Reagen.....	70
Tabel A. 5.1 Use Case Skenario Mengelola Transaksi Reagen Masuk	71
Tabel A. 6.1 Use Case Skenario Melihat Transaksi Reagen Masuk.....	72
Tabel A. 7.1 Use Case Skenario Mengelola Transaksi Reagen Keluar	73
Tabel A. 8.1 Use Case Skenario Melihat Transaksi Reagen Keluar.....	74
Tabel D. 1 Kode Program Login (class controllers/Login).....	90
Tabel D. 2 Kode Program Login (class models/M_Pegawai).....	91
Tabel D. 3 Kode Program Menampilkan Home (class controllers/Home).....	93
Tabel D. 4 Kode Program Melihat dan Mengola Data Reagen (class controllers/DataReagen).....	94
Tabel D. 5 Kode Program Melihat dan Mengola Data Reagen (class models/M_Reagen)	97

Tabel D. 6 Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk (class controllers/DataBeli)	98
Tabel D. 7 Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk (class controllers/ Form TransIn)	102
Tabel D. 8 Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar (class controllers/DataPakai).....	106
Tabel D. 9 Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar (class controllers/ Form TransOut)	109
Tabel D. 10 Kode Program class models/M_Trans	114
Tabel D. 11 Kode Program class models/M_DetTrans	116
Tabel D. 12 Kode Program class models/M_Pegawai.....	117
Tabel F. 1 Test Case function submit().....	122

BAB. 1 PENDAHULUAN

Bab ini merupakan bab awal dari laporan tugas akhir. Pada bab ini akan dibahas tentang latar belakang, perumusan masalah, tujuan dan manfaat, batasan masalah, dan sistematika penulisan.

1.1. Latar Belakang

Kesehatan merupakan faktor penting dalam kehidupan untuk melakukan berbagai aktivitas sehari - hari dan merupakan harta yang paling berharga bagi manusia. Berbagai macam cara dapat dilakukan untuk menjaga dan memantau kondisi tubuh manusia. Menjaga pola hidup dengan mengatur makanan, rajin olahraga, dan rutin melakukan *checkup* adalah salah satu cara untuk menjaga dan memantau kondisi tubuh.

Laboratorium klinik merupakan salah satu perusahaan yang bergerak dibidang pelayanan kesehatan. Laboratorium klinik sendiri merupakan salah satu media untuk memantau kondisi tubuh, mendiagnosis penyakit, dan membantu dokter dalam menentukan suatu penyakit. Laboratorium Klinik dikenal dengan berbagai nama seperti *Medicine Laboratory*, *Clinical Laboratory*, *Clinical Phatology*, atau *Clinical and Phatology Laboratory*. Secara umum di Indonesia lebih dikenal dengan nama Laboratorium Klinik dan Patologi Klinik, yang biasanya dipakai di beberapa rumah sakit besar. Hasil diagnosa seorang penderita yang dikeluarkan oleh laboratorium merupakan penunjang pemeriksaan yang sangat dibutuhkan dokter, karena hasil test yang dikeluarkan membantu dokter untuk menentukan penyakit yang diderita pasien. Laboratorium klinik tidak selalu tergabung dalam rumah sakit, ada laboratorium yang tergabung dengan praktek dokter, atau yang berdiri sendiri (Mulyono, 2007).

Prosenda adalah laboratorium klinik yang menyediakan layanan jasa kesehatan. Prosenda menyediakan jasa *general checkup* sebagai penunjang hasil pemeriksaan

dokter. Beberapa pasien yang dimiliki laboratorium ini adalah pasien yang datang dari rujukan dokter, tetapi tidak sedikit pasien yang datang untuk *checkup* secara pribadi tanpa rujukan dokter.

Kegiatan yang dilakukan Laboratorium Klinik Prosenda meliputi: pengambilan sampel, pemeriksaan sampel (proses laboratorium), dan penyampaian hasil pemeriksaan. Pengambilan sampel dilakukan oleh perawat dari laboratorium yang kemudian diserahkan kepada analis laboratorium untuk di proses. Hasil dari pemeriksaan sampel disampaikan kepada pasien oleh petugas dari laboratorium klinik Prosenda.

Seiring dengan berjalannya waktu, kebutuhan dan kesadaran masyarakat terhadap kesehatan semakin meningkat. Banyaknya permintaan yang masuk, menuntut pihak laboratorium untuk selalu siap melayani baik dari fasilitas maupun tenaga kerja. Manajemen yang baik sangat dibutuhkan dalam rencana pelayanan laboratorium. Salah satunya yaitu penyediaan reagen untuk kebutuhan proses laboratorium.

Reagen adalah zat atau senyawa yang digunakan untuk melihat reaksi pada objek penelitian. Reagen sendiri memiliki sifat mudah berubah pada kondisi yang tidak sesuai. Reagen juga tidak boleh dipakai jika keadaan fisiknya mengalami perubahan atau telah melewati tanggal kadaluarsa karena akan mempengaruhi hasil proses (Sulistyaningsih, 2003).

Dalam merencanakan pengadaan reagen diperlukan ketepatan antara jumlah pengadaan dengan jumlah kebutuhan. Pengadaan yang berlebihan akan menyebabkan penumpukan reagen di gudang. Penyimpanan yang terlalu lama akan mempengaruhi kualitas reagen sehingga mengurangi kualitas hasil pemeriksaan sampel. Begitu juga sebaliknya pengadaan yang terlalu sedikit akan menyebabkan kekurangan stok reagen dan akan menghambat proses pemeriksaan sampel. Hal tersebut menjadi kendala pihak manajemen Prosenda. Selama ini dalam menentukan jumlah pengadaan reagen pada

bulan yang akan datang, pihak manajemen Prosenda sendiri melakukan perkiraan berdasarkan jumlah pemeriksaan bulan sebelumnya.

Berdasarkan kondisi tersebut peramalan bisa dijadikan alat yang efektif dalam membantu memprediksi kebutuhan reagen pada bulan atau periode mendatang. Dalam penelitian ini metode *Exponential Smoothing Holt-winters Multiplicative* merupakan metode yang tepat dalam melakukan peramalan, karena metode tersebut melakukan perhitungan dengan melihat data masa lalu yang pola datanya mengandung unsur musiman (Padang et al, 2013).

1.2. Rumusan Masalah

Berdasarkan latar belakang yang ada, maka didapatkan perumusan masalah sebagai berikut.

1. Bagaimana melakukan peramalan kebutuhan reagen menggunakan metode *Exponential Smoothing Holt-winters Multiplicative* dalam sistem informasi peramalan kebutuhan reagen?
2. Bagaimana merancang dan membangun sistem informasi peramalan kebutuhan reagen dengan metode *Exponential Smoothing Holt-Winters*?

1.3. Tujuan dan Manfaat

Berdasarkan rumusan masalah yang telah disampaikan sebelumnya, maka tujuan dan manfaat yang ingin dicapai sebagai berikut:

1.3.1. Tujuan

Adapun tujuan yang ingin dicapai dalam penelitian ini adalah :

- a. Dapat menerapkan metode *Exponential Smoothing Holt-winters* dalam sistem informasi peramalan kebutuhan reagen.
- b. Dapat memperoleh solusi dalam proses perencanaan pengadaan reagen.

- c. Membangun sistem penunjang keputusan yang dapat membantu Prosenda Laboratorium Klinik dalam pengambilan keputusan pengadaan reagen untuk kebutuhan di masa mendatang.

1.3.2. Manfaat

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

- a. Bagi Instansi

Sebagai masukan bagi instansi tentang jumlah kebutuhan reagen yang harus di penuhi pada bulan yang akan datang,

- b. Bagi Akademis

Hasil penelitian ini diharapkan dapat memberikan pandangan dan masukan bagi siapa saja yang membutuhkan informasi yang berhubungan dengan judul penelitian ini. Selain itu, hasil penelitian ini merupakan upaya untuk menambah perbendaharaan judul penelitian yang ada di Program Studi Sistem Informasi Universitas Jember.

- c. Bagi Peneliti

Mengetahui bagaimana proses penerapan *Exponential Smoothing Holt-winters* pada sistem prediksi kebutuhan reagen laboratorium Klinik

- d. Bagi pihak lain

Penelitian ini dapat dijadikan bahan referensi bagi peneliti lain yang ingin mengembangkan hasil penelitian ini di kemudian hari.

1.4. Batasan Masalah

Terdapat beberapa batasan masalah yang diangkat dalam penelitian ini, antara lain sebagai berikut :

- a. Penelitian dilakukan di Prosenda Laboratorium Klinik Banyuwangi
- b. Perancangan dan pembangunan sistem ini digunakan untuk membantu dalam peramalan kebutuhan reagen pada Laboratorium Klinik Prosenda.

- c. Metode yang digunakan untuk menentukan peramalan kebutuhan reagen adalah metode *Exponential Smoothing Holt-winters Multiplicative* .

1.5. Sistematika Penulisan

Adapun sistematika penulisan skripsi setelah Bab 1. Pendahuluan adalah sebagai berikut:

- a. Bab 2. Tinjauan Pustaka

Bab ini memaparkan tinjauan terhadap hasil-hasil penelitian terdahulu berkaitan dengan masalah yang dibahas, landasan materi dan konsep prediksi, dan kajian teori metode analisis data yang berkaitan dengan masalah dalam penelitian.

- b. Bab 3. Metodologi Penelitian

Bab ini menguraikan tentang tempat dan waktu penelitian, metode penelitian, metode pengumpulan data, metode analisis data, dan teknik pengembangan sistem yang digunakan dalam penelitian.

- c. Bab 4. Pengembangan Sistem

Bab ini berisi uraian tentang langkah-langkah yang dilakukan dalam proses menganalisis dan merancang sistem yang hendak dibangun meliputi desain, kode program, dan pengujian sistem.

- d. Bab 5. Hasil dan Pembahasan

Bab ini memaparkan secara rinci pemecahan masalah melalui analisis yang disajikan dalam bentuk deskripsi dibantu dengan ilustrasi berupa tabel dan gambar untuk memperjelas hasil penelitian.

- e. Bab 6. Penutup

Bab ini terdiri atas kesimpulan atas penelitian yang telah dilakukan dan saran untuk penelitian selanjutn

BAB 2. TINJAUAN PUSTAKA

Pada bagian ini dipaparkan tinjauan yang berkaitan dengan masalah yang dibahas, kajian teori yang berkaitan dengan masalah, dan juga penelitian-penelitian terdahulu.

2.1. Penelitian Terdahulu

Penelitian sebelumnya adalah penerapan metode *Exponential Smoothing Holt-winters* pada “Peramalan Jumlah Penumpang Kereta Api Medan-Rantau Prapat Dengan Metode *Exponential Smoothing Holt-winters*”. Penelitian ini membandingkan peramalan pada jumlah penumpang kereta api Medan-Rantau Prapat dengan 2 jenis metode Pemulusan Eksponensial *Holt-Winter* yaitu Metode Perkalian Musiman (*Holt-winters Multiplicative Method*) dan Metode Penambahan Musiman (*Holt-winters Additive Method*). Dari data penumpang pada masa lalu dilakukan perhitungan menggunakan Metode *Holt-winters Multiplicative* dan Metode *Holt-winters Additive*. Kemudian dari hasil kedua metode tersebut dicari nilai erornya menggunakan perhitungan *MSE* dan *MAPE* dengan nilai $\alpha = 0,2$ $\beta = 0,1$ $\gamma = 0,5$. Hasil dari perbandingan kedua metode tersebut menunjukkan bahwa metode *Holt-winters Multiplicative* memiliki nilai eror yang lebih kecil. Hasil yang ditunjukkan dalam peramalan periode ke-61 dengan metode *Holt-winters Additive* memiliki nilai *MSE* = 12.454.563,78 dan *MAPE*=0,8, sedangkan dari perhitungan menggunakan metode *Holt-winters Multiplicative*, memiliki nilai *MSE* = 10.288.981 dan *MAPE* = 0,6. Dari perbandingan tersebut dapat disimpulkan bahwa metode *Holt-winters Multiplicative* lebih baik dalam melakukan peramalan jumlah penumpang yang memiliki pola data musiman dibandingkan metode *Holt-winters Additive*.

2.2. Laboratorium Klinik

Laboratorium klinik adalah tempat dimana berbagai macam tes dilakukan pada spesimen biologis untuk mendapatkan informasi kesehatan pasien. (Andriyas, tt)

Kegiatan – kegiatan dasar pada laboratorium terbagi menjadi empat, yaitu: (Mulyono, 2007)

1. Kegiatan pelayanan Kimia Klinik adalah kegiatan yang menetapkan keadaan biokimiawi dan metabolisme tubuh, seperti kadar gula, fungsi hati, kadar kolesterol, lemak darah, dan fungsi ginjal.
2. Kegiatan pelayanan Hematologis adalah kegiatan yang menelaah unsur seluler dan fungsional darah, seperti kadara hemoglobin, fungsi penghentian darah.
3. Kegiatan pelayanan mikrobiologis adalah kegiatan yang memeriksa keberadaan jasad renik penyebab inveksi, seperti virus, bakteri, dan jamur.
4. Kegiatan pelayanan imunologis adalah kegiatan yang memeriksa keberadaan antibodi.

Hasil dari pemeriksaan tersebut akan menjadi pendukung keputusan dokter dalam menentukan penyakit yang diderita pasien.

2.3. Peramalan (*Forecasting*)

Peramalan (*forecasting*) merupakan bagian vital bagi setiap organisasi bisnis dan untuk setiap pengampilan keputusan manajemen yang sangat signifikan. Peramalan merupakan hal yang sangat penting dalam perencanaan jangka panjang pada sautu perusahaan. Hampir semua peramalan didasarkan pada asumsi bahwa masa lampau akan berulang (Padang et al, 2013).

Bedasrkan teknik yang digunakan, metode peramalan dapat digolongkan menjadi dua kategori, yaitu: (Hakimah, Muhima, & Yustina, 2015)

1. Metode peramalan kualitatif

Peramalan kualitatif adalah metode peramalan yang berdasarkan dari data kualitatif masa lalu. Hasil peramalan yang dibuat bergantung kepada orang yang menyusun.

2. Metode peramalan kuantitatif

Peramalan kuantitatif adalah peramalan yang berdasarkan atas data kuantitatif masa lalu. Hasil peramalan yang dibuat bergantung pada metode yang digunakan. Dengan metode yang berbeda maka hasil yang diramalkan akan berbeda. Semakin kecil nilai eror pada suatu metode semakin baik metode tersebut digunakan untuk peramalan.

Metode kuantitatif dapat diterapkan apabila terdapat kondisi sebagai berikut :

- a. Tersedia informasi (data) tentang masa lalu.
- b. Informasi tersebut dapat dikuantitatifkan dalam bentuk angka.

Dapat diasumsikan pola data masa lalu akan terjadi kembali pada masa mendatang.

2.4. Metode *Exponential Smoothing Holt-winters*

Metode peramalan *Holt-winters* adalah metode yang digunakan untuk peramalan dengan data yang memiliki pola musiman dan mengandung unsur tren. Metode *Holt-winters* memiliki tiga persamaan penghalusan, yaitu persamaan penghalusan keseluruhan, penghalusan tren, dan persamaan penghalusan musiman (Suwandi et al, 2015).

Metode *Holt-winters* terbagi menjadi 2 bagian, yaitu: (Padang et al, 2013)

1. Penghalusan eksponensial *Holt-winters* dengan metode *multiplicative* :

Penghalusan keseluruhan

$$S_t = \alpha \frac{X_t}{I_{t-l}} + (1 - \alpha)(S_{t-1} + b_{t-1}) \quad (2.1)$$

Penghalusan tren

$$T_t = \beta(S_t - S_{t-1}) + (1 - \beta)b_{t-1} \quad (2.2)$$

Penghalusan musiman

$$I_t = \gamma \frac{X_t}{S_t} + (1 - \gamma)I_{t-L} \quad (2.3)$$

Ramalan

$$F_{t-m} = (S_t + b_t m)I_{1-L+m} \quad (2.4)$$

2. Penghalusan eksponensial *Holt-winters* dengan metode *additive* :

Penghalusan keseluruhan

$$S_t = \alpha(X_t - I_{1-L}) + (1 - \alpha)(S_{t-1} + b_{t-1}) \quad (2.5)$$

Penghalusan tren

$$b_t = \beta(S_t - S_{t-1}) + (1 - \beta)b_{t-1} \quad (2.6)$$

Penghalusan musiman

$$I_t = \gamma(X_t - S_t) + (1 - \gamma)I_{t-L} \quad (2.7)$$

Ramalan

$$F_{t-m} = S_t + b_t m + I_{t-L+m} \quad (2.8)$$

Dimana :

X_t : nilai aktual pada periode t

α : parameter penghalusan keseluruhan ($0 < \alpha < 1$)

- β : parameter penghalusan tren ($0 < \beta < 1$)
- γ : parameter penghalusan musiman ($0 < \gamma < 1$)
- I : faktor penyesuaian musiman
- L : panjang periode dalam satu siklus
- F_{t-m} : ramalan untuk m periode kedepan dari t

Dalam metode ini, nilai awal sangat di butuhkan untuk menentukan $t-1$ yang belum tersedia. Artinya nilai ramalan S_{t-1} belum ada (Padang et al, 2013).

Misalnya pada persamaan (2.1), dimana X_t adalah nilai aktual yang paling baru, S_t adalah ramalan yang paling akhir. S_{t+1} adalah ramalan pada periode berikutnya. Jika $t = 2$, maka persamaan (2.1) akan menjadi,

$$S_2 = \alpha \frac{X_2}{I_{2-L}} + (1 - \alpha)(S_1 + b_1) \quad (2.9)$$

Untuk memperoleh nilai S_2 , maka nilai S_1 harus diketahui. Nilai S_1 memiliki persamaan sebagai berikut :

$$S_1 = \alpha \frac{X_1}{I_{1-L}} + (1 - \alpha)(S_0 + b_0) \quad (2.10)$$

Dari persamaan (2.10) dapat dilihat bahwa nilai S_0 , dan b_0 tidak dapat ditemukan. Sehingga perlu ditetapkan nilai awal.

Rumus metode ini dapat digunakan dengan mengambil beberapa nilai awal dengan persamaan sebagai berikut:

Untuk model *additive*:

$$S_L = \frac{1}{L}(X_1 + X_2 + \dots + X_L) \quad (2.11)$$

$$T_L = \frac{1}{K} \left(\frac{X_{L+1} - X_1}{L} + \frac{X_{L+2} - X_2}{L} + \dots + \frac{X_{L+k} - X_k}{L} \right) \quad (2.12)$$

$$I_k = X_k - S_L \quad (2.13)$$

Untuk nilai awal *multiplicative*, perhitungan nilai awal untuk S_L dan b_L sama dengan *additive* kecuali untuk penghalusan musiman menggunakan

$$I_k = \frac{X_k}{S_L} \quad (2.14)$$

2.5. Mean Absolute Percentage Error (MAPE)

Suatu peramalan tidak selalu tepat dengan nilai aktualnya. Hasil peramalan akan selalu berbeda dengan nilai aktual. Perbedaan nilai peramalan dengan nilai aktual disebut kesalahan peramalan. Meskipun perbedaan hasil peramalan tidak dapat dihindari, akan tetapi tujuan peramalan adalah agar kesalahan yang didapat bernilai sekecil mungkin. MAPE adalah salah satu metode untuk menghitung nilai eror atau kesalahan dalam peramalan. Metode ini menghitung perbedaan antara nilai aktual dengan nilai hasil peramalan. Nilai perbedaan itu kemudian diabsolutkan, dan dihitung ke dalam persentase terhadap data asli. Berikut ini adalah persamaan dalam menghitung nilai MAPE

$$MAPE = \sum_{t=1}^n \frac{|PE_t|}{n} \quad (2.15)$$

Dimana :

$$PE_t = \text{kesalahan persentase} = \frac{X_t - F_t}{X_t} * 100$$

$$E_t = \text{kesalah periode } t = X_t - F_t$$

$$X_t = \text{data aktual periode } t$$

$$n = \text{banyak periode } t$$

BAB 3. METODOLOGI PENELITIAN

Bab ini menjelaskan tentang metode yang digunakan dalam merancang dan membangun sistem menggunakan metode pengembangan.

3.1 Jenis Penelitian

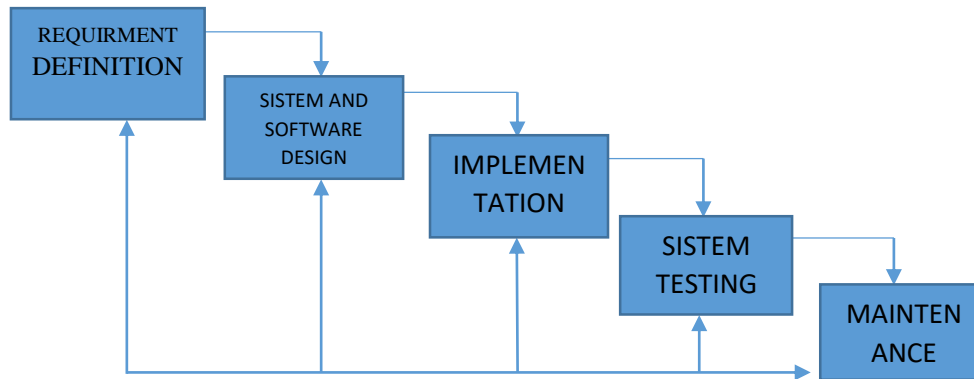
Jenis penelitian yang akan digunakan dalam penelitian ini adalah penelitian pengembangan. Jenis penelitian pengembangan ini karena penelitian ini mengembangkan sistem informasi peramalan kebutuhan reagen pada laboratorium klinik Prosenda.

3.2 Tempat dan Waktu

Penelitian ini dilakukan di Prosenda Laboratorium Klinik yang beralamat di Jalan Jember No 34 B Genteng, Banyuwangi. Waktu penelitian dilakukan selama 2 (dua) bulan, dimulai pada bulan Juli 2016 sampai dengan bulan Agustus 2016.

3.3 Tahapan Penelitian

Metode pengembangan yang digunakan adalah *Sistem Development Life Cycle (SDLC)* dengan model *waterfall*. Model *waterfall* adalah model yang melakukan pendekatan secara sistematis dan urut mulai dari analisis, desain, *coding*, *testing*, dan pemeliharaan yang dapat dilihat pada Gambar 3.1 (Pascapraharastyan et al, 2014).



Gambar 3. 1 Tahapan metode *Waterfall* (Pascapraharastyan, Supriyanto, & Sudarmaningtyas, 2014)

3.3.1 Tahapan Analisis Kebutuhan

Tahapan ini menganalisis kebutuhan dengan mencari permasalahan yang muncul. Data dan permasalahan dapat diperoleh dari wawancara dan menganalisis dokumen yang berhubungan dengan penelitian.

Analisis kebutuhan yang dilakukan pada penelitian ini dengan wawancara kepada pihak terkait yaitu bagian manajemen Laboratorium Klinik Prosenda dan mengambil data yang dibutuhkan yaitu data transaksi reagen selama kurang lebih dua tahun terakhir.

3.3.2 Tahapan Desain Sistem

Pembuatan desain sistem dalam penelitian ini menggunakan *Unified Modeling Language* (UML) yang dirancang dengan konsep *Object-Oriented Programming* (OOP). Dengan menggunakan UML akan dibangun 7 diagram desain sistem yaitu:

1. *Business Process*
2. *Use Case Diagram*
3. *Scenario*
4. *Activity Diagram*
5. *Sequence Diagram*
6. *Class Diagram*
7. *Entity Relationship Diagram*

a. *Business Process*

Business Process merupakan gambaran masuknya data hingga keluarnya hasil data dari proses yang dijalankan sistem. Data yang dibutuhkan oleh sistem (*input*) berupa data penggunaan reagen dalam kurun waktu 2 tahun terakhir, keluaran data yang dihasilkan (*output*) adalah prediksi kebutuhan reagen pada periode bulan selanjutnya.

b. *Use Case Diagram*

Use case diagram adalah dokumentasi yang menggambarkan fitur dan aktor yang dapat mengakses fitur – fitur pada sistem. Gambaran yang ditekankan pada *use case* adalah “apa” yang dapat dilakukan oleh sistem, bukan “bagaimana” sistem bekerja. Aktor yang dapat mengakses sistem ini adalah pihak Gudang sebagai admin dan Manajer yang berperan untuk memantau transaksi berlangsung. Diagram *use case* dapat sangat membantu dalam penyusunan *requirements* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua fitur yang ada pada sistem.

c. *Use Case Scenario*

Use case scenario berfungsi untuk menjelaskan alur sistem dari fitur yang ada pada *use case diagram*, skenario juga berfungsi menjelaskan reaksi yang akan terjadi pada sistem setelah menerima aksi dari aktor. Use case scenario menggambarkan keadaan normal saat setiap aktor mengakses sistem dan keadaan alternatif yang terjadi pada kondisi tertentu.

d. *Activity Diagram*

Activity Diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, meliputi bagaimana masing-masing aktivitas dimulai, *decision* yang mungkin terjadi, dan bagaimana aktivitas berakhir. *Activity Diagram* juga menggambarkan aktivitas sistem setelah menerima aksi dari aktor.

e. *Sequence Diagram*

Sequence Diagram menggambarkan aliran logika dan interaksi yang terjadi di dalam sistem sesuai urutan dijalankannya sistem tersebut. Interaksi tersebut meliputi, aktor yang mengakses fitur sesuai dengan *use case diagram*, tampilan sistem, *controller*, model, dan pesan yang disampaikan jika terjadi suatu kondisi tertentu..

f. *Class Diagram*

Class diagram merupakan sebuah spesifikasi yang menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class Diagram* menggambarkan struktur dan deskripsi *class* serta hubungan antar *class*, sehingga memudahkan dalam proses pengkodean.

g. *Entity Relationship Diagram (ERD)*

ERD merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan data yang mempunyai hubungan antar relasi.

3.3.3 Tahapan Implementasi Sistem

Setelah desain sistem selesai dilakukan maka akan dilanjutkan ketahap pembuatan sistem dan implementasi. Pembuatan sistem meliputi penulisan kode program dan pembuatan basis data. Penulisan kode dilakukan menggunakan *tools* Sublime Text 2 sebagai *editor* dengan bahasa pemrograman HTML, PHP, CSS yang terintegrasi dengan DBMS MySQL sebagai manajemen basis data.

3.3.4 Tahapan Pengujian Sistem

Pengujian digunakan untuk mengetahui sejauh mana sistem ini dapat berjalan. Serta untuk mengetahui letak kekurangan yang ada pada sistem ini. Terdapat dua metode yang digunakan untuk pengujian yaitu *White Box* dan *Black Box*. Sedangkan pengujian yang dilakukan untuk mengukur tingkat akurasi metode menggunakan *MAPE*.

h. *White Box Testing*

White box adalah pengujian yang dilakukan dengan membandingkan aplikasi dengan modul yang telah dibuat. Jika hasil yang dikeluarkan sistem tidak sesuai dengan modul yang telah ditetapkan maka perlu ada baris kode, paramater, atau variabel yang perlu dicek dan diperbaiki ulang

i. *Black Box Testing*

Berbeda dengan *white box testing*, *Black box testing* merupakan metode pengujian perangkat lunak yang memeriksa fungsionalitas aplikasi. Pengujian ini lebih difokuskan kepada kerja program. Pengujian dilakukan dengan melibatkan mahasiswa secara acak sebagai pengguna.

j. *MAPE*

MAPE adalah salah satu metode untuk menghitung nilai eror atau kesalahan dalam peramalan. Metode ini menghitung perbedaan antara nilai aktual dengan

nilai hasil peramalan. Nilai perbedaan itu kemudian diabsolutkan, dan dihitung ke dalam persentase terhadap data asli.

3.3.5 Tahapan Pemeliharaan Sistem

Pemeliharaan merupakan proses perawatan sistem setelah sistem digunakan oleh pengguna. Pada tahap pemeliharaan ini dilakukan untuk mengatasi masalah sistem yang ada saat melakukan *testing White box* dan *Black box*. Jika hasil *testing* telah didapatkan, pihak *developer* segera memperbaiki kekurangan dan kesalahan sesuai hasil *testing*.

3.4 Gambaran Umum Sistem

Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* ini dibangun untuk mempermudah pengguna dalam melakukan transaksi reagen dan melakukan prediksi kebutuhan reagen pada bulan berikutnya. Hal ini dilakukan karena transaksi reagen dan peramalan dilakukan secara manual dan belum terkomputerisasi.

Sistem ini akan diterapkan pada Laboratorium Klinik Prosenda sesuai dengan data-data yang diberikan. Sistem ini akan menerima masukan data berupa data transaksi dan pemakaian reagen setiap bulan. Kumpulan data tadi akan di proses menggunakan metode *Exponential Smoothing Holt-winters Additive*. Proses dari perhitungan metode tersebut akan menghasilkan angka prediksi sebagai informasi kebutuhan reagen pada bulan berikutnya.

BAB 4. PENGEMBANGAN SISTEM

Bab ini menguraikan mengenai analisis kebutuhan, desain, implementasi, dan pengujian sistem yang digunakan dalam proses pengembangan sistem informasi peramalan kebutuhan reagen pada laboratorium klinik dengan metode *Exponential Smoothing Holt-winters*. Di mana tahapan analisis kebutuhan hingga pengujian yang dilakukan sesuai dengan metode pengembangan *waterfall*.

4.1 Deskripsi Umum Sistem

Deskripsi umum dari sistem informasi peramalan kebutuhan reagen pada laboratorium klinik dengan metode *Exponential Smoothing Holt-winters* yang dibangun dalam penelitian ini akan dijelaskan lebih detail pada SOP (*statement of purpose*) sistem dan fungsi sistem.

4.1.1 SOP (*statement of purpose*)

Sistem informasi peramalan kebutuhan reagen pada laboratorium klinik dengan metode *Exponential Smoothing Holt-winters* merupakan sistem yang mampu menampilkan hasil peramalan berdasarkan data kebutuhan pada masa lalu yang diproses dengan perhitungan menggunakan metode *Exponential Smoothing Holt-winters*. Data yang digunakan adalah data kebutuhan reagen setiap bulan terhitung sejak tahun 2016. Data tersebut kemudian dihitung sehingga menghasilkan angka prediksi kebutuhan reagen untuk periode bulan selanjutnya. Hasil prediksi tersebut akan menjadi pendukung dan bahan pertimbangan dalam menentukan jumlah pembelian reagen periode bulan selanjutnya.

4.2 Pengumpulan Data

Pengumpulan data pada penelitian ini dilakukan melalui wawancara langsung ke kantor laboratorium Klinik Prosenda Genteng, Banyuwangi tepatnya pada bagian pengadaan dan gudang. Data yang diperlukan adalah data penggunaan reagen setiap bulan dalam kurun waktu 2 tahun terakhir.

4.3 Analisis Kebutuhan

Berdasarkan metode pengembangan sistem model *waterfall*, tahapan awal yang dilakukan adalah tahapan analisis kebutuhan. Tahapan ini dilakukan analisis terhadap kebutuhan-kebutuhan dari sistem yang dibangun, baik berupa kebutuhan fungsional maupun kebutuhan non-fungsional. Hasil analisis tersebut sangat mempengaruhi fungsionalitas sistem yang dibangun untuk dapat digunakan sesuai dengan fungsi dan kebutuhan pengguna yaitu memprediksi kebutuhan reagen pada periode bulan selanjutnya.

4.3.1 Kebutuhan Fungsional

Kebutuhan fungsional sistem berisi fitur-fitur inti yang harus dipenuhi dalam sistem agar sistem mampu difungsikan sesuai dengan tujuan dan kebutuhan pengguna terhadap sistem itu sendiri. Kebutuhan fungsional dari Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* yaitu:

- a. Sistem mampu mengelola data reagen (*view, insert, edit, delete*).
- b. Sistem mampu mengelola data transaksi pembelian dan pemakaian reagen (*view, insert, delete*).
- c. Sistem mampu mengelola data prediksi kebutuhan reagen (*view, insert*).

4.3.2 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional merupakan fitur-fitur yang dimiliki untuk mendukung sistem dalam memenuhi fungsionalitasnya untuk dapat memenuhi kebutuhan dari

pengguna. Kebutuhan non-fungsional dari Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* yaitu:

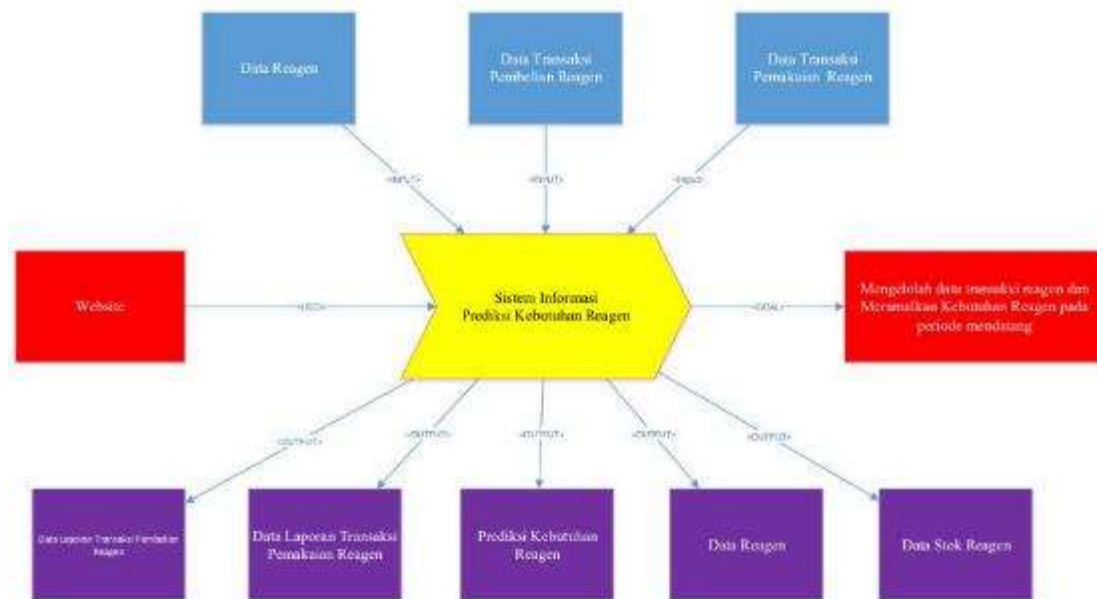
- a. Sistem bekerja sesuai dengan fungsinya dan dapat dijalankan pada semua komputer dan semua *browser* yang berbeda.
- b. Tampilan dan bahasa komunikasi sistem mudah dimengerti oleh pengguna untuk memberikan kenyamanan pemakaian dan memudahkan pengoperasian.
- c. Sistem menggunakan *username* dan *password* untuk autentifikasi akses admin terhadap sistem.

4.4 Desain Sistem

Tahapan yang dilakukan setelah melakukan analisis kebutuhan sistem yaitu tahap perencanaan pembangunan sistem yang dapat digambarkan dengan desain sistem. Desain sistem yang dibuat untuk Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* meliputi *business proses*, *use case diagram*, *scenario*, *activity diagram*, *sequence diagram*, *class diagram*, dan *entity relationship diagram*.

4.4.1 Business Proses

Selain dapat dideskripsikan dalam sebuah SOP (*Statement Of Purpose*), gambaran umum Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* dapat digambarkan melalui sebuah *business proses*. Seperti yang dapat kita lihat pada Gambar 4.1 menggambarkan data-data yang digunakan sebagai masukan, data keluaran, *uses* sistem yang dibangun hingga *goal* dari dibangunnya sistem sendiri.

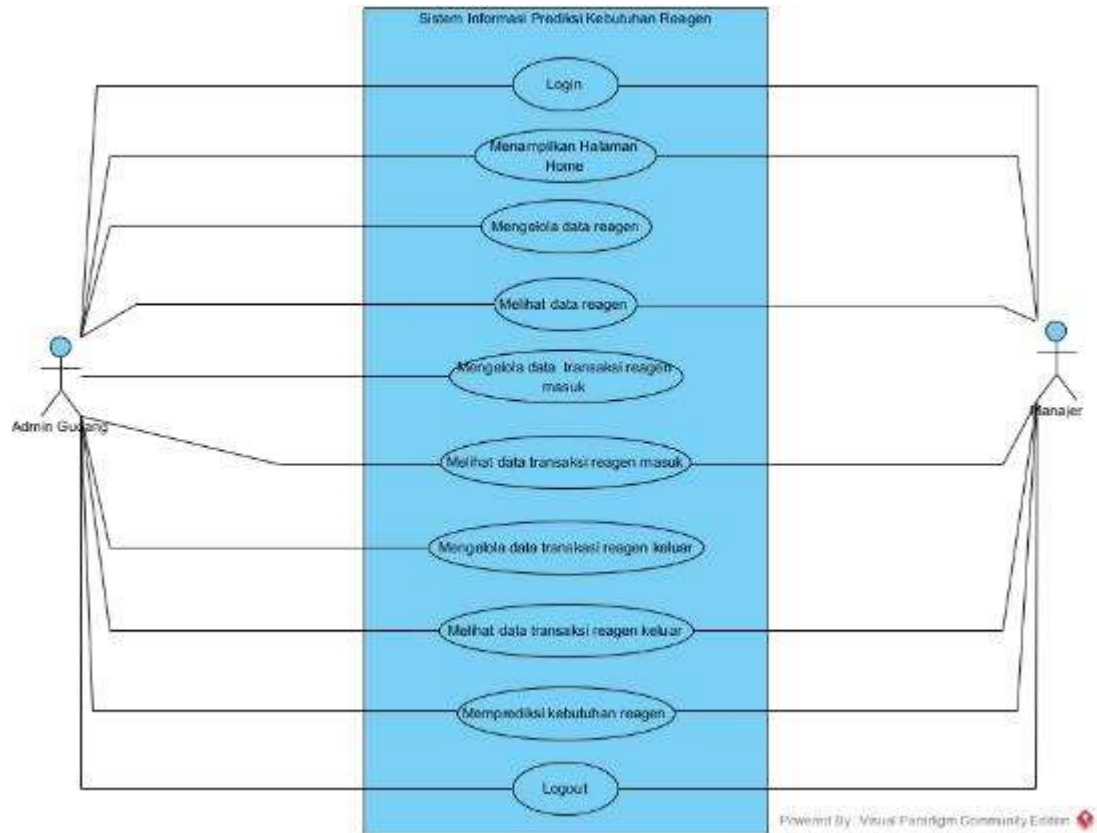


Gambar 4. 1 *Business Proses* Sistem

4.4.2 Use Case Diagram

Use case diagram digunakan untuk dapat menggambarkan interaksi antara aktor dengan Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* yang akan dibangun. Melalui *use case* diagram dapat diketahui interaksi yang dapat dilakukan aktor terhadap sistem sesuai dengan hak akses yang dimiliki oleh masing-masing aktor atau pengguna. Pada Gambar 4.2 digambarkan *use case* diagram yang terdiri atas dua aktor dengan delapan *use case*.

Berdasarkan *use case* diagram pada Gambar 4.2 terdapat dua pengguna, yaitu *admin* dan manajer. Berikut deskripsi dari masing-masing aktor yang dapat dilihat pada tabel 4.1



Gambar 4. 2 Use Case Diagram

Tabel 4. 1 Deskripsi pembagian aktor sistem

No	Aktor	Deskripsi
1	Admin Gudang	Aktor ini memiliki hak untuk mengelola data reagen, mengolah transaksi reagen yang masuk dan keluar serta melakukan prediksi kebutuhan reagen.
2	Manajer	Aktor hanya memiliki hak akses terbatas. Aktor ini hanya dapat melihat data reagen, melihat data transaksi reagen yang masuk dan keluar serta melakukan prediksi kebutuhan reagen.

Selain memiliki dua aktor, dalam *use case* diagram juga terdapat sepuluh *use case*.

Deskripsi dari *use case* tersebut dapat dilihat pada tabel 4.2

Tabel 4. 2 Deskripsi *use case* sistem

No	<i>Use Case</i>	Deskripsi
1	2	3
1	<i>Login</i>	<i>Use case</i> yang menggambarkan proses masuk ke dalam Sistem.
2	Menampilkan <i>Home</i>	<i>Use case</i> yang menggambarkan proses menampilkan halaman <i>home</i> yang berisi deskripsi singkat tentang sistem.
3	Mengelola Data Reagen	<i>Use case</i> yang menggambarkan proses pengolahan data reagen meliputi menambahkan, merubah, dan menghapus data.
4	Melihat Data Reagen	<i>Use case</i> yang menggambarkan proses menampilkan data reagen.
5	Mengelola Data Transaksi Reagen Masuk	<i>Use case</i> yang menggambarkan proses pengolahan data transaksi reagen masuk meliputi menambahkan, merubah, dan menghapus data.
6	Melihat Data Transaksi Reagen Masuk	<i>Use case</i> yang menggambarkan proses menampilkan data transaksi reagen yang masuk.
7	Mengelola Data Transaksi Reagen Keluar	<i>Use case</i> yang menggambarkan proses pengolahan data transaksi reagen keluar meliputi menambahkan, merubah, dan menghapus data.

Dilanjutkan

Lanjutan

1	2	3
8	Melihat Data Transaksi Reagen Keluar	<i>Use case</i> yang menggambarkan proses menampilkan data transaksi reagen yang keluar.
9	Memprediksi Kebutuhan Reagen	<i>Use case</i> yang menggambarkan proses menampilkan hasil prediksi yang dihitung dengan metode <i>Exponential Smoothing Holt-winters</i>
10	Logout	<i>Use case</i> yang menggambarkan proses keluar dari Sistem.

4.4.3 Use Case Skenario

Use case skenario adalah dokumentasi terhadap kebutuhan fungsional sistem. *Use case* skenario Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* adalah sebagai berikut:

a. Skenario *Login*

Skenario *login* merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario *login*. Pada skenario tersebut dapat dilihat bagaimana sebuah alur *login* berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon reaksi tersebut sehingga aktor telah melakukan *login* pada sistem tersebut dan berhasil masuk sistem sebagai user yang dapat mengoperasikan sistem. Proses lengkap skenario *login* dapat dilihat pada Lampiran A Tabel A.1.

b. Skenario Menampilkan Halaman *Home*

Skenario menampilkan *Home* merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal *use case* skenario menampilkan *home*. Pada skenario tersebut dapat dilihat bagaimana sebuah alur menampilkan *home* berjalan

setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor dapat mengakses halaman *home*. Proses lengkap skenario menampilkan *home* dapat dilihat pada Lampiran A Tabel A.2.

c. Skenario Mengelola Data Reagen

Skenario mengelola data reagen merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario mengelola data reagen seperti memasukan, mengubah, dan menghapus data. Pada skenario tersebut dapat dilihat bagaimana sebuah alur mengelola data reagen berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut, sehingga aktor dapat mengelola data reagen sesuai keinginan. Proses lengkap skenario mengelola data reagen dapat dilihat pada Lampiran A Tabel A.3.

d. Skenario Melihat Data Reagen

Skenario melihat data reagen merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario melihat data reagen. Pada skenario tersebut dapat dilihat bagaimana sebuah alur melihat data reagen berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah melihat data reagen. Proses lengkap skenario melihat data reagen dapat dilihat pada Lampiran A Tabel A.4

e. Skenario Mengelola Data Transaksi Reagen Masuk

Skenario mengelola data transaksi reagen masuk merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario mengelola data transaksi reagen masuk. Pada skenario tersebut dapat dilihat bagaimana sebuah alur mengelola data transaksi reagen masuk berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah mengelola data transaksi reagen masuk. Proses lengkap skenario mengelola data transaksi reagen masuk dapat dilihat pada Lampiran A Tabel A. 5.

f. Skenario Melihat Data Transaksi Reagen Masuk

Skenario melihat data transaksi reagen masuk merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario melihat data transaksi reagen masuk. Pada skenario tersebut dapat dilihat bagaimana sebuah alur melihat data transaksi reagen masuk berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah melihat data transaksi reagen masuk. Proses lengkap skenario melihat data transaksi reagen masuk dapat dilihat pada Lampiran A Tabel A. 6.

g. Skenario Mengelola Data Transaksi Reagen Keluar

Skenario mengelola data transaksi reagen keluar merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario mengelola data transaksi reagen keluar. Pada skenario tersebut dapat dilihat bagaimana sebuah alur mengelola data transaksi reagen keluar berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah mengelola data transaksi reagen keluar. Proses lengkap skenario mengelola data transaksi reagen keluar dapat dilihat pada Lampiran A Tabel A.7.

h. Skenario Melihat Data Transaksi Reagen Keluar

Skenario melihat data transaksi reagen keluar merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario melihat data transaksi reagen keluar. Pada skenario tersebut dapat dilihat bagaimana sebuah alur melihat data transaksi reagen keluar berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah melihat data transaksi reagen keluar. Proses lengkap skenario melihat data transaksi reagen keluar dapat dilihat pada Lampiran A Tabel A.8.

i. Skenario Prediksi Kebutuhan Reagen

Skenario memprediksi kebutuhan reagen merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif *use case* skenario memprediksi kebutuhan reagen. Pada skenario tersebut dapat dilihat bagaimana sebuah alur melihat data transaksi reagen keluar berjalan setelah mendapat aksi dari aktor dan reaksi dari sistem untuk merespon aksi tersebut sehingga aktor telah memprediksi kebutuhan reagen. Proses lengkap skenario memprediksi kebutuhan reagen dapat dilihat pada Tabel 4.3.

Tabel 4. 3 Skenario Prediksi Kebutuhan Reagen

ID	USC 12
Nama Use Case	Memprediksi Kebutuhan Reagen
Aktor	Manajer
Entry Condition	Aktor memprediksi kebutuhan reagen pada periode mendatang
Exit Condition	Aktor telah memiliki angka prediksi kebutuhan reagen
Skenario Normal Memprediksi Kebutuhan Reagen	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda Manajer
2. Memilih menu Prediksi kebutuhan	
	3. Menampilkan <i>form</i> peramalan
4. Mengisi <i>form</i> peramalan	
5. Klik tombol “Prediksi”	
	6. Mengeksekusi <i>form</i> peramalan
	7. Menampilkan Tabel prediksi
Skenario Alternatif Isi <i>form</i> tidak sesuai	
5a. Klik tombol “Prediksi”	
	7a. Menampilkan pesan “Lakukan pengisian dengan benar”

4.4.4 Activity Diagram

Activity Diagram menggambarkan aliran aktivitas Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* yang akan dibangun. Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* memiliki sepuluh *activity diagram* yaitu sebagai berikut:

a. *Activity Diagram Login*

Activity diagram Login dilakukan oleh admin gudang dan manajer Laboratorium Klinik. *Activity diagram login* menjelaskan tentang bagaimana aktor memasukkan data username dan password untuk mendapatkan akses sistem dan sistem merespon dengan menjalankan fungsi melakukan autentifikasi hak akses semua aktor. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B.1.

b. *Activity Diagram Menampilkan Home*

Activity diagram menampilkan home dapat dilakukan oleh admin gudang dan manajer Laboratorium Klinik Prosenda. *Activity diagram menampilkan home* berisi alur bagaimana aktor dapat mengakses halaman *home* yang di dalamnya terdapat deskripsi singkat sistem dan *stock* reagen. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B.2.

c. *Activity Diagram Mengelola Data Reagen*

Activity diagram mengelola data reagen dapat dilakukan oleh admin gudang Laboratorium Klinik Prosenda. *Activity diagram mengelola data reagen* menjelaskan alur bagaimana admin dapat mengelola data reagen pada sistem meliputi menambahkan, merubah, dan menghapus data. Ketika aktor memilih tombol tambah maka sistem akan menampilkan *form* pengisian data reagen, sedangkan jika memilih ikon *edit* maka sistem akan menampilkan *form* berisi data reagen yang akan diubah, begitu juga ketika aktor memilih ikon hapus maka sistem

akan menampilkan konfirmasi hapus data. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B.3.

d. *Activity Diagram* Melihat Data Reagen

Activity diagram melihat data reagen dapat dilakukan oleh admin gudang dan manajer Laboratorium Klinik Prosenda. *Activity diagram* ini menjelaskan alur ketika aktor memilih menu Data reagen dan sistem bereaksi dengan menampilkan daftar reagen yang dimiliki Laboratorium Klinik Prosenda. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B.4.

e. *Activity Diagram* Mengelola Data Transaksi Reagen Masuk

Activity diagram mengelola data transaksi reagen masuk dapat dilakukan oleh admin gudang Laboratorium Klinik Prosenda. *Activity diagram* mengelola data transaksi reagen masuk pada sistem meliputi menambahkan, merubah, dan menghapus data. Ketika aktor memilih tombol tambah transaksi maka sistem akan menampilkan *form* transaksi data reagen, sedangkan ketika aktor memilih ikon hapus maka sistem akan menampilkan konfirmasi hapus data. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B. 5.

f. *Activity Diagram* Melihat Data Transaksi Reagen Masuk

Activity diagram melihat data transaksi reagen masuk dapat dilakukan oleh admin gudang dan manajer Laboratorium Klinik Prosenda. *Activity diagram* melihat data transaksi reagen masuk berisi alur diagram yang menjelaskan ketika aktor memilih menu transaksi dan sub menu transaksi masuk maka sistem dapat menampilkan data transaksi reagen masuk setiap bulan. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B. 6.

g. *Activity Diagram* Mengelola Data Transaksi Reagen Keluar

Activity diagram mengelola data transaksi reagen keluar dapat dilakukan oleh admin gudang Laboratorium Klinik Prosenda. *Activity diagram* mengelola data

transaksi reagen keluar pada sistem meliputi menambahkan, merubah, dan menghapus data. Ketika aktor memilih tombol tambah transaksi maka sistem akan menampilkan *form* transaksi data reagen, sedangkan ketika aktor memilih ikon hapus maka sistem akan menampilkan konfirmasi hapus data. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B. 7.

h. *Activity Diagram* Melihat Data Transaksi Reagen Keluar

Activity diagram melihat data transaksi reagen keluar dapat dilakukan oleh admin gudang dan manajer Laboratorium Klinik Prosenda. *Activity diagram* melihat data transaksi reagen keluar berisi alur diagram yang menjelaskan ketika aktor memilih menu transaksi dan sub menu transaksi keluar maka sistem dapat menampilkan data transaksi reagen keluar setiap bulan. Proses lengkapnya dapat dilihat pada Lampiran B Gambar B. 8.

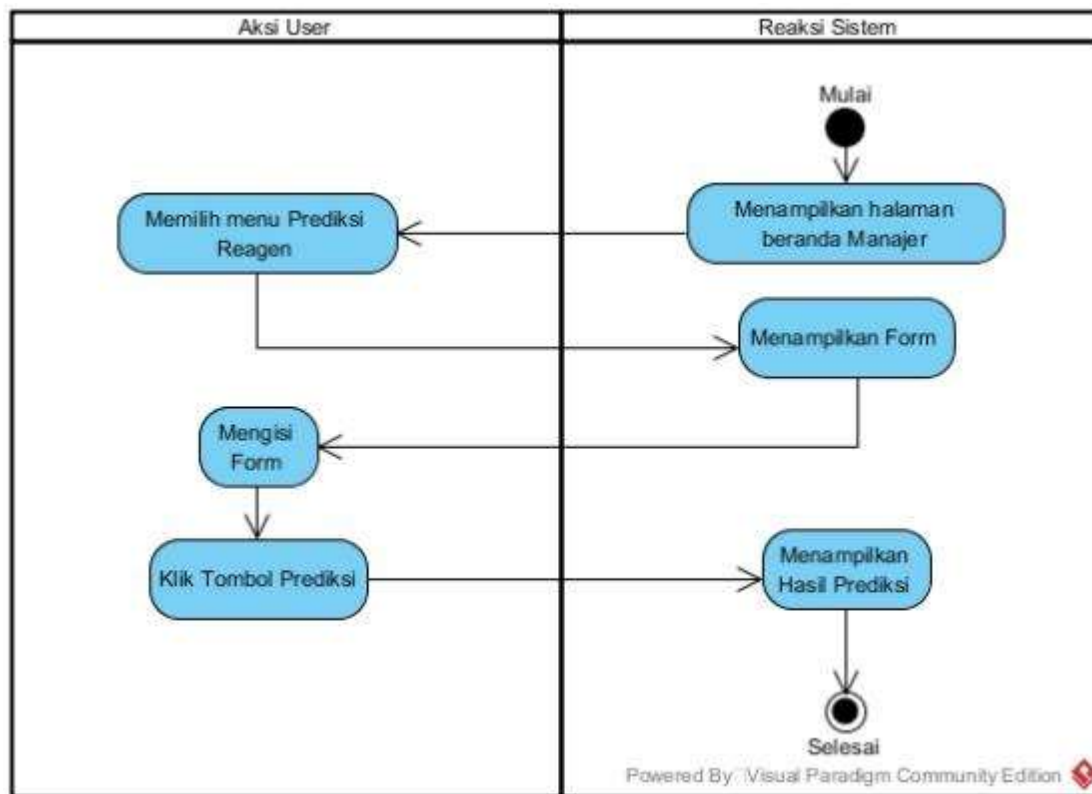
i. *Activity Diagram* Memprediksi Kebutuhan Reagen

Activity diagram memprediksi kebutuhan reagen dapat dilakukan oleh admin gudang dan manajer Laboratorium Klinik Prosenda. *Activity diagram* memprediksi kebutuhan reagen berisi alur diagram yang menjelaskan ketika aktor memilih menu prediksi reagen maka sistem akan menampilkan *form* prediksi yang wajib diisi oleh aktor sehingga sistem dapat memproses perhitungan metode *Exponential Smoothing Holt-winters* dan menampilkan halaman hasil prediksi. Proses lengkapnya dapat dilihat pada Gambar 4.3.

4.4.5 *Sequence Diagram*

Sequence diagram adalah dokumentasi berupa diagram yang menampilkan interaksi di dalam sistem sesuai dengan urutan dijalankannya sistem tersebut. Interaksi tersebut meliputi aktor yang mengakses fitur tersebut sesuai dengan *use case diagram*, tampilan sistem, *controller*, *model*, dan pesan yang disampaikan jika terjadi kondisi tertentu. *Sequence diagram* digunakan untuk menggambarkan skenario dan aliran

logika dalam sistem sehingga tahapan yang dibangun jelas dan teratur. *Sequence* diagram dari Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* adalah sebagai berikut:



Gambar 4. 3 Activity Diagram Prediksi Kebutuhan Reagen

a. *Sequence* Diagram Login

Penggambaran *sequence* diagram *login* digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case login*. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Pada *Sequence Login* dapat dilihat alur perjalanan fitur *Login* yang dilakukan oleh aktor (Admin dan Manajer). Aktor mengisi username, password dan menekan tombol *login*. Selanjutnya *controller*

Login akan melakukan pengecekan terhadap database yang diproses pada *model Login* apakah username dan password dari aktor benar atau tidak, jika benar sistem akan menampilkan halaman *Home*, jika tidak maka akan kembali ke halaman *login* dengan menampilkan peringatan *Login Gagal*. *Sequence* diagram *login* lebih lengkap akan dijelaskan pada Lampiran C Gambar C.1.

b. *Sequence* Diagram Menampilkan Halaman *Home*

Penggambaran *sequence* menampilkan *home* digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* menampilkan halaman *home*. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Pada *Sequence* diagram menampilkan *home* fitur tersebut dapat diakses oleh dua aktor yaitu admin gudang dan Manajer. Diagram ini menggambarkan alur dimana aktor memilih menu *Home* dan selanjutnya *controller Home* memanggil tampilan *home*. *Sequence* diagram menampilkan *home* lebih lengkap akan dijelaskan pada Lampiran C Gambar C.1.

c. *Sequence* Diagram Mengelola Data Reagen

Penggambaran *sequence* diagram mengelola data reagen digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* mengelola data reagen. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Pada *sequence* ini fitur bisa diakses oleh admin gudang Laboratorium Klinik Prosenda. Diagram ini menggambarkan alur ketika aktor memilih menu Tambah Data Reagen lalu mengisi *form* reagen dan memilih tombol simpan, maka *controller* *DataReagen* akan memproses data tersebut ke dalam database melalui model *M_Reagen*. Selanjutnya *controller* *DataReagen* menampilkan halaman Data Reagen kembali sebagai tanda penambahan data berhasil, sedangkan jika pengisian kurang tepat atau kosong maka *controller* akan menampilkan view gagal menambahkan data. Diagram ini juga menggambarkan alur ketika Aktor memilih ikon ubah data maka *conttroller* akan memanggil data yang akan diubah sesuai id

di database melalui model, selanjutnya *controller* menampilkan *form* berisi data yang telah dipilih untuk diubah. Selanjutnya *controller* DataReagen menampilkan halaman Data Reagen kembali sebagai tanda penambahan data berhasil, sedangkan jika pengisian kurang tepat atau kosong maka *controller* akan menampilkan view gagal menambahkan data. Alur ketika aktor memilih ikon Hapus data maka *controller* akan memproses id data yang akan di hapus melalui model dan menampilkan halaman data Reagen sebagai tanda bahwa data berhasil dihapus. *Sequence* diagram mengelola data reagen lebih lengkap akan dijelaskan pada Lampiran C Gambar C.2.

d. *Sequence* Diagram Melihat Data Reagen

Penggambaran *sequence* diagram mengelola data reagen digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* mengelola data reagen. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Pada *sequence* ini fitur bisa diakses oleh admin gudang Laboratorium Klinik Prosenda. Diagram ini menggambarkan alur ketika aktor memilih menu Data Reagen maka *controller* DataReagen akan memanggil data di database melalui model M_Reagen. Selanjutnya *controller* DataReagen menampilkan halaman Data Reagen dengan menampilkan data reagen yang sudah diambil melalui model. *Sequence* diagram melihat data reagen lebih lengkap akan dijelaskan pada Lampiran C Gambar C.3.

e. *Sequence* Diagram Mengelola Data Transaksi Reagen Masuk

Penggambaran *sequence* diagram mengelola data transaksi reagen masuk digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* mengelola data transaksi reagen masuk. Pada *sequence* ini fitur bisa diakses oleh admin gudang Laboratorium Klinik Prosenda. Diagram ini menggambarkan alur ketika aktor memilih menu Tambah Data Transaksi Reagen lalu mengisi *form* transaksi reagen dan memilih tombol simpan, maka *controller* DataBeli akan

memproses data tersebut ke dalam database melalui model M_Trans. Selanjutnya *controller* DataReagen menampilkan halaman Data Transaksi Reagen Masuk kembali sebagai tanda penambahan data berhasil, sedangkan jika pengisian kurang tepat atau kosong maka *controller* akan menampilkan view gagal menambahkan data. Alur ketika aktor memilih ikon Hapus data maka *controller* akan memproses id data yang akan di hapus melalui model dan menampilkan halaman data Transaksi Reagen sebagai tanda bahwa data berhasil dihapus. *Sequence* diagram mengelola data transaksi reagen masuk lebih lengkap akan dijelaskan pada Lampiran C Gambar C.4.

f. *Sequence* Diagram Melihat Data Transaksi Reagen Masuk

Penggambaran *sequence* diagram melihat data transaksi reagen masuk digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* melihat data transaksi reagen masuk. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Diagram ini menggambarkan alur ketika aktor memilih menu Data Transaksi Reagen maka *controller* DataBeli akan memanggil data di database melalui model M_Trans. Selanjutnya *controller* DataBeli menampilkan halaman Data Transaksi Reagen dengan menampilkan data transaksi reagen yang sudah diambil melalui model. *Sequence* diagram melihat data transaksi reagen masuk lebih lengkap akan dijelaskan pada Lampiran C Gambar C.5.

g. *Sequence* Diagram Mengelola Data Transaksi Reagen Keluar

Penggambaran *sequence* diagram mengelola data transaksi reagen keluar digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* mengelola data transaksi reagen keluar. Pada *sequence* ini fitur bisa diakses oleh admin gudang Laboratorium Klinik Prosenda. Diagram ini menggambarkan alur ketika aktor memilih menu Tambah Data Transaksi Reagen lalu mengisi *form* transaksi reagen dan memilih tombol simpan, maka *controller* DataPakai akan memproses data tersebut ke dalam database melalui model M_Trans. Selanjutnya

controller DataPakai menampilkan halaman Data Transaksi Reagen Keluar kembali sebagai tanda penambahan data berhasil, sedangkan jika pengisian kurang tepat atau kosong maka *controller* akan menampilkan view gagal menambahkan data. Alur ketika aktor memilih ikon Hapus data maka *controller* akan memproses id data yang akan di hapus melalui model dan menampilkan halaman data Transaksi Reagen sebagai tanda bahwa data berhasil dihapus. *Sequence* diagram mengelola data transaksi reagen masuk lebih lengkap akan dijelaskan pada Lampiran C Gambar C.6..

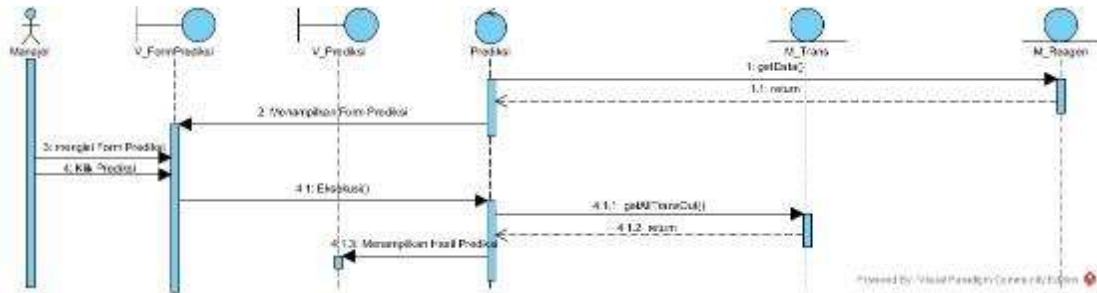
h. *Sequence* Diagram Melihat Data Transaksi Reagen Keluar

Penggambaran *sequence* diagram melihat data transaksi reagen keluar digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* melihat data transaksi reagen keluar. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Diagram ini menggambarkan alur ketika aktor memilih menu Data Transaksi Reagen maka *controller* DataPakai akan memanggil data di database melalui model M_Trans. Selanjutnya *controller* DataPakai menampilkan halaman Data Transaksi Reagen dengan menampilkan data transaksi reagen yang sudah diambil melalui model. *Sequence* diagram melihat data transaksi reagen masuk lebih lengkap akan dijelaskan pada Lampiran C Gambar C.7.

i. *Sequence* Diagram Memprediksi Kebutuhan Reagen

Penggambaran *sequence* diagram memprediksi kebutuhan reagen digunakan untuk menjelaskan fungsi atau *method* yang akan dibuat pada *use case* memprediksi kebutuhan reagen. Masing – masing *class* akan ditampilkan secara *visual* dengan gambar. Pada *sequence* ini fitur bisa diakses oleh admin gudang Laboratorium Klinik dan Manajer Prosenda. Diagram ini menggambarkan ketika aktor memilih tombol prediksi maka *controller* akan memproses data pemakaian reagen melalui model M_Prediksi dan akan ditampilkan oleh *controller* dalam

wujud tampilan prediksi. *Sequence* diagram memprediksi kebutuhan reagen lebih lengkap akan dijelaskan pada Gambar 4.4.



Gambar 4. 4 *Sequence* Diagram Prediksi Kebutuhan Reagen

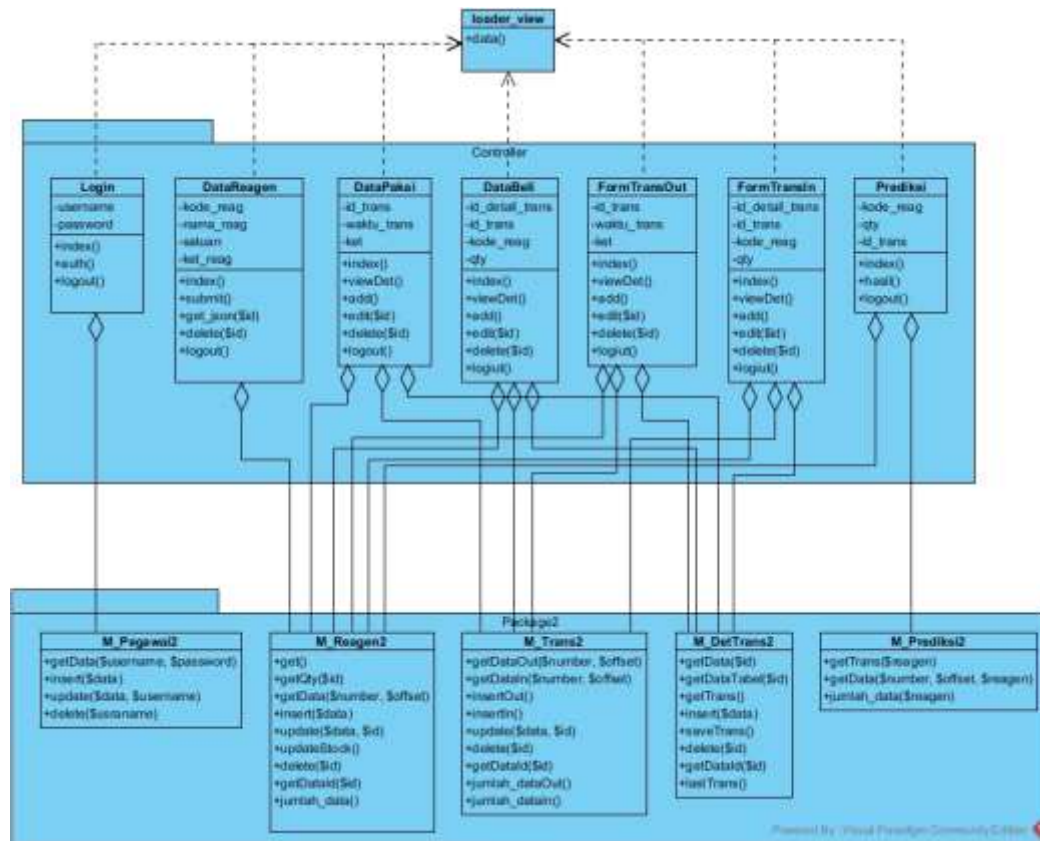
4.4.6 *Class* Diagram

Class diagram menggambarkan hubungan antara kelas yang digunakan untuk membangun suatu sistem. Dalam paradigma OOP (*Object Oriented Programming*) terdapat 3 jenis kelas yaitu *model*, *view* dan *controller*. Berdasarkan *sequence* diagram yang telah dibangun, *class* diagram sistem informasi peramalan kebutuhan reagen pada laboratorium klinik menggunakan metode *exponential smoothing Holt-winters* terdiri dari 5 *class* model dan 7 *class* *controller*. Pada *class* tersebut memiliki fungsi masing-masing, *class* *controller* berfungsi untuk mengakses data yang ada pada *class* Model. Jika kita lihat garis *controller* Prediksi terhubung dengan model M_Prediksi dan M_Reagen yang artinya ada transaksi data antara *controller* dan model tersebut. *Class* diagram sistem dapat dilihat pada Gambar 4.1.

4.4.7 *Entity Relationship* Diagram (ERD)

Entity Relationship Diagram (ERD Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* merupakan gambaran komponen, relasi dan struktur *database* yang

digunakan dalam pembuatan sistem. ERD yang diimplementasikan pada sistem ini terdiri dari 3 entitas yang dapat dilihat pada Gambar 4.6.



Gambar 4. 5 Class Diagram



Gambar 4. 6 Entity Relationship

4.5 Implementasi

Desain yang telah dibuat akan diimplementasi ke dalam kode program. Beberapa hal yang dilakukan dalam tahap implementasi antara lain:

- Penulisan kode program menggunakan bahasa pemrograman *Page Hyper Text Pre-Processor (PHP)* dengan bantuan *framework Code Igniter (CI)*.
- Manajemen basis data menggunakan *DBMS MySQL*.

Kode program perhitungan metode *Exponential Smoothing Holt-winters* terdapat di *class Prediksi* pada *package controller* dan *class M_Prediksi* pada *package models*. Penulisan kode program perhitungan *Exponential Smoothing Holt-winters* dapat dilihat pada Tabel 4.4

Tabel 4. 4 Kode Metode *Exponential Smoothing Holt-winters function* hasil()

Kode
<pre> public function hasil() { \$reagen=\$this->input->post('namarea'); \$data['detail']=\$this->M_Prediksi- >getTrans(\$reagen)->result_array(); \$data['details']=\$this->M_Prediksi- >getTrans(\$reagen)->num_rows(); \$pegLogin = \$this->session- >userdata('pegLogin'); \$periode=12; \$alpha=\$this->input->post('alpha'); \$beta=\$this->input->post('beta'); \$gama=\$this->input->post('gama'); \$m=1; \$totin= 0; \$totb= 0; for (\$i=0; \$i < \$periode; \$i++) { \$data['l'][\$i]=0; \$data['b'][\$i]=0; \$data['s'][\$i]=0; \$a = \$data['detail'][\$i]['qty']; \$totin = \$totin + \$a; \$totb = \$totb + ((\$data['detail'][\$periode+\$i]['qty'] -\$data['detail'][\$i]['qty'])/\$periode); } \$data['l'][\$periode-1] = \$totin / \$periode; \$data['b'][\$periode-1] = (1/\$periode)*\$totb; for (\$i=0; \$i <\$data['details']; \$i++) { \$data['s'][\$i]= \$data['detail'][\$i]['qty']/\$data['l'][\$periode-1]; \$data['f'][\$i]= 0; } for (\$i=12; \$i<\$data['details']; \$i++) { // // \$t=\$periode+\$i; \$data['l'][\$i] = (\$alpha*(\$data['detail'][\$i]['qty']/\$data['s'][\$i- \$periode]))+(1-\$alpha)*(\$data['l'][\$i-1]+\$data['b'][\$i- 1]); \$data['b'][\$i] = (\$beta*(\$data['l'][\$i]- \$data['l'][\$i-1]))+(1-\$beta)*\$data['b'][\$i-1]; </pre>

Dilanjutkan

```

        $data['s'][$i] =
($gama*($data['detail'][$i]['qty']/$data['l'][$i]))+((1-
$gama)*$data['s'][$i-$periode]);
        $data['f'][$i+$m] =
($data['l'][$i]+($data['b'][$i]*$m))*$data['s'][$i-
$periode+$m];
    }

    if (empty($pegLogin)) {
        $this->session->set_flashdata('gagal',
'Silahkan Login');
        redirect(base_url());
    }
    else{
        // $data['detail']=$this->M_Prediksi-
>getTrans()->result_array();
        $this->session->set_flashdata('sukses',
$pegLogin['nama'].' anda login sebagai
'.$pegLogin['level']);

        $jumlah_data = $this->M_Prediksi-
>jumlah_data($reagen);
        $this->load->library('pagination');
        $config['base_url'] =
base_url().'index.php/Prediksi/index/';
        $config['total_rows'] = $jumlah_data;
        $config['per_page'] = 12;
        $from = $this->uri->segment(3);
        $this->pagination->initialize($config);

        $data['trans'] = $this->M_Prediksi-
>getData($config['per_page'], $from, $reagen);
        // echo "<pre>";
        // print_r($data);die();
        if ($pegLogin['level'] ==
'Administrator') {
            $this->load->view('admin/header');
            $this->load->view('admin/menubar',
$pegLogin);

            $this->load-
>view('admin/prediksi/V_Prediksi', $data);
            $this->load->view('admin/footer');
        }
        else{

```

Lanjutan

```

                                $this->session-
>set_flashdata('sukses',          'Selamat          datang
'. $pegLogin['nama'].'          anda          login          sebagai
'. $pegLogin['level']);
                                $this->load-
>view('manajer/header');
                                $this->load-
>view('manajer/menubar', $pegLogin);
                                $this->load-
>view('manajer/V_DataReagen', $data);
                                $this->load-
>view('manajer/footer');
                                }
}}

```

Pada Tabel 4.4 merupakan kode perhitungan metode *Exponential Smoothing Holt-winters* pada package *controller* perhitungan. Pada kode program tersebut menjelaskan tahapan atau langkah – langkah proses perhitungan metode *Exponential Smoothing Holt-winters* yang menghasilkan perhitungan kriteria yang ada. Hasil perhitungan tersebut adalah prediksi kebutuhan reagen periode selanjutnya, hasil perhitungan dapat dilihat dari data penggunaan reagen selama dua tahun terakhir.

4.6 Pengujian

Tahapan pengujian sistem merupakan suatu tahapan yang dilakukan secara sistematis untuk menguji dan mengevaluasi sistem dengan menggunakan sebuah metode pengujian sistem untuk mengetahui apakah kebutuhan sistem telah terpenuhi dan sistem layak untuk digunakan oleh pengguna. Agar pengujian yang dilakukan lebih valid, maka tahap pengujian Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik Menggunakan Metode *Exponential Smoothing Holt-winters* ini dilakukan dengan menggunakan dua metode, yaitu *white box* dan *black box* serta *MAPE* untuk melihat akurasi perhitungan metode.

4.6.1 Metode White Box

White Box Testing merupakan pengujian pada modul pengkodean program untuk menjamin kode program bebas dari kesalahan sintaks maupun logika. Dalam pengujian white box terdapat beberapa tahapan pembuatan dokumentasi pengujian yaitu *cyclomatic complexity (CC)*, *listing program*, dan *test case*. Pengujian untuk fitur prediksi kebutuhan reagen dapat dilihat pada Gambar 4.7 dan 4.8 serta Tabel 4.6. Sedangkan untuk fitur yang lain dapat dilihat pada lampiran E.

a. Listing Program

Listing program merupakan baris-baris kode yang nantinya akan diuji. Setiap langkah dari kode-kode yang ada diberi nomor baik menjalankan *statement* biasa atau penggunaan kondisi dalam program.

Tabel 4. 5 Listing program *function* hasil()

Kode
<pre> public function hasil() { \$reagen=\$this->input->post('namarea'); \$data['detail']=\$this->M_Prediksi- >getTrans(\$reagen)->result_array(); \$data['details']=\$this->M_Prediksi- >getTrans(\$reagen)->num_rows(); \$pegLogin = \$this->session->userdata('pegLogin'); \$periode=12; \$alpha=\$this->input->post('alpha'); \$beta=\$this->input->post('beta'); \$gama=\$this->input->post('gama'); \$m=1; \$totin= 0; \$totb= 0; for (\$i=0; \$i < \$periode; \$i++) { \$data['l'][\$i]=0; \$data['b'][\$i]=0; \$data['s'][\$i]=0; \$a = \$data['detail'][\$i]['qty']; \$totin = \$totin + \$a; </pre>

Dilanjutkan

Lanjutan

```

        $totb = $totb +
(($data['detail'][$periode+$i]['qty']
-$data['detail'][$i]['qty'])/$periode);
    }
    $data['l'][$periode-1] = $totin / $periode;
    $data['b'][$periode-1] = (1/$periode)*$totb;
    for ($i=0; $i <$data['details']; $i++) {
        $data['s'][$i]=
$($data['detail'][$i]['qty']/$data['l'][$periode-1]);
        $data['f'][$i]= 0;
    }
    for ($i=12; $i<$data['details']; $i++) {
        // // $t=$periode+$i;
        $data['l'][$i] =
($alpha*($data['detail'][$i]['qty']/$data['s'][$i-
$periode]))+(1-$alpha)*($data['l'][$i-1]+$data['b'][$i-1]);
        $data['b'][$i] = ($beta*($data['l'][$i]-
$($data['l'][$i-1])))+(1-$beta)*$($data['b'][$i-1]);
        $data['s'][$i] =
($gama*($data['detail'][$i]['qty']/$($data['l'][$i])))+(1-
$gama)*$($data['s'][$i-$periode]);
        $data['f'][$i+$m] =
($($data['l'][$i]+$($data['b'][$i]*$m))*$($data['s'][$i-
$periode+$m]));
    }

    if (empty($pegLogin)) {
        $this->session->set_flashdata('gagal',
'Silahkan Login');
        redirect(base_url());
    }
    else{
        // $data['detail']=$this->M_Prediksi-
>getTrans()->result_array();
        $this->session->set_flashdata('sukses',
$pegLogin['nama'].' anda login sebagai '.$pegLogin['level']);

        $jumlah_data = $this->M_Prediksi-
>jumlah_data($reagen);
        $this->load->library('pagination');
        $config['base_url'] =
base_url(). 'index.php/Prediksi/index/';
        $config['total_rows'] = $jumlah_data;
        $config['per_page'] = 12;
        $from = $this->uri->segment(3);
    }

```

Dilanjutkan

Lanjutan

```

        $this->pagination->initialize($config);

        $data['trans']      =      $this->M_Prediksi-
>getData($config['per_page'], $from, $reagen);
        // echo "<pre>";
        // print_r($data);die();
        if ($pegLogin['level'] == 'Administrator') {
            $this->load->view('admin/header');
            $this->load->view('admin/menuabar',
$pegLogin);

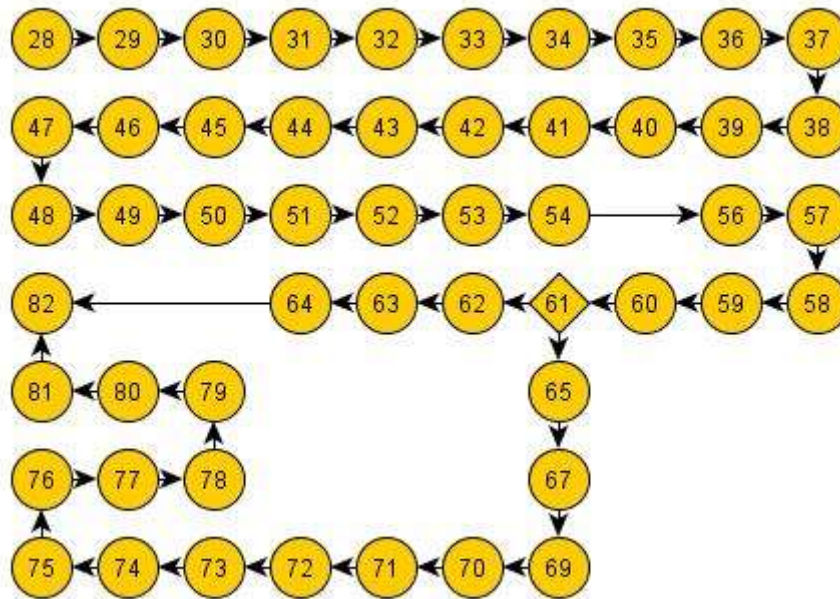
            $this->load-
>view('admin/prediksi/V_Prediksi', $data);
            $this->load->view('admin/footer');
        }
        else{
            $this->session->set_flashdata('sukses',
'Selamat datang '.$pegLogin['nama'].' anda login sebagai
'.$pegLogin['level']);
            $this->load->view('manajer/header');
            $this->load->view('manajer/menuabar',
$pegLogin);

            $this->load-
>view('manajer/V_DataReagen', $data);
            $this->load->view('manajer/footer');
        }
    }}

```

b. Diagram alir

Diagram alir merupakan notasi yang digunakan untuk merepresentasikan aliran kontrol. Aliran kontrol yang digambarkan merupakan hasil penomoran dari listing program. Diagram alir digambarkan dengan node-node (simpul) yang dihubungkan dengan *edge-edge* (garis) yang menggambarkan alur jalannya program.



Gambar 4. 7 Diagram alir *function* hasil()

c. Kompleksitas siklomatik (*cyclomatic complexity*)

Kompleksitas siklomatik merupakan metrik perangkat lunak yang menyediakan ukuran kuantitatif dari kompleksitas logis suatu program. Bila digunakan dalam konteks teknik pengujian jalur dasar, nilai yang dihitung untuk kompleksitas siklomatik mendefinisikan jumlah jalur independen dalam basis set suatu program. Perhitungan kompleksitas siklomatik menggunakan persamaan berikut ini.

function submit()

$$V(G) = E - N + 2 = 52 - 52 + 2 = 2$$

$$V(G) = E - N + 2$$

Keterangan :

V(G) : Kompleksitas siklomatik

E : Jumlah *edge*

N : Jumlah *node*

d. *Test Case*

Pada bagian ini diberikan contoh data yang akan memaksa pelaksanaan jalur di basis set. Data yang dieksekusi dimasukkan ke dalam grafik alir apakah sudah melewati basis set yang tersedia. Sistem telah memenuhi syarat kelayakan perangkat lunak jika salah satu jalur yang dieksekusi setidaknya satu kali.

Tabel 4. 6 *Test case function* hasil()

<i>Test case function</i> hasil()	
<i>Test case</i>	Jika menampilkan halaman prediksi <i>reagen</i>
Target yang diharapkan	Menampilkan halaman prediksi <i>reagen</i>
Hasil pengujian	Benar
<i>Path/Jalur</i>	29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-56-57-58-59-60-61-62-63-64-65-67-68-69-70-71-72-73-74-75-76-77-78-79-80-81-82

4.6.2 Metode *Black Box*

Dokumentasi hasil pengujian fitur prediksi kebutuhan reagen dapat dilihat pada Tabel 4.7. Sedangkan untuk fitur yang lain dapat dilihat pada Lampiran E (Pengujian *Black Box*).

Tabel 4. 7 Pengujian Black Box Prediksi Kebutuhan Reagen

No	Fitur	Kasus	Hasil	Keterangan	
				Berhasil	Tidak
1.	Prediksi <i>Reagen</i>	Ketika klik salah satu <i>form</i> , menampilkan beberapa pilihan data	Menampilkan beberapa data berdasarkan <i>form</i> yang dipilih	V	
		Ketika klik tombol <i>simpan</i>	Menampilkan hasil prediksi daftar penggunaan reagen berdasarkan pilihan dari <i>user</i>	V	

4.6.3 Pengujian Tingkat Kebenaran Peramalan

Pengujian ini akan dijelaskan pada sub bab 5.3. Tujuan dari pengujian ini adalah untuk mengetahui apakah metode *exponential smoothing Holt-winters* mampu memberikan nilai peramalan yang mendekati nilai aktual kebutuhan reagen pada Laboratorium Klinik Prosenda Genteng. Pengujian ini dihitung berdasarkan rumus *Percentage Error* (PE) dan MAPE yang rumusnya dapat dilihat pada persamaan (2.15).

BAB 6. PENUTUP

Bab ini berisi kesimpulan dan saran dari penelitian yang telah dilakukan. Kesimpulan dan saran ini diharapkan mampu menjadi acuan untuk melakukan penelitian selanjutnya.

6.1 Kesimpulan

Kesimpulan dari penelitian yang telah dilakukan adalah:

1. Peramalan jumlah pengadaan reagen pada Laboratorium Klinik Prosenda Genteng dilakukan menggunakan metode *exponential smoothing* khususnya *exponential smoothing Holt-winters Multiplicative* dengan nilai $\alpha = 0.1$, $\beta = 0.1$, $\gamma = 0.9$ dan data aktual penggunaan reagen selama tahun 2017 hingga 2019. Dari hasil peramalan didapatkan Nilai MAPE pada tahun 2018 sebesar 10,58%.
2. Perancangan dan pembangunan Sistem Informasi Peramalan Kebutuhan Reagen Pada Laboratorium Klinik dengan Metode *Exponential Smoothing Holt-winters* ini menggunakan metode SDLC model *waterfall*. Adapun hambatan dalam pembangunan sistem menggunakan model *waterfall* ini adalah proses analisis data yang membutuhkan waktu yang lama. Hal tersebut dikarenakan tahap analisis data merupakan langkah awal yang menentukan tahapan selanjutnya dalam proses pembangunan. Analisis data yang kurang matang akan menimbulkan kesalahan pada tahap-tahap selanjutnya dan dalam model *waterfall* ini, apabila terdapat kesalahan kita tidak bisa langsung memperbaiki kesalahan pada tahap kesalahan itu terjadi, melainkan harus mengulang dari tahap awal. Namun karena sistem informasi peramalan jumlah kebutuhan reagen ini tergolong sistem berskala kecil, maka hambatan tersebut tidak begitu menjadi masalah besar.

6.2 Saran

Adapun saran yang ditujukan untuk memberikan masukan yang lebih baik yaitu :

1. Sistem informasi peramalan jumlah kebutuhan reagen diharapkan pada pengembang selanjutnya memberikan fitur yang lebih lengkap hingga mencakup pengelolaan setiap jenis reagen yang ada di Laboratorium Klinik Prosenda Genteng.
2. Proses peramalan jumlah kebutuhan reagen dilakukan dengan metode peramalan yang berbeda atau mengkombinasikan metode *exponential smoothing Holt-winters multiplicative* dengan metode lainnya.

DAFTAR PUSTAKA

- Andriyas, M. (n.d.). *Rancang Bangun Sistem Informasi Laboratorium Klinik Berbasis Web Pada Laboratorium Klinik Utama SAFIRAH Sidoarjo*. Surabaya.
- Andriyas, M. (n.d.). *Rancang Bangun Sistem Informasi Laboratorium Klinik Berbasis Web Pada Laboratorium Klinik Utama SAFIRAH Sidoarjo*. Surabaya.
- Basuki, A. (2006). *Pengembangan Perangkat Lunak*. Surabaya: PENS-ITS.
- Hakimah, M., Muhima, R. R., & Yustina, A. (2015). Rancang Bangun Aplikasi Peramalan Persediaan Barang dengan Metode Trend Projection. *Jurnal Simantec*, V(1), 37-48.
- Herjanto, E. (2008). *Manajemen Operasi*. Jakarta: Grasindo.
- Markadis, S. (2014). *Forecasting, Method and Applications*. Canada: John Wiley and Sons Inc.
- MJ, N. (2014, Maret 22). *EXPONENTIAL SMOOTHING*. Retrieved from <http://mjnurul.blogspot.co.id/2014/03/exponential-smoothing.html>
- Mulyono. (2007). Penentuan Strategik Prioritas Pelayanan Laboratorium Klinik. *Indonesian Journal of Clinical Pathology and Medical Laboratory*, XIII(2), 82-92.
- Nazim, A., & Aftharnohan, A. (2014). A comparison between single *exponential smoothing* (SES), double *exponential smoothing* (DES), holt's (brown) and adaptive response rate *exponential smoothing* (ARRES) techniques in forecasting Malaysia population. *Global Journal of Mathematical Analysis*.

- Padang, E., Tarigan, G., & Sinulingga, U. (2013). Peramalan Jumlah Penumpang Kereta Api Medan-Rantau Prapat dengan Metode Pemulusan Eksponensial *Holt-winters*. *Saintia Matematika*, *I*(2), 161-174.
- Pascapraharastyan, R. A., Supriyanto, A., & Sudarmaningtyas, P. (2014). Rancang Bangun Sistem Informasi Manajemen Arsip Rumah Sakit Bedah Surabaya Berbasis Web. *Jurnal Sistem Informasi*, *3*(1), 140-143.
- Pratiwi, W. H. (2017). SISTEM INFORMASI PERAMALAN JUMLAH PERMINTAAN DARAH MENGGUNAKAN METODE *EXPONENTIAL SMOOTHING* (STUDI KASUS : UTD PMI KABUPATEN JEMBER).
- Sukmarani, N. P. (2016). Penerapan Metode *Exponential Smoothing* pada Peramalan Penjualan dalam Penentuan Kuantitas Produksi Roti (Studi Kasus Perusahaan Roti Dhiba Kendari. *semanTIK*, *2*.
- Sulistyaningsih, K. (2003). Perancangan Sistem Informasi Pemantauan Persediaan Reagen di Laboratorium Air DIInas Kesehatan dan Kesejahteraan Sosial Kabupaten Semarang.
- Suwandi, A., Annisa, & Jaya, A. K. (2015). Peramalan Data Time Series Dengan Metode Penghalusan Eksponensial Holt - Winters.

LAMPIRAN
LAMPIRAN A. USE CASE SKENARIO

A.1 *Use Case* Skenario *Login*

Tabel A. 1.1 *Use Case Login*

ID	USC 01
Nama Use Case	<i>Login</i>
Aktor	Admin Gudang, Manajer
Entry Condition	Memasukan Username dan Password untuk masuk ke dalam sistem
Exit Condition	Aktor telah memasuki sistem
Skenario Normal <i>Login</i>	
Aksi Aktor	Reaksi Sistem
1. Memasukan username dan password	
2. Klik tombol Sign in	
	3. Eksekusi validasi username dan password di dalam database
	4. Menampilkan halaman <i>Home</i>
Skenario Alternatif Username atau Password salah	
2. Klik tombol Sign in	
	3. Menampilkan <i>form login</i> dan Menampilkan pesan “gagal <i>login</i> ”

A.2 *Use Case* Skenario Menampilkan Halaman *Home*

Tabel A. 2.1 *Use Case* Menampilkan Halaman *Home*

ID	USC 02
Nama Use Case	Menampilkan <i>Home</i>
Aktor	Admin Gudang, Manajer
Entry Condition	Sistem dalam Kondisi sesi Aktif (User telah melakukan <i>Login</i>)
Exit Condition	Menampilkan Halaman <i>Home</i>

Skenario Normal <i>Login</i>	
Aksi Aktor	Reaksi Sistem
1. Klik menu <i>Home</i>	
	2. Menampilkan halaman <i>Home</i>

A.3 Use Case Skenario Mengelola Data Reagen

Tabel A. 3.1 Use Case Skenario Mengelola Data Reagen

ID	USC 03
Nama Use Case	Mengelola Data Reagen
Aktor	Admin Gudang
Entry Condition	Memasukan data reagen baru dan mengubah data reagen
Exit Condition	Telah memasukan data reagen baru Telah mengolah data reagen
Skenario Normal Memasukan data reagen baru	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda admin
2. Memilih menu data reagen	
	3. Menampilkan halaman data reagen
4. Klik tombol “Tambah Reagen”	
	5. Menampilkn <i>form</i> reagen
6. Mengisi <i>form</i> reagen	
7. Klik tombol “Simpan”	
	8. Mengeksekusi <i>form</i> reagen
	9. Menampilkan pesan “Data Berhasil Disimpan”
10. Klik tombol “Ok”	
	11. Menampilkan halaman data reagen
Skenario Alternatif Isi <i>form</i> tidak sesuai	
8a. Klik tombol “Simpan”	
	9a. Menampilkan pesan “Lakukan pengisian dengan benar”

Skenario Normal	
Mengubah data reagen	
12. Klik tombol “Ubah”	
	13. Menampilkn <i>form</i> reagen
14. Mengisi <i>form</i> reagen	
15. Klik tombol “Simpan”	
	16. Mengeksekusi <i>form</i> reagen
	17. Menampilkan pesan “Data Berhasil Disimpan”
18. Klik tombol “Ok”	
	19. Menampilkan halaman data reagen
Skenario Alternatif	
Isi <i>form</i> tidak sesuai	
16a. Klik tombol “Simpan”	
	17a. Menampilkan pesan “Lakukan pengisian dengan benar”
Skenario Normal	
Menghapus data reagen	
20. Klik tombol “Hapus”	
	21. Menampilkn pesan “Apakah anda yakin?”
22. Klik tombol “Hapus”	
	23. Menampilkan halaman data reagen
Skenario Alternatif	
Batal menghapus data reagen	
23a. Klik tombol “Batal”	
	24a. Menampilkan halaman data reagen

A.4 Use Case Skenario Melihat Data Reagen

Tabel A. 4.1 Use Case Skenario Melihat Data Reagen

ID	USC 06
Nama Use Case	Melihat Data Reagen
Aktor	Admin Gudang
Entry Condition	Aktor melihat data reagen
Exit Condition	Aktor telah melihat data reagen
Skenario Normal	

Melihat Data Reagen	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda admin
2. Memilih menu data reagen	
	3. Menampilkan halaman data reagen
	4. Menampilkan tabel data reagen

A.5 Use Case Skenario Mengelola Transaksi Reagen Masuk

Tabel A. 5.1 Use Case Skenario Mengelola Transaksi Reagen Masuk

ID	USC 04
Nama Use Case	Mengelola Data Transaksi Reagen Masuk
Aktor	Admin Gudang
Entry Condition	Memasukan data transaksi pembelian reagen baru dan mengubah data transaksi pembelian reagen
Exit Condition	Telah memasukan data pembelian reagen baru Telah mengolah data pembelian reagen
Skenario Normal	
Memasukan data pembelian reagen baru	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda admin
2. Memilih menu data Transaksi	
	3. Menampilkan Submenu Transaksi Masuk dan Transaksi Keluar
4. Memilih submenu Transaksi Masuk	
	5. Menampilkan Halaman data transaksi reagen masuk
6. Klik tombol “Tambah Transaksi”	
	7. Menampilkn <i>form</i> tambah item dan daftar transaksi masuk

8. Mengisi <i>form</i> Tambah item	
9. Klik tombol “ <i>Submit</i> ”	
	10. Mengeksekusi <i>form</i> Tambah item
	11. Menampilkan item transaksi masuk di daftar item transaksi
12. Klik tombol “Simpan”	
	13. Menampilkan Halaman data transaksi reagen masuk
Skenario Alternatif Isi <i>form</i> tidak sesuai	
9a. Klik tombol “ <i>Submit</i> ”	
	10a. Menampilkan pesan “Lakukan pengisian dengan benar”

A.6 Use Case Skenario Melihat Transaksi Reagen Masuk

Tabel A. 6.1 Use Case Skenario Melihat Transaksi Reagen Masuk

ID	USC 08
Nama Use Case	Melihat Data Transaksi Reagen Masuk
Aktor	Admin Gudang
Entry Condition	Aktor melihat data transaksi pembelian reagen
Exit Condition	Aktor telah melihat data transaksi pembelian reagen
Skenario Normal Melihat Data Transaksi Pembelian Reagen	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda admin
2. Memilih menu data transaksi	
3. Memilih submenu transaksi Masuk	
	4. Menampilkan halaman data transaksi pembelian reagen
	5. Menampilkan tabel data transaksi pembelian reagen
6. Memilih tombol detail	

	7. Menampilkan halaman detail data transaksi pembelian reagen
--	---

A.7 Use Case Skenario Mengelola Transaksi Reagen Keluar

Tabel A. 7.1 Use Case Skenario Mengelola Transaksi Reagen Keluar

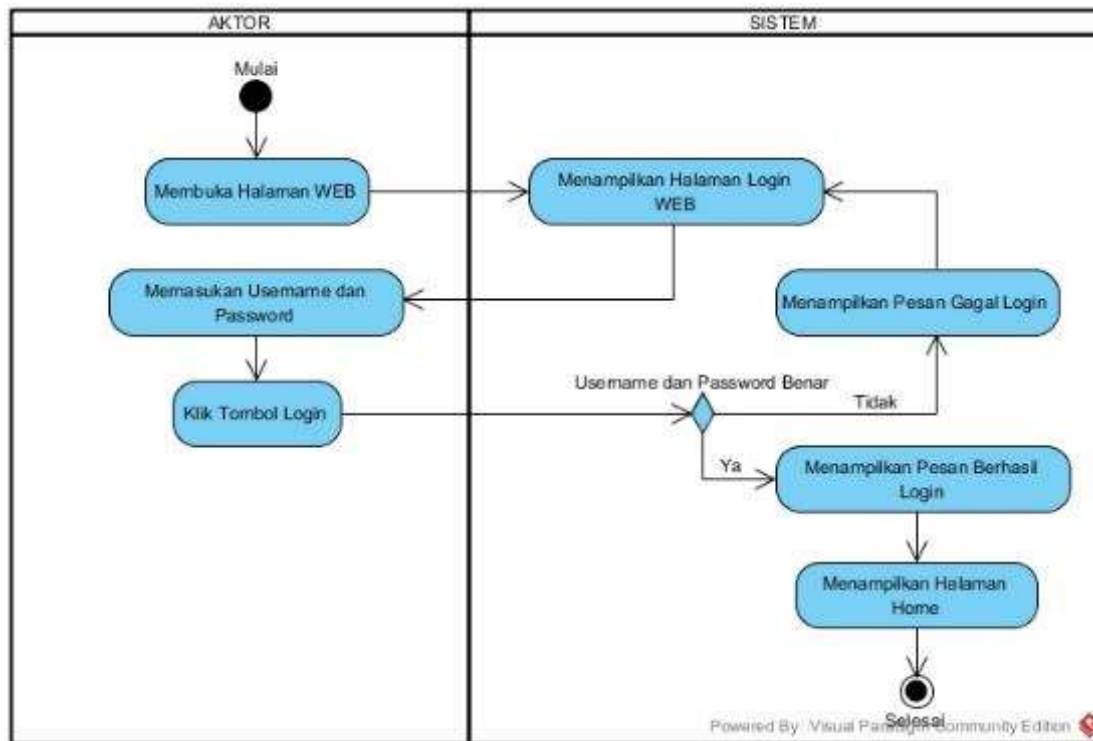
ID	USC 05
Nama Use Case	Mengolah Data Transaksi Reagen Keluar
Aktor	Admin Gudang
Entry Condition	Memasukan data transaksi pemakaian reagen baru dan mengubah data transaksi pemakaian reagen
Exit Condition	Telah memasukan data pemakaian reagen baru Telah mengolah data pemakaian reagen
Skenario Normal	
Memasukan data pembelian reagen baru	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman <i>Home</i> admin
2. Memilih menu data Transaksi	
	3. Menampilkan Submenu Transaksi Masuk dan Transaksi Keluar
4. Memilih submenu Transaksi Keluar	
	5. Menampilkan Halaman data transaksi reagen keluar
6. Klik tombol “Tambah Transaksi”	
	7. Menampilkn <i>form</i> tambah item dan daftar transaksi masuk
8. Mengisi <i>form</i> Tambah item	
9. Klik tombol “ <i>Submit</i> ”	
	10. Mengeksekusi <i>form</i> tambah item
	11. Menampilkan item transaksi keluar di daftar item transaksi
12. Klik tombol “Simpan”	

	13. Menampilkan Halaman data transaksi reagen keluar
Skenario Alternatif Isi form tidak sesuai	
9a. Klik tombol “ <i>Submit</i> ”	
	10a. Menampilkan pesan “Lakukan pengisian dengan benar”

A.8 Use Case Skenario Melihat Transaksi Reagen Keluar

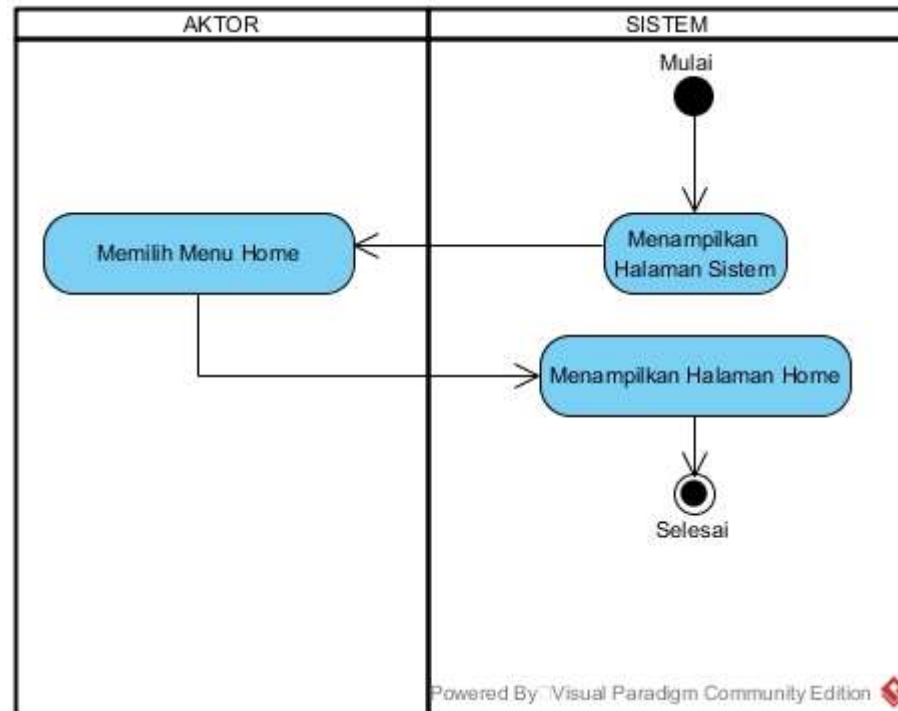
Tabel A. 8.1 Use Case Skenario Melihat Transaksi Reagen Keluar

ID	USC 10
Nama Use Case	Melihat Data Transaksi Reagen Keluar
Aktor	Admin Gudang
Entry Condition	Aktor melihat data transaksi pemakaian reagen
Exit Condition	Aktor telah melihat data transaksi pemakaian reagen
Skenario Normal Melihat Data Transaksi Pemakaian Reagen	
Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman beranda admin
2. Memilih menu data transaksi	
3. Memilih submenu pemakaian	
	4. Menampilkan halaman data transaksi pemakaian reagen
	5. Menampilkan tabel data transaksi pemakaian reagen
6. Memilih tombol detail	
	7. Menampilkan halaman detail data transaksi pemakaian reagen

LAMPIRAN B. ACTIVITY DIAGRAM**B.1 Activity Diagram Login**

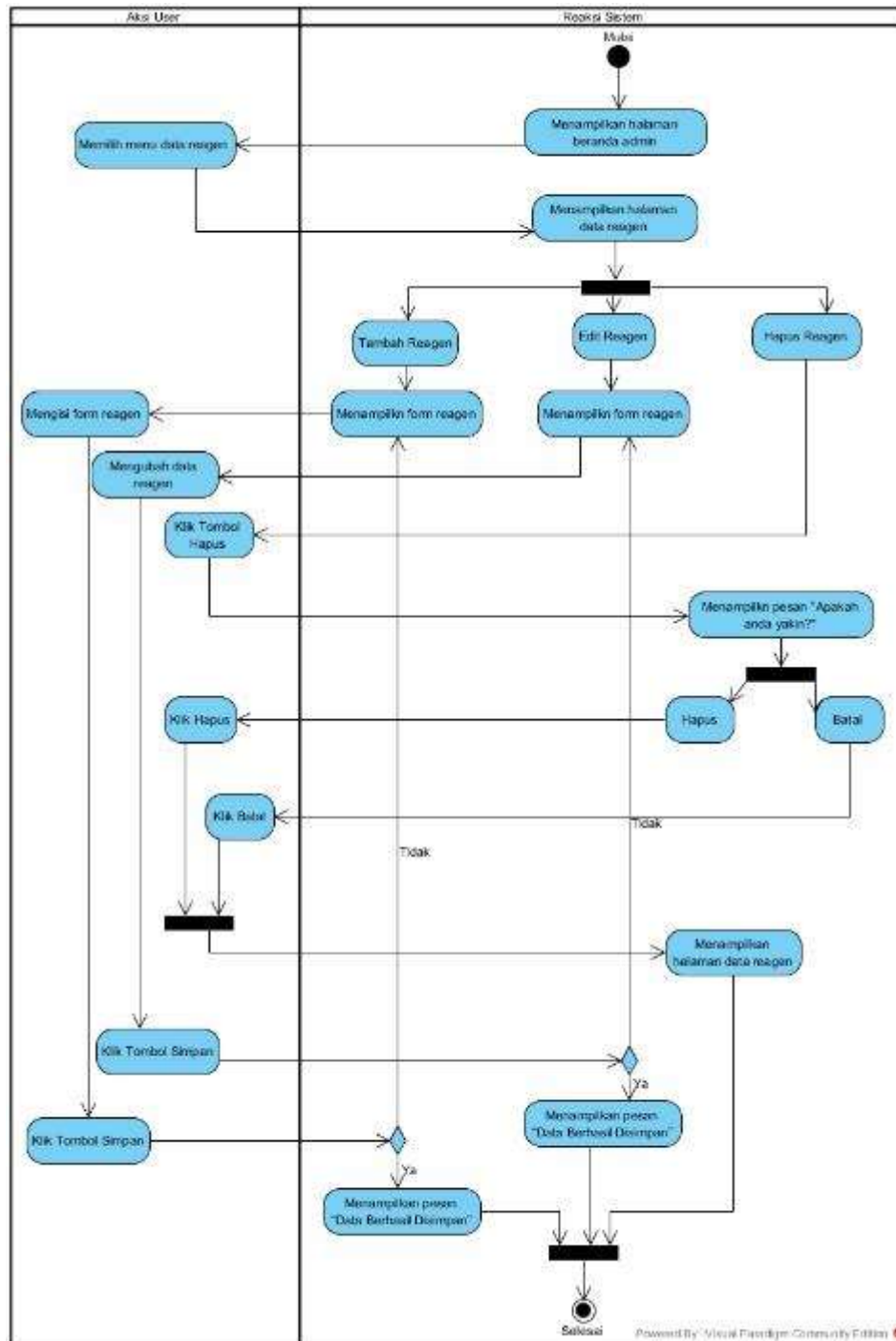
Gambar B. 1 Activity Diagram Login

B.2 Activity Diagram Menampilkan Home



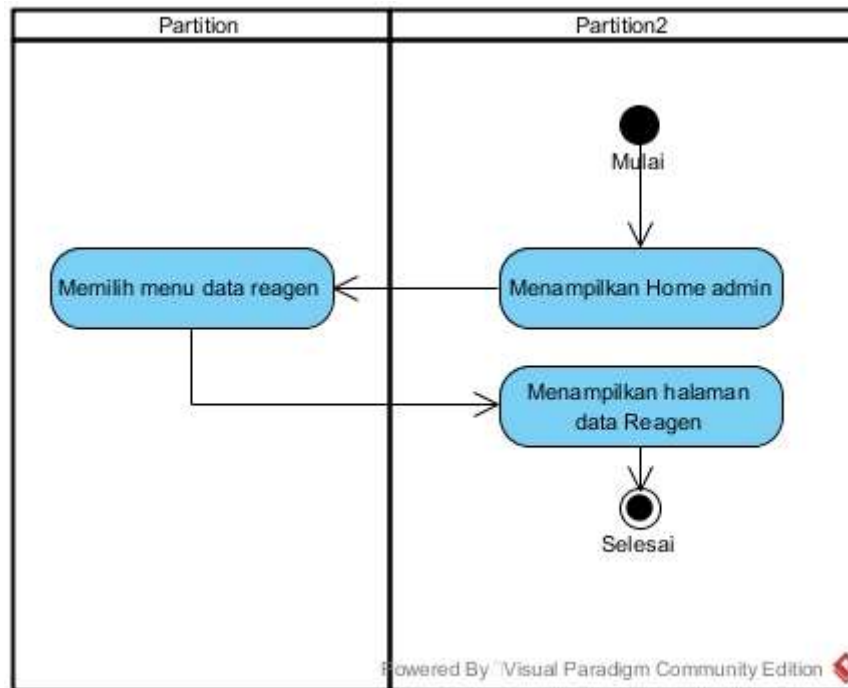
Gambar B. 2 Activity Diagram Menampilkan Home

B.3 Activity Diagram Mengelola Data Reagen



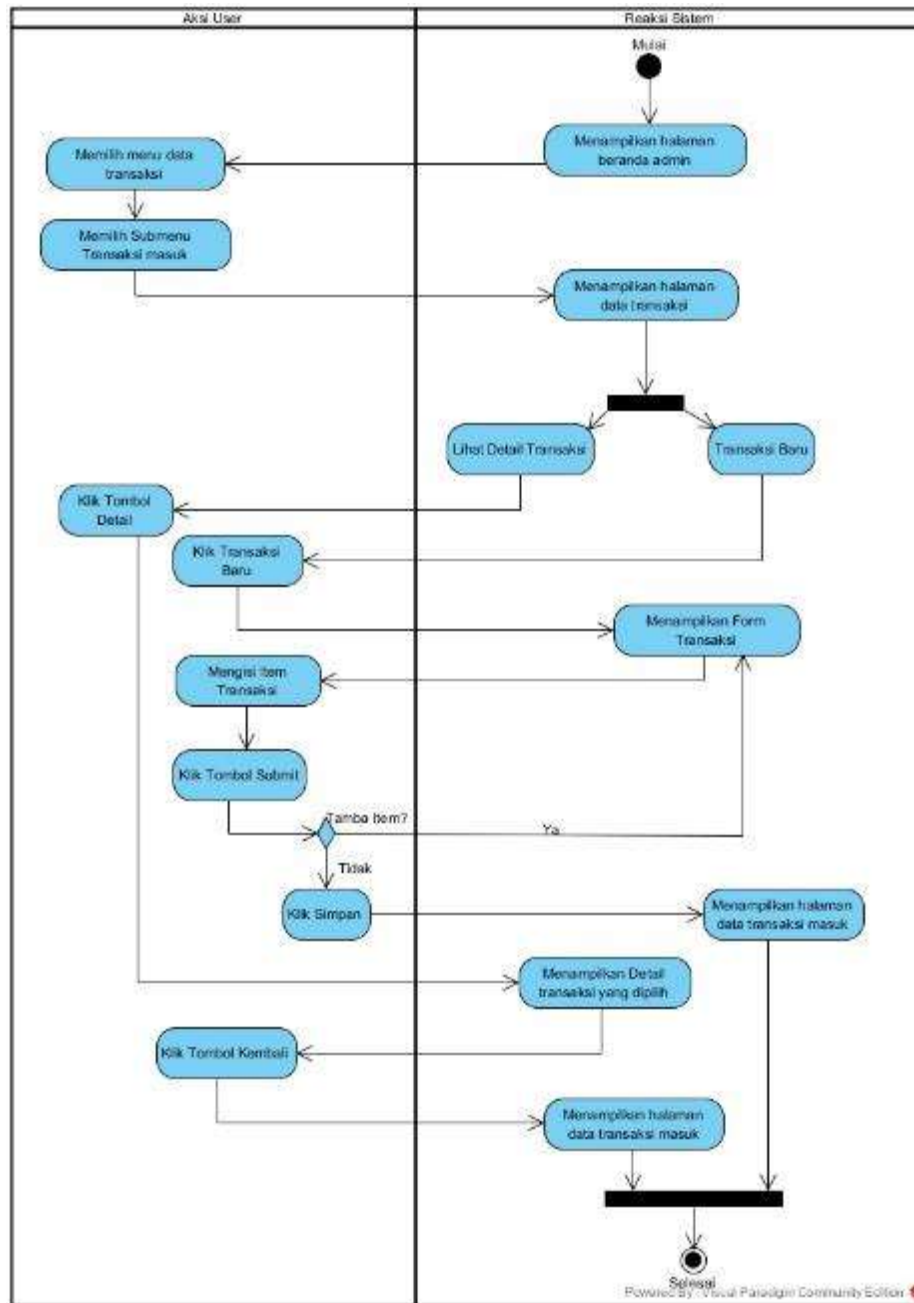
Gambar B. 3 Activity Diagram Mengelola Data Reagen

B.4 Activity Diagram Melihat Data Reagen



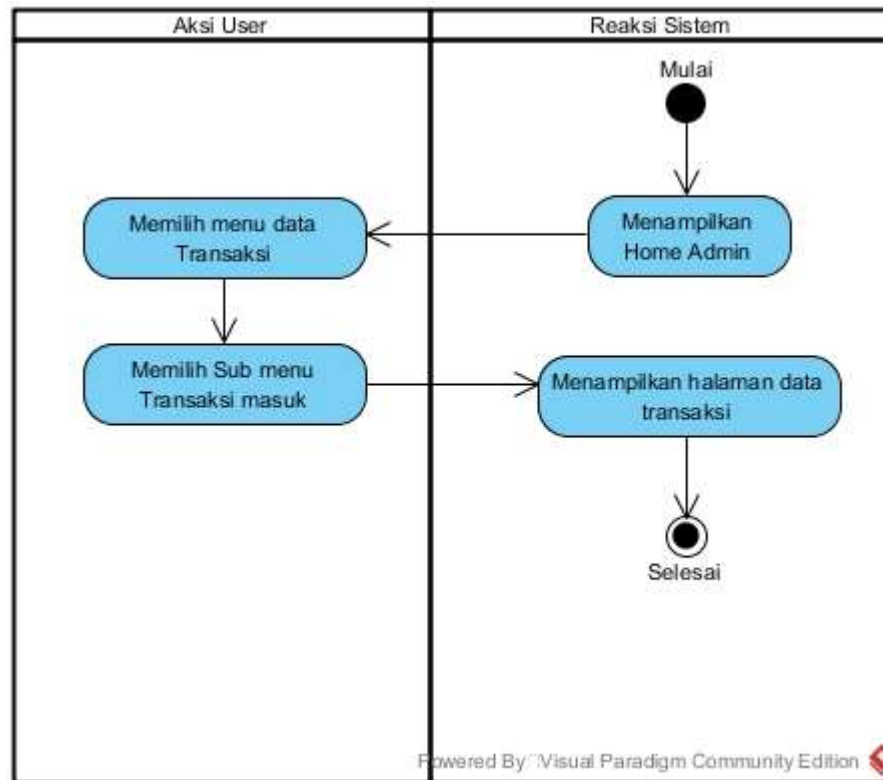
Gambar B. 4 Activity Diagram Melihat Data Reagen

B.5 Activity Diagram Mengelola Transaksi Reagen Masuk



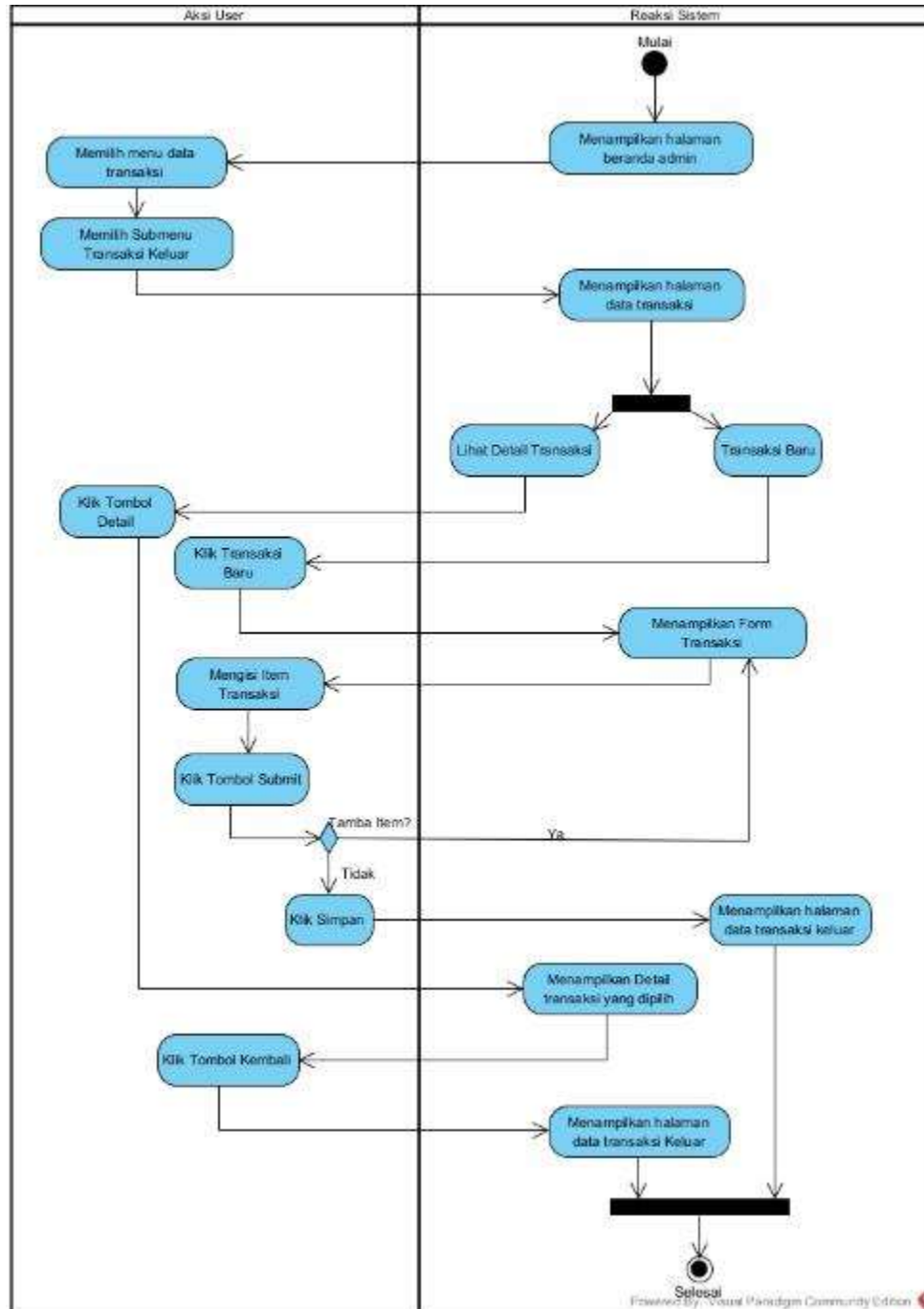
Gambar B. 5 Activity Diagram Mengelola Transaksi Reagen Masuk

B.6 Activity Diagram Melihat Transaksi Reagen Masuk



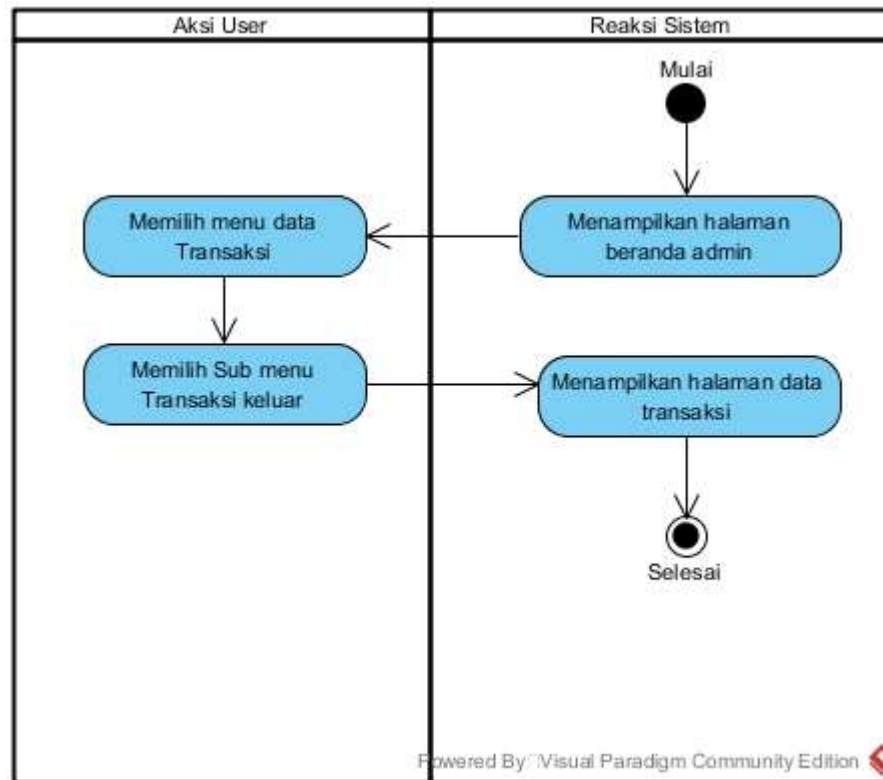
Gambar B. 6 Activity Diagram Melihat Transaksi Reagen Masuk

B.7 Activity Diagram Mengelola Transaksi Reagen Keluar



Gambar B. 7 Activity Diagram Mengelola Transaksi Reagen Keluar

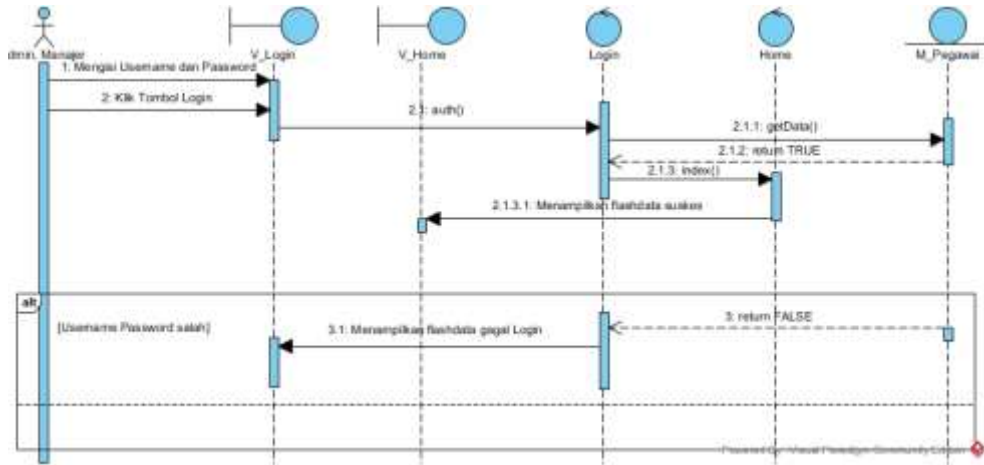
B.8 Activity Diagram Melihat Transaksi Reagen Keluar



Gambar B. 8 Activity Diagram Melihat Transaksi Reagen Keluar

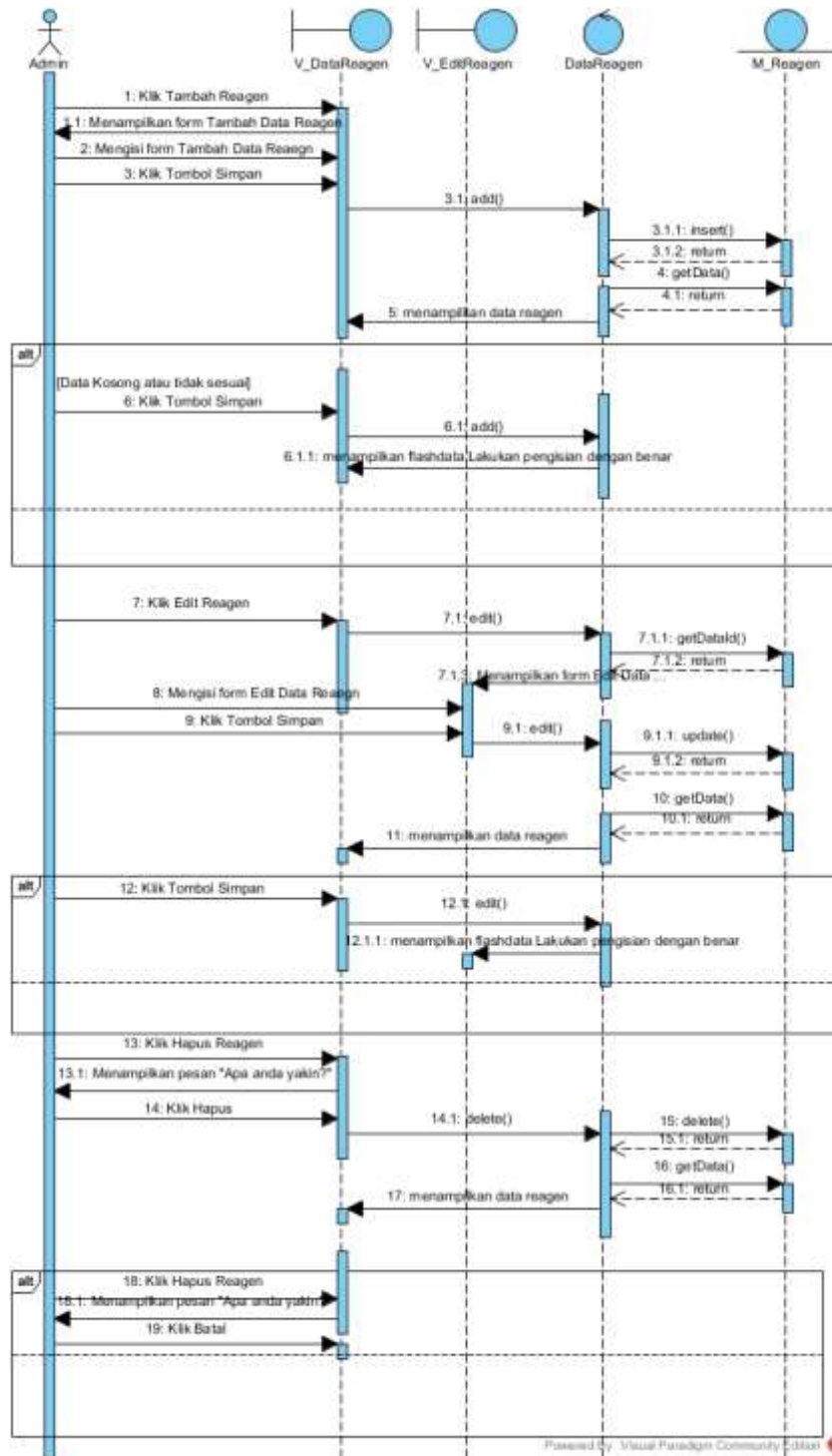
LAMPIRAN C. SEQUENCE DIAGRAM

C.1 Sequence Diagram Login dan Menampilkan Home



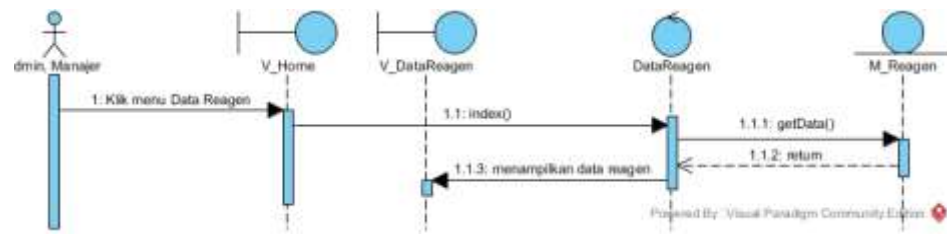
Gambar C. 1 Sequence Diagram Login dan Menampilkan Home

C.2 Sequence Diagram Mengelola Data reagen



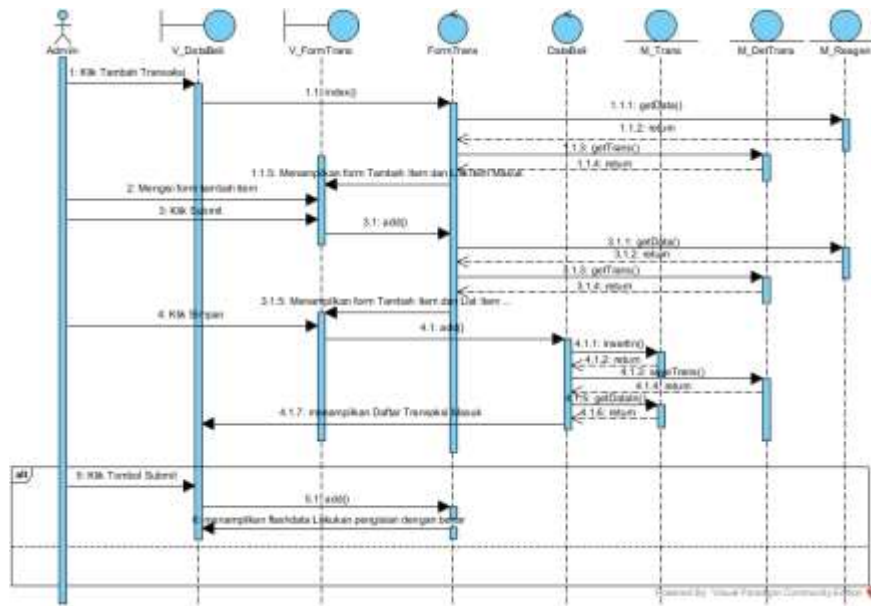
Gambar C. 2 Sequence Diagram Mengelola Data Reagen

C.3 Sequence Diagram Melihat Data Reagen



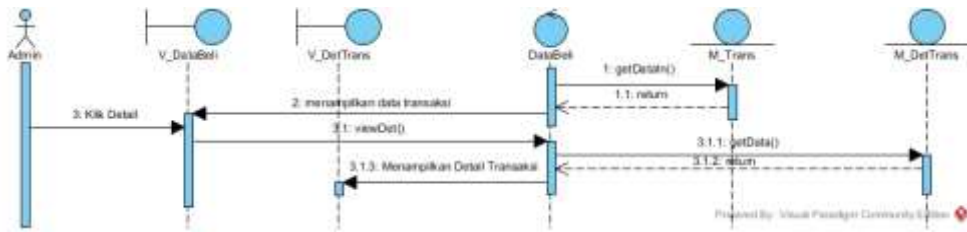
Gambar C. 3 Sequence Diagram Melihat data Reagen

C.4 Sequence Diagram Mengelola Transaksi Reagen Masuk



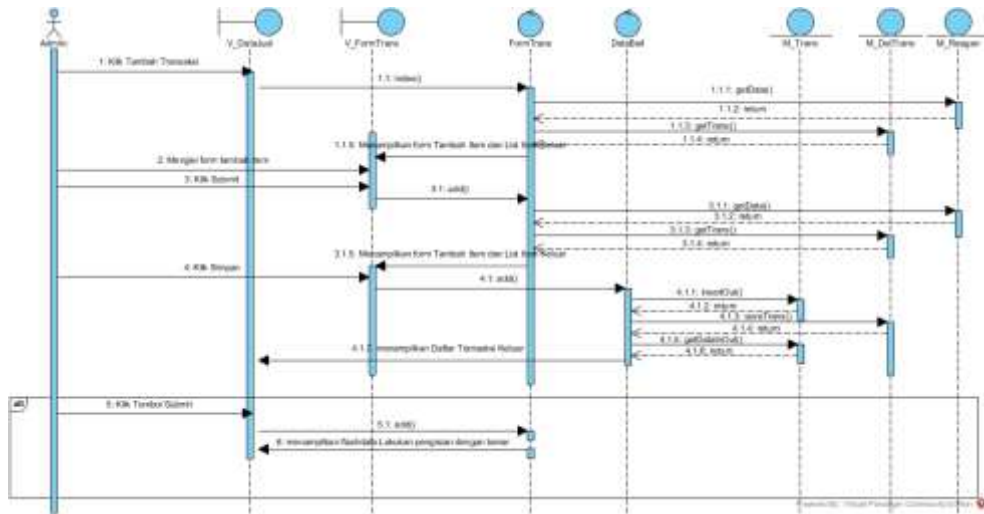
Gambar C. 4 Sequence Diagram Mengelola Transaksi Reagen Masuk

C.5 Sequence Diagram Melihat Transaksi Reagen Masuk



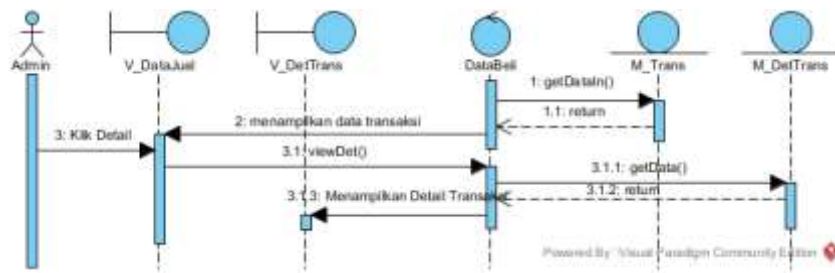
Gambar C. 5 Sequence Diagram Melihat Transaksi Reagen Masuk

C.6 Sequence Skenario Mengelola Transaksi Reagen Keluar



Gambar C. 6 Sequence Diagram Mengelola Transaksi Reagen Keluar

C.7 Sequence Skenario Melihat Transaksi Reagen Keluar



Gambar C. 7 Sequence Diagram Melihat Transaksi Reagen Keluar

LAMPIRAN D. KODE PROGRAM

D.1 Kode Program *Login*

Tabel D. 1 Kode Program *Login* (*class controllers/Login*)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->load->view('V_Login'); } else{ \$this->session->set_flashdata('sukses', 'Selamat datang '.\$pegLogin['nama'].' anda login sebagai '.\$pegLogin['level']); \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load->view('admin/V_home'); \$this->load->view('admin/footer'); } } public function auth() { \$user = \$this->input->post('username'); \$pass = \$this->input->post('password'); \$hasil = \$this->M_Pegawai->getData(\$user, md5(\$pass))->result_array(); if (empty(\$hasil)) { \$this->session->set_flashdata('gagal', 'Username atau Password salah'); \$this->load->view('V_Login'); } else { \$session = array('username' => \$hasil[0]['username'], 'nama' => \$hasil[0]['nama_peg'], 'level' => \$hasil[0]['akses']); \$this->session->set_userdata('pegLogin', \$session); \$pegLogin = \$this->session- >userdata('pegLogin'); if (\$pegLogin['level']=='Administrator') { </pre>

```

        $this->session->set_flashdata('sukses',
'Selamat datang '.$pegLogin['NIP'].' '.$pegLogin['nama'].'
anda mengakses sebagai '.$pegLogin['level'].'');
        redirect(base_url().'index.php/Home');
    }
    else{
        $this->session->set_flashdata('sukses',
'Selamat datang '.$pegLogin['NIP'].' '.$pegLogin['nama'].'
anda mengakses sebagai '.$pegLogin['level'].'');
        redirect(base_url().'index.php/Home');
    }
}
}
}

```

Tabel D. 2 Kode Program *Login (class models/M_Pegawai)*

KODE
<pre> public function get_data(\$id = null){ if(empty(\$id)){ // jika id kosong maka get semua data return \$this->db->get('tb_pegawai')- >result_array(); }else{ // jika id tidak kosong, maka get data untuk id tersebut (row_array digunakan untuk mengambil data indeks pertama) \$this->db->where('username', \$id); return \$this->db->get('tb_pegawai')- >row_array(); } } # Menginputkan \$data kedalam tabel public function insert(\$data){ return \$this->db->insert('tb_pegawai', \$data); } # Mengupdate data dengan array: \$data pada id baris: \$id public function update(\$data, \$id){ \$this->db->where('usermae', \$id); return \$this->db->update('tb_pegawai', \$data); } # delete data pada id tertentu </pre>

```
public function delete($id){  
    $this->db->where('username', $id);  
    return $this->db->delete('tb_pegawai');  
}
```

D.2 Kode Program Menampilkan *Home* (class *controllers/Home*)Tabel D. 3 Kode Program Menampilkan *Home* (class *controllers/Home*)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); \$x['data']=\$this->M_DetTrans->lastTrans(); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ \$this->session->set_flashdata('sukses', 'Selamat datang '.\$pegLogin['nama'].' anda login sebagai '.\$pegLogin['level']); \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load->view('admin/V_Home', \$x); \$this->load->view('admin/footer_pick'); \$this->load->view('admin/footer'); } } </pre>

D.3 Kode Program Melihat dan Mengola Data Reagen (*class controllers/DataReagen*)

Tabel D. 4 Kode Program Melihat dan Mengola Data Reagen (*class controllers/DataReagen*)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ // \$data['reagen']=\$this->M_Reagen- >getData()->result_array(); \$this->session->set_flashdata('sukses', \$pegLogin['nama'].' anda login sebagai '. \$pegLogin['level']); \$jumlah_data = \$this->M_Reagen- >jumlah_data(); \$this->load->library('pagination'); \$config['base_url'] = base_url(). 'index.php/DataReagen/index/'; \$config['total_rows'] = \$jumlah_data; \$config['per_page'] = 5; \$from = \$this->uri->segment(3); \$this->pagination->initialize(\$config); \$data['reagen'] = \$this->M_Reagen- >getData(\$config['per_page'], \$from); if (\$pegLogin['level'] == 'Administrator') { \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load- >view('admin/reagen/V_DataReagen', \$data); \$this->load->view('admin/footer'); \$this->load->view('admin/footer_Alert '); } else{ \$this->load->view('manajer/header'); </pre>

```

        $this->load->view('manajer/menubar',
$pegLogin);
        $this->load-
>view('manajer/reagen/V_DataReagen', $data);
        $this->load->view('manajer/footer');
        $this->load->view('manajer/footer_Alert
');
    }
}
}

public function submit() {
    $pegLogin = $this->session->userdata('pegLogin');
    $config = array(
        array(
            'field'=>'namerea',
            'label'=>'Nama Reagen ahaha',
            'rules'=>'required'
        ),
        array(
            'field'=>'satrea',
            'label'=>'Satuan Reagen',
            'rules'=>'required'
        )
    );
    $this->form_validation->set_rules($config);
    if ($this->form_validation->run()) {
        $data=array(
            'kode_reag'=>$this->input->post('id'),
            'nama_reag'=> $this->input-
>post('namerea'),
            'keterangan_reag'=> $this->input-
>post('ketrea'),
            'satuan'=> $this->input->post('satrea')
        );
        $id=$data['kode_reag'];
        unset($data['kode_reag']);
        if ($id == 0) {
            $query = $this->M_Reagen-
>insert($data);
            // Jika tambah query berhasil, maka return 1
            if( $query === TRUE) {
                // Setup session pesan tambah jika
berhasil
                $this->session-
>set_flashdata('sinput', 'Data Berhasil Ditambahkan');

```

```

        } else {
            // Setup session pesan tambah jika
gagal
                $this->session->
>set_flashdata('ginput', 'Data Gagal Ditambahkan');
            }
        } else {
            $query = $this->M_Reagen->update($data,
$cid);
            // Jika tambah query berhasil, maka return 1
            if( $query === TRUE) {
                // Setup session pesan tambah jika
berhasil
                    $this->session->
>set_flashdata('sinput', 'Data Berhasil Ditambahkan');
            } else {
                // Setup session pesan tambah jika
gagal
                    $this->session->
>set_flashdata('ginput', 'Data Gagal Ditambahkan');
            }
        }
        }
        redirect('DataReagen');
    }
    else{
        redirect('DataReagen');
    }
}

public function get_json($id){
    $data = $this->M_Reagen->getDataId($id);
    echo json_encode($data);
}

public function delete($id){
    $query = $this->M_Reagen->delete($id);
    // Jika delete query berhasil, maka return 1
    if($query == 1) {
        // Setup session pesan delete jika berhasil
        $this->session->set_flashdata('sukses',
'Data Berhasil dihapus');
    } else {
        // Setup session pesan delete jika gagal
        $this->session->set_flashdata('gagal', 'Data
Gagal DIhapus');
    }
}

```

```

    }
    // Redirect ke halaman list, dilengkapi dengan
    session data
    redirect('DataReagen');
}

```

Tabel D. 5 Kode Program Melihat dan Mengola Data Reagen (*class models/M_Reagen*)

KODE
<pre> public function get(){ return \$this->db->get('tb_reagen'); } public function getQty(\$id){ \$this->db->select('stock'); \$this->db->where('kode_reag', \$id); return \$this->db->get('tb_reagen'); } public function getData(\$number, \$offset){ return \$query = \$this->db- >get('tb_reagen',\$number, \$offset)->result_array(); // return \$this->db->get('tb_reagen'); } public function insert(\$data){ return \$this->db->insert('tb_reagen', \$data); } public function update(\$data, \$id){ \$this->db->where('kode_reag', \$id); return \$this->db->update('tb_reagen', \$data); } public function updateStock(\$id, \$qty){ \$this->db->set('stock', \$qty); \$this->db->where('kode_reag', \$id); return \$this->db->update('tb_reagen'); } public function delete(\$id){ \$this->db->where('kode_reag', \$id); return \$this->db->delete('tb_reagen'); } public function getDataId(\$id){ \$this->db->where('kode_reag', \$id); return \$this->db->get('tb_reagen')->row(); } public function jumlah_data(){ return \$this->db->get('tb_reagen')->num_rows(); } </pre>

D.4 Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk (*class controllers/DataBeli*)

Tabel D. 6 Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk (*class controllers/DataBeli*)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ // \$data['reagen']=\$this->M_Reagen- >getData()->result_array(); // \$data['detail']=\$this->M_Trans- >getDataOut()->result_array(); \$this->session->set_flashdata('sukses', \$pegLogin['nama'].' anda login sebagai '. \$pegLogin['level']); \$jumlah_data = \$this->M_Trans- >jumlah_dataIn(); \$this->load->library('pagination'); \$config['base_url'] = base_url(). 'index.php/DataBeli/index/'; \$config['total_rows'] = \$jumlah_data; \$config['per_page'] = 6; \$from = \$this->uri->segment(3); \$this->pagination->initialize(\$config); \$data['detail'] = \$this->M_Trans- >getDataIn(\$config['per_page'], \$from); if (\$pegLogin['level'] == 'Administrator') { \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load- >view('admin/pembelian/V_DataPakai', \$data); \$this->load->view('admin/footer'); \$this->load->view('admin/footer_Alert '); } </pre>

```

                else{
                    $this->load->view('manajer/header');
                    $this->load->view('manajer/menubar',
$pegLogin);
                    $this->load-
>view('manajer/pembelian/V_DataPakai', $data);
                    $this->load->view('manajer/footer');
                    $this->load->view('manajer/footer_Alert
');
                }
            }
        }
        public function viewDet() {
            $id = $this->input->post('id');
            $data = $this->M_DetTrans->getDataTabel($id)-
>result_array();
            echo json_encode($data);
        }
        public function add() {
            $pegLogin = $this->session->userdata('pegLogin');
            if (empty($pegLogin)) {
                $this->session->set_flashdata('gagal',
'Silahkan Login');
                redirect(base_url());
            }
            else{
                $this->M_Trans->insert();
                $this->M_DetTrans->saveTrans();
                $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
                $this->session->set_flashdata('sukses',
$pegLogin['nama'].' anda login sebagai
'.$pegLogin['level']);
                if ($pegLogin['level'] == 'Administrator') {
                    redirect('DataBeli');
                }
                else{
                    $this->load->view('manajer/header');
                    $this->load->view('manajer/menubar',
$pegLogin);
                    $this->load-
>view('manajer/V_DataReagen', $data);
                    $this->load->view('manajer/footer');
                }
            }
        }
    }
}

```

```

public function edit($id){
    $pegLogin = $this->session->userdata('pegLogin');
    $config = array(
        array(
            'field'=>'koderea',
            'label'=>'Kode Reagen',
            'rules'=>'required'
        ),
        array(
            'field'=>'namerea',
            'label'=>'Nama Reagen',
            'rules'=>'required'
        )
    );
    $this->form_validation->set_rules($config);
    if ($this->form_validation->run()) {
        $data=array(
            'kode_reag'=> $this->input-
>post('koderea'),
            'nama_reag'=> $this->input-
>post('namerea'),
            'keterangan_reag'=> $this->input-
>post('ketrea')
        );
        $query = $this->M_Reagen->update($data,
        $id);
        // Jika tambah query berhasil, maka return 1
        if( $query === TRUE) {
            // Setup session pesan tambah jika
berhasil
            $this->session->set_flashdata('sinput',
'Data Berhasil Ditambahkan');
        } else {
            // Setup session pesan tambah jika
gagal
            $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
        }
        redirect('DataReagen');
    }
    else{
        if (empty($pegLogin)) {
            $this->session->set_flashdata('gagal',
'Silahkan Login');
            redirect(base_url());
        }
        else{

```

```

        $data['reagen']=$this->M_Reagen-
>getDataId($id)->result_array();
        // $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
        // $this->session-
>set_flashdata('sukses', $pegLogin['nama'].' anda login
sebagai '.$pegLogin['level']);
        if ($pegLogin['level'] ==
'Administrator') {
                $this->load-
>view('admin/header');
                $this->load-
>view('admin/menubar', $pegLogin);
                $this->load-
>view('admin/reagen/V_EditReagen',$data);
                $this->load-
>view('admin/footer');
        }
        else{
                // $this->session-
>set_flashdata('sukses', 'Selamat datang
'.$pegLogin['nama'].' anda login sebagai
'.$pegLogin['level']);
                $this->load-
>view('manajer/header');
                $this->load-
>view('manajer/menubar', $pegLogin);
                $this->load-
>view('manajer/V_DataReagen', $data);
                $this->load-
>view('manajer/footer');
        }
    }
}

public function delete($id){
    $query = $this->M_Trans->delete($id);
    redirect('DataBeli');
}

```

Tabel D. 7 Kode Program Melihat dan Mengelola Data Transaksi Reagen Masuk
(class controllers/ Form TransIn)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ \$data['reagen']=\$this->M_Reagen->get()- >result_array(); \$this->session->set_flashdata('sukses', \$pegLogin['nama'].' anda login sebagai '. \$pegLogin['level']); if (\$pegLogin['level'] == 'Administrator') { \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load- >view('admin/pembelian/V_form Trans', \$data); \$this->load->view('admin/footer'); } else{ \$this->load->view('manajer/header'); \$this->load->view('manajer/menubar', \$pegLogin); \$this->load- >view('manajer/pembelian/V_form Trans', \$data); \$this->load->view('manajer/footer'); } } } public function add() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ \$this->M_Trans->insertIn(); } } </pre>

```

        $sid = $this->db->insert_id();
        $nm = $this->input->POST('postnamerea');
        $data2 = array();
        foreach ($nm as $key => $value){
            $data2[] = array(
                'id_trans' => $sid,
                'kode_reag' =>
$_POST['postnamerea'][$key],
                'qty' => $_POST['postqty'][$key]
            );
        }
        for ($i=0; $i < count($data2); $i++) {
            $stock = $this->M_Reagen-
>getQty($data2[$i]['kode_reag'])->result_array();
            $query = $this->M_DetTrans-
>insert($data2[$i]);
            $hasil = $stock[0]['stock'] +
            $data2[$i]['qty'];
            $this->M_Reagen-
>updateStock($data2[$i]['kode_reag'], $hasil);
        }
        redirect('DataBeli');
    }

    public function edit($id){
        $pegLogin = $this->session->userdata('pegLogin');
        $config = array(
            array(
                'field'=>'koderea',
                'label'=>'Kode Reagen',
                'rules'=>'required'
            ),
            array(
                'field'=>'namerea',
                'label'=>'Nama Reagen',
                'rules'=>'required'
            )
        );
        $this->form_validation->set_rules($config);

        if ($this->form_validation->run()) {
            $data=array(
                'kode_reag'=> $this->input-
>post('koderea'),
                'nama_reag'=> $this->input-
>post('namerea'),

```

```

        'keterangan_reag'=> $this->input-
>post('ketrea')
        );
        $query = $this->M_Reagen->update($data,
$Id);
        // Jika tambah query berhasil, maka return 1
        if( $query === TRUE) {
            // Setup session pesan tambah jika
berhasil
            $this->session->set_flashdata('sinput',
'Data Berhasil Ditambahkan');
        } else {
            // Setup session pesan tambah jika
gagal
            $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
        }
        redirect('DataReagen');
    }
    else{
        if (empty($pegLogin)) {
            $this->session->set_flashdata('gagal',
'Silahkan Login');
            redirect(base_url());
        }
        else{
            $data['reagen']=$this->M_Reagen-
>getDataId($Id)->result_array();
            if ($pegLogin['level'] ==
'Administrator') {
                $this->load-
>view('admin/header');
                $this->load-
>view('admin/menubar', $pegLogin);
                $this->load-
>view('admin/reagen/V_EditReagen',$data);
                $this->load-
>view('admin/footer');
            }
            else{
                $this->load-
>view('manajer/header');
                $this->load-
>view('manajer/menubar', $pegLogin);
                $this->load-
>view('manajer/V DataReagen', $data);

```

```
                                $this->load-  
>view('manajer/footer');  
                                }  
                                }  
                                }  
  
    public function delete($id){  
        $query = $this->M_DetTrans->delete($id);  
        // Jika delete query berhasil, maka return 1  
        if( $query == 1) {  
            // Setup session pesan delete jika berhasil  
            $this->session->set_flashdata('sukses',  
'Data Berhasil dihapus');  
        } else {  
            // Setup session pesan delete jika gagal  
            $this->session->set_flashdata('gagal', 'Data  
Gagal DIhapus');  
        }  
        // Redirect ke halaman list, dilengkapi dengan  
session data  
        redirect('DataBeli');  
    }  
}
```


D.5 Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar (*class controllers/DataPakai*)

Tabel D. 8 Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar (*class controllers/DataPakai*)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } else{ // \$data['reagen']=\$this->M_Reagen- >getData()->result_array(); // \$data['detail']=\$this->M_Trans- >getDataOut()->result_array(); \$this->session->set_flashdata('sukses', \$pegLogin['nama'].' anda login sebagai '.'. \$pegLogin['level']); \$jumlah_data = \$this->M_Trans- >jumlah_dataOut(); \$this->load->library('pagination'); \$config['base_url'] = base_url().'index.php/DataPakai/index/'; \$config['total_rows'] = \$jumlah_data; \$config['per_page'] = 6; \$from = \$this->uri->segment(3); \$this->pagination->initialize(\$config); \$data['detail'] = \$this->M_Trans- >getDataOut(\$config['per_page'], \$from); if (\$pegLogin['level'] == 'Administrator') { \$this->load->view('admin/header'); \$this->load->view('admin/menubar', \$pegLogin); \$this->load- >view('admin/pemakaian/V_DataPakai', \$data); \$this->load->view('admin/footer'); \$this->load->view('admin/footer_Alert '); } } </pre>

```

                else{
                    $this->load->view('manajer/header');
                    $this->load->view('manajer/menuubar',
$pegLogin);
                    $this->load-
>view('manajer/pemakaian/V_DataPakai', $data);
                    $this->load->view('manajer/footer');
                    $this->load->view('manajer/footer_Alert
');
                }
            }
        }
        public function viewDet() {
            $id = $this->input->post('id');
            $data = $this->M_DetTrans->getDataTabel($id)-
>result_array();
            echo json_encode($data);
        }
        public function add() {
            $pegLogin = $this->session->userdata('pegLogin');
            if (empty($pegLogin)) {
                $this->session->set_flashdata('gagal',
'Silahkan Login');
                redirect(base_url());
            }
            else{
                $this->M_Trans->insert();
                $this->M_DetTrans->saveTrans();
                $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
                $this->session->set_flashdata('sukses',
$pegLogin['nama'].' anda login sebagai
'.$pegLogin['level']);
                if ($pegLogin['level'] == 'Administrator') {
                    redirect('DataBeli');
                }
                else{
                    $this->load->view('manajer/header');
                    $this->load->view('manajer/menuubar',
$pegLogin);
                    $this->load-
>view('manajer/V_DataReagen', $data);
                    $this->load->view('manajer/footer');
                }
            }
        }
    }
}

```

```

public function edit($id){
    $pegLogin = $this->session->userdata('pegLogin');
    $config = array(
        array(
            'field'=>'koderea',
            'label'=>'Kode Reagen',
            'rules'=>'required'
        ),
        array(
            'field'=>'namerea',
            'label'=>'Nama Reagen',
            'rules'=>'required'
        )
    );
    $this->form_validation->set_rules($config);
    if ($this->form_validation->run()) {
        $data=array(
            'kode_reag'=> $this->input-
>post('koderea'),
            'nama_reag'=> $this->input-
>post('namerea'),
            'keterangan_reag'=> $this->input-
>post('ketrea')
        );
        $query = $this->M_Reagen->update($data,
$id);
        // Jika tambah query berhasil, maka return 1
        if( $query === TRUE) {
            // Setup session pesan tambah jika
berhasil
            $this->session->set_flashdata('sinput',
'Data Berhasil Ditambahkan');
        } else {
            // Setup session pesan tambah jika
gagal
            $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
        }
        redirect('DataReagen');
    }
    else{
        if (empty($pegLogin)) {
            $this->session->set_flashdata('gagal',
'Silahkan Login');
            redirect(base_url());
        }
    }
}

```

```

        else{
            $data['reagen']=$this->M_Reagen-
>getDataId($id)->result_array();
            if ($pegLogin['level'] ==
'Administrator') {
                $this->load-
>view('admin/header');
                $this->load-
>view('admin/menubar', $pegLogin);
                $this->load-
>view('admin/reagen/V_EditReagen', $data);
                $this->load-
>view('admin/footer');
            }
            else{
                $this->load-
>view('manajer/header');
                $this->load-
>view('manajer/menubar', $pegLogin);
                $this->load-
>view('manajer/V_DataReagen', $data);
                $this->load-
>view('manajer/footer');
            }
        }
    }

    public function delete($id){
        $query = $this->M_Trans->delete($id);
        // Redirect ke halaman list, dilengkapi dengan
session data
        redirect('DataPakai');
    }

```

Tabel D. 9 Kode Program Melihat dan Mengelola Data Transaksi Reagen Keluar
(class controllers/ Form TransOut)

KODE
<pre> public function index() { \$pegLogin = \$this->session->userdata('pegLogin'); if (empty(\$pegLogin)) { \$this->session->set_flashdata('gagal', 'Silahkan Login'); redirect(base_url()); } </pre>


```

        );
    }
    for ($i=0; $i < count($data2); $i++) {
        $stock = $this->M_Reagen-
>getQty($data2[$i]['kode_reag'])->result_array();
        $query = $this->M_DetTrans-
>insert($data2[$i]);
        $hasil = $stock[0]['stock'] -
        $data2[$i]['qty'];
        $this->M_Reagen-
>updateStock($data2[$i]['kode_reag'], $hasil);
    }
    redirect('DataPakai');
}

public function edit($id){
    $pegLogin = $this->session->userdata('pegLogin');
    $config = array(
        array(
            'field'=>'koderea',
            'label'=>'Kode Reagen',
            'rules'=>'required'
        ),
        array(
            'field'=>'namerea',
            'label'=>'Nama Reagen',
            'rules'=>'required'
        )
    );
    $this->form_validation->set_rules($config);

    if ($this->form_validation->run()) {
        $data=array(
            'kode_reag'=> $this->input-
>post('koderea'),
            'nama_reag'=> $this->input-
>post('namerea'),
            'keterangan_reag'=> $this->input-
>post('ketrea')
        );
        $query = $this->M_Reagen->update($data,
        $id);

        // Jika tambah query berhasil, maka return 1
        if( $query === TRUE) {
            // Setup session pesan tambah jika
berhasil

```

```

                $this->session->set_flashdata('sinput',
'Data Berhasil Ditambahkan');
            } else {
                // Setup session pesan tambah jika
gagal
                $this->session->set_flashdata('ginput',
'Data Gagal Ditambahkan');
            }
            redirect('DataReagen');
        }
        else{
            if (empty($pegLogin)) {
                $this->session->set_flashdata('gagal',
'Silahkan Login');
                redirect(base_url());
            }
            else{
                $data['reagen']=$this->M_Reagen-
>getDataId($id)->result_array();
                if ($pegLogin['level'] ==
'Administrator') {
                    $this->load-
>view('admin/header');
                    $this->load-
>view('admin/menubar', $pegLogin);
                    $this->load-
>view('admin/reagen/V_EditReagen',$data);
                    $this->load-
>view('admin/footer');
                }
                else{
                    $this->load-
>view('manajer/header');
                    $this->load-
>view('manajer/menubar', $pegLogin);
                    $this->load-
>view('manajer/V_DataReagen', $data);
                    $this->load-
>view('manajer/footer');
                }
            }
        }
    }

    public function delete($id){

```

```
$query = $this->M_DetTrans->delete($id);
// Jika delete query berhasil, maka return 1
if( $query == 1) {
    // Setup session pesan delete jika berhasil
    $this->session->set_flashdata('sukses',
'Data Berhasil dihapus');
} else {
    // Setup session pesan delete jika gagal
    $this->session->set_flashdata('gagal', 'Data
Gagal DIhapus');
}
// Redirect ke halaman list, dilengkapi dengan
session data
redirect('DataBeli');
}
```


D.6 Kode Program *class models/M_Trans*Tabel D. 10 Kode Program *class models/M_Trans*

KODE
<pre> public function getDataOut(\$number, \$offset){ \$this->db->where('ket', 2); \$this->db->where('status', 1); \$this->db->group_by('Month(waktu_trans), Year(waktu_trans)'); \$this->db->order_by('waktu_trans', 'desc'); return \$this->db->get('tb_transaksi',\$number, \$offset)->result_array(); } public function getDataIn(\$number, \$offset){ \$this->db->where('ket', 1); \$this->db->where('status', 1); return \$this->db->get('tb_transaksi',\$number, \$offset)->result_array(); } public function insertOut(){ \$a = date('Y-m-d H:i:s'); \$data=array('id_trans' => '', 'waktu_trans' => \$a, 'ket' => '2', 'status' => '1'); return \$this->db->insert('tb_transaksi', \$data); } public function insertIn(){ \$a = date('Y-m-d H:i:s'); \$data=array('id_trans' => '', 'waktu_trans' => \$a, 'ket' => '1', 'status' => '1'); return \$this->db->insert('tb_transaksi', \$data); } public function update(\$data, \$id){ \$this->db->where('id_trans', \$id); return \$this->db->update('tb_transaksi', \$data); } public function delete(\$id){ \$this->db->where('id_trans', \$id); \$data = array(</pre>

```
        'status' => '0'
    );
    return $this->db->update('tb_transaksi', $data);
}
public function getDataId($id){
    $this->db->where('id_trans', $id);
    return $this->db->get('tb_transaksi');
}
public function jumlah_dataOut(){
    $this->db->where('ket', 2);
    $this->db->where('status', 1);
    return $this->db->get('tb_transaksi')->num_rows();
}
public function jumlah_dataIn(){
    $this->db->where('ket', 1);
    $this->db->where('status', 1);
    return $this->db->get('tb_transaksi')->num_rows();
}
```

D.7 Kode Program *class models/ M_DetTrans*Tabel D. 11 Kode Program *class models/M_DetTrans*

KODE
<pre> public function getData(\$id){ \$this->db->where('id_trans', \$id); return \$this->db->get('tb_detail_trans'); } public function getDataTabel(\$id){ \$this->db->join('tb_reagen', 'tb_detail_trans.kode_reag = tb_reagen.kode_reag'); \$this->db->where('id_trans', \$id); return \$this->db->get('tb_detail_trans'); } public function getTrans(){ \$this->db->join('tb_reagen', 'tb_detail_trans.nama_reag = tb_reagen.nama_reag','inner'); \$this->db->where('id_trans', NULL); return \$this->db->get('tb_detail_trans'); } public function insert(\$data){ return \$this->db->insert('tb_detail_trans', \$data); } public function saveTrans(){ \$this->db->select_max('id_trans'); \$sql = \$this->db->get('tb_transaksi')- >result_array(); \$data=array('id_trans' => \$sql[0]['id_trans']); \$this->db->where('id_trans', NULL); return \$this->db->update('tb_detail_trans', \$data); } public function delete(\$id){ \$this->db->where('id_detail_trans', \$id); return \$this->db->delete('tb_detail_trans'); } public function getDataId(\$id){ \$this->db->where('nama_reag', \$id); return \$this->db->get('tb_detail_trans'); } public function lastTrans(){ </pre>

```

        $query=$this->db->query("SELECT
MAX(CONCAT(YEAR(waktu_trans), '-', MONTH(waktu_trans))) as
waktu_trans, nama_reag, SUM(qty) as qty FROM tb_transaksi a
join tb_detail_trans b ON (a.id_trans=b.id_trans) JOIN
tb_reagen c ON (b.kode_reag=c.kode_reag) WHERE a.ket='2' and
a.status='1' GROUP BY b.kode_reag;");

        if ($query->num_rows() > 0) {
            foreach ($query->result() as $data) {
                $hasil[] = $data;
            }
            return $hasil;
        }
    }
}

```

D.8 Kode Program *class models/M_Pegawai*

Tabel D. 12 Kode Program *class models/M_Pegawai*

KODE
<pre> public function getData(\$username, \$password) { \$this->db->where('username', \$username); \$this->db->where('password', \$password); \$this->db->where('status', 1); return \$this->db->get('tb_pegawai'); } public function insert(\$data){ return \$this->db->insert('tb_pegawai', \$data); } public function update(\$data, \$username){ \$this->db->where('username', \$username); return \$this->db->update('tb_pegawai', \$data); } public function delete(\$username){ \$this->db->where('username', \$username); return \$this->db->delete('tb_pegawai'); } </pre>

LAMPIRAN E. PENGUJIAN BLACK BOX

No	Fitur	Kasus	Hasil	Keterangan	
				Berhasil	Tidak
1.	<i>Login (admin)</i>	Ketika klik tombol <i>login</i> dan semua <i>field</i> pada <i>form login</i> sudah terisi dengan benar	Menampilkan halaman awal admin	V	
		Ketika klik tombol <i>login</i> dan belum mengisi <i>form</i> secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan <i>Error message</i> username atau password salah	V	
2.	Tambah data <i>reagen</i>	Ketika klik tombol tambah dan semua <i>field</i> pada <i>form</i> tambah sudah terisi dengan benar	Menambah data <i>reagen</i> sesuai yang <i>diinputkan</i>	V	
		Ketika klik tombol simpan dan belum mengisi <i>form</i> tambah secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan pesan data tidak boleh kosong	V	
3.	<i>Edit data reagen</i>	Ketika klik tombol <i>update</i> dan semua <i>field</i> pada <i>form edit</i>	<i>Update data reagen</i> sesuai yang <i>diinputkan</i>	V	

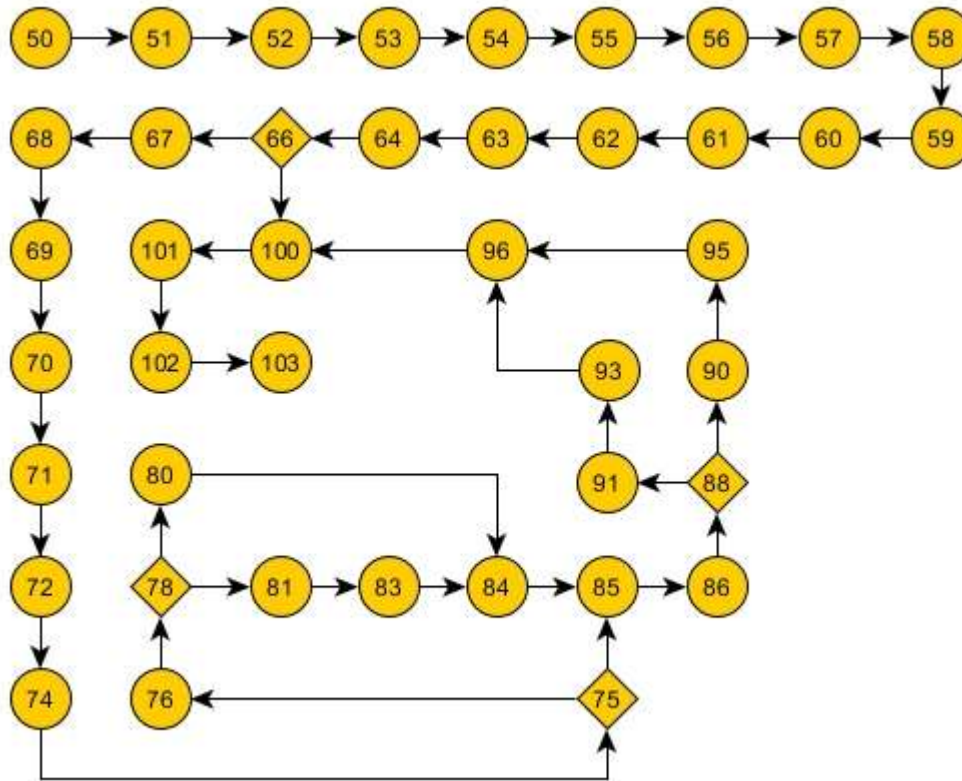
		data sudah terisi dengan benar			
		Ketika klik tombol simpan dan belum mengisi secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan pesan data tidak boleh kosong	V	
4.	Hapus data <i>reagen</i>	Ketika klik tombol delete data yang akan dihapus dan klik tombol delete pada pop – up	<i>Delete</i> data <i>reagen</i> sesuai yang dipilih oleh pakar	V	
		Ketika klik tombol <i>delete</i> dan memlih <i>cancel</i>	Menampilkan halaman sebelumnya	V	
5.	Tambah data pemakaian <i>reagen</i>	Ketika klik tombol tambah dan semua <i>field</i> pada <i>form</i> tambah data pemakaian <i>reagen</i> sudah terisi dengan benar	Menambah data pemakaian <i>reagen</i> sesuai yang diinputkan oleh admin	V	
		Ketika klik tombol simpan dan belum mengisi <i>form</i> tambah secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan pesan data tidak boleh kosong	V	

6.	<i>Edit data pemakaian reagen</i>	Ketika klik tombol <i>update</i> dan semua <i>field</i> pada <i>form edit</i> data pemakaian <i>reagen</i> sudah terisi dengan benar	<i>Update</i> data pemakaian <i>reagen</i> sesuai yang <i>diinputkan</i> oleh admin	V	
		Ketika klik tombol <i>update</i> dan belum mengisi <i>form edit</i> data pemakaian <i>reagen</i> secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan pesan data tidak boleh kosong	V	
7.	Hapus data pemakaian <i>reagen</i>	Ketika klik tombol <i>delete</i> data yang akan dihapus dan klik tombol <i>delete</i> pada <i>pop – up</i>	<i>Delete</i> data pemakaian <i>reagen</i> sesuai yang dipilih admin	V	
		Ketika klik tombol <i>delete</i> dan memilih <i>cancel</i>	Menampilkan halaman sebelumnya	V	
8.	Tambah data pembelian <i>reagen</i>	Ketika klik tombol tambah dan semua <i>field</i> pada <i>form</i> tambah data pembelian <i>reagen</i> sudah terisi dengan benar	Menambah data pembelian <i>reagen</i> sesuai yang <i>diinputkan</i> oleh admin	V	

		Ketika klik tombol simpan dan belum mengisi <i>form</i> tambah pembelian <i>reagen</i> secara lengkap / <i>field</i> masih ada yang kosong	Menampilkan pesan data tidak boleh kosong	V	
9.	Hapus data data pembelian <i>reagen</i>	Ketika klik tombol delete data yang akan dihapus dan klik tombol <i>delete</i> pada <i>pop – up</i>	<i>Delete</i> data pembelian <i>reagen</i> sesuai yang dipilih admin	V	
		Ketika klik tombol <i>delete</i> dan memlih <i>cancel</i>	Menampilkan halaman sebelumnya	V	

LAMPIRAN F. PENGUJIAN WHITE BOX

F.1 Pengujian White Box Tambah Data Reagen



Gambar F. 1 Diagram Alir *Function* Tambah Data Reagen

F.2 *function submit()* : $V(G) = E - N + 2 = 41 - 38 + 2 = 5$

Tabel F. 1 Test Case *function submit()*

<i>Test case function submit()</i>	
<i>Test case</i>	Jika <i>form</i> tambah <i>reagen</i> menampilkan <i>pop up</i> tambah data <i>reagen</i>
Target yang diharapkan	Menampilkan <i>form</i> tambah data dengan tombol <i>action</i>
Hasil pengujian	Benar
<i>Path/Jalur</i>	50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-100-101-102-103
<i>Test case function submit()</i>	

<i>Test case</i>	Jika <i>form</i> tambah <i>reagen</i> menampilkan <i>pop up</i> tambah data <i>reagen</i> dan mengisi data
Target yang diharapkan	Menampilkan <i>form</i> tambah data dengan tombol <i>action</i>
Hasil pengujian	Benar
<i>Path/Jalur</i>	50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-100-101-102-103
<i>Test case function submit()</i>	
<i>Test case</i>	Jika <i>form</i> tambah <i>reagen</i> menampilkan <i>pop up</i> tambah data <i>reagen</i> dan menyimpan data dari <i>form</i>
Target yang diharapkan	Menampilkan <i>form</i> tambah data dengan tombol <i>action</i> dan data tersimpan
Hasil pengujian	Benar
<i>Path/Jalur</i>	50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-100-101-102-103
<i>Test case function submit()</i>	
<i>Test case</i>	Jika klik <i>icon edit</i> kemudian menampilkan halaman <i>edit</i>
Target yang diharapkan	Menampilkan data yang dipilih
Hasil pengujian	Benar
<i>Path/Jalur</i>	50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-100-101-102-103
<i>Test case function submit()</i>	
<i>Test case</i>	Jika klik <i>icon edit</i> kemudian menampilkan halaman <i>edit</i> setelah itu data disimpan
Target yang diharapkan	Menampilkan <i>form edit</i> data dan data baru disimpan
Hasil pengujian	Benar
<i>Path/Jalur</i>	50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-66-100-101-102-103